Building Mobile Mashup Applications. Some Challenges Encountered in Computer Science Degrees

Javier Díaz[1], **Claudia A. Queiruga**[1] , **Pablo J. Iuliano**[1] , **Jorge H. Rosso**[1] , **Isabel Kimura**[1], **Matías Brown Barnetche**[1]

[1]LINTI, Computer Science School, Universidad Nacional de La Plata, La Plata, Buenos Aires, Argentina

Abstract - the utilization and development of innovative technologies are suitable elements to take into account in Computer Science degrees. Modern technologies must be incorporated to portray the fundamental concepts and, in this way, to familiarize students with their future workplaces. The development of mashups for mobile devices turns out to be an appealing topic of great interest for students that attend technology and computer science courses. It represents a challenge when it comes to develop software for platforms with peculiarities such as screen and hardware restrictions, and with new advantageous features like sensors of movement, GPS and camera. In this article, it is described the experience in the development of mobile mashups applications written in JAVA in its two platforms: JME and Android. This work is the result of an articulation between a research project on a mobile application testbed and a Computer Science School subject in which the students obtain new skills in the development of software.

Keywords: mashups, Android, JME, education in computer science

1 Introduction

A mashup application uses and combines data, presentations and functions from more than one source in order to create new services. The term mashup implies easy and fast integration, often using open APIs and data sources to produce enriched results different from the application's original purpose. This kind of application is predominantly user oriented, as it focuses mainly on user-friendly features, simplicity and usability [1]. Likewise, it is more oriented towards integration with existing applications, than to the software developing process [2].

In the software development process, integration plays a key role. Aside from functional requirements, developers must take into account requirements related to communication and services their tool could offer. Facebook is one of the most popular social networks, allowing users to share photos, personal information and preferences. It offers an open and easily accessible API which allows for the creation of integrated applications and mashup applications that consume the services Facebook provides. Google Maps is another example of a system that offers the possibility of creating enriched applications with maps through an extensive API that is adapted to mobile devices, as well as desktop and web applications.

Today, mobile technology makes it very easy to collect and share information about ourselves and our environment. This, together with uninterrupted Internet access, the advent of the semantic web, 2.0 web and service-oriented web, has resulted in a reformulation of the way in which we communicate, interact and socialize.

This paper describes our experience in the development of mobile mashup applications written in Java, to address local issues. This work is the result of an articulation between a research project on a mobile application testbed and a Computer Science School subject, in which the students acquire new skills in software development, by taking into consideration matters such as user-oriented design, mobility and integration.

2 Motivation

Teaching a new mobile software development paradigm based on open APIs and third-party service integration is a great challenge to Computer Science teachers. The topic is motivating and state-of-the-art. It poses interesting challenges, such as entirely different development and execution platforms, and requires mobile device emulation tools. It is fundamental for the developer to be familiar with the specific characteristics of each device (operating system, type of keyboard, camera, GPS, Internet access). Manufacturers releases SDKs (Software Development Kits) for developers to use.

There is a widespread assumption that developing mobile applications is equivalent to developing traditional, but “smaller”, applications. The real challenge posed by taking a project to a mobile device is providing a valuable experience in a generally small interface that allows an entirely different kind of interaction from desktop applications.

When developing mobile applications, it is necessary to establish whether they will be aimed at low-end, mid-range or high-end devices, and what operating system they will feature, since operating capacity, display size and a number of other features play important roles in the design.

Important changes in today's society force us to think of new ways to “reach out” to users. Web access is increasingly

aimed at obtaining information about recent events (online news, Craigslist), expressing personal opinions (Twitter and blogs), gaining access to media files (YouTube and Flickr) and creating communities and social networks (MySpace, Facebook, LinkedIn) [3]. Although all these activities require a computer with Internet access, they are being increasingly accessed through mobile devices, which are always on. Worldwide, the ubiquity of these devices makes them the main, and sometimes only, interaction with computers.

Integration is now quite common in software development, allowing projects to generate or gather data from open services such as Facebook, Twitter, YouTube, Vimeo, Flickr, Google Maps, Google Search, Google Analytics and others. Mashups allow for fast data and process integration. If we also integrate the services in a mobile application, we can obtain applications that "go everywhere" with the user, accessible to everyone and as rich as those found on the web. A combination of the information obtained from different sources with the information obtained from the context through sensors and other components present in current mobile device generations provide a richer basis for the development of mashup applications.

3 Why Teach Mashups?

The relevance acquired in the last few years by mashups and other web 2.0 technologies, together with the fact that our students are digital natives and users of social networks such as Facebook, MySpace, YouTube, and Twitter make in necessary to modify the syllabus of computer science degrees to incorporate these innovations [4]. Likewise, the popularization of computing solutions that collaborate with each other, which is what makes mashups possible, allows teachers to orient tasks towards increasing interaction with established web technologies such as standards related to the development of applications that are highly dependent to others in production or legacy (maximum cohesion) and that require applications to be independent from each other (minimum coupling). In the framework of web technologies used to provide integration and minimum coupling, web services and XML are the most widespread. Thus, our students develop mashup applications while incorporating useful techniques for managing integration between new and legacy applications that must be autonomous. At the same time, using XML as a communication protocol between services and the mashup, allows our students to acquire problem-solving mechanisms for the establishment of a "common language" between or among applications for their interaction.

Teachers also consider that the development of collaborative applications is entirely beneficial, if collaborative development is understood as applications that interrelate and benefit from the services they provide to each other. When students code applications that use data gathered from other applications, they see the resulting added value to the information. This has a positive impact in their motivation, one of the students participating in the *mAvatar4Moodle* project proposed and implemented additional features to share avatars through bluetooth and email.

All the preceding shows the pedagogical benefits of teaching mashups, mainly as regards motivation, as mashups extends learning beyond the limits of each kind of development, and in some cases, even the limits of computer science, as some applications have been taken to the social sciences.

4 Architectures for Mashups

There are many software architectures that can be used in the construction of mashup applications: server-based [5], client-based [6], and mobile [7].

Although we could divide a mashup application into several software layers for a detailed analysis, we generally find three main layers that allow us to understand the choice of architecture among those presented earlier.

Content provider or data source layer: data are available by means of an API and web protocols such as RSS, REST and Web Service.

Mashup layer: where the new service is constructed, nurturing from different sources to generate new information (which it does not own) and resulting in added value for the user.

Mashup presentation layer : mashup user interface.

Figure 1 shows the server-based architecture: the mashup layer is in the server and the presentation layer, in the client.

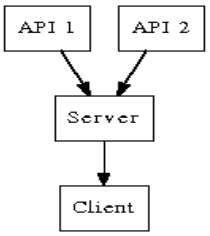

Figure 1. Server-based architecture

Figure 2 shows a client-based architecture: both the mashup and the presentation layers are in the client, which is an application in a browser.

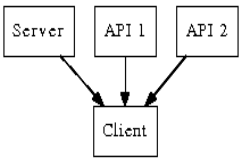

Figure 2. Client-based architecture

Figure 3 shows a mobile architecture: both the mashup and the presentation layers are in the client, which is a native mobile application.

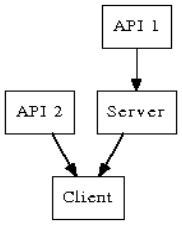

Figure 3. Mobile architecture

Native mobile applications offer maximum experience for mobile device users, as they allow for integration on a deeper level than a web browser. For a native mobile application, users have direct access to device features such as the GPS, accelerometer, camera and storage, and offers innovative interfaces based on gestures, directions, voice, etc. On the other hand, the development of native mobile applications available in a wide range of devices currently requires the development of as many applications as there are mobile operating systems.

5 Cases

In the Software Laboratory course, which is included in the fourth year of the B.S. in Computer Science and the B.S. in Systems at the Computer Science School (UNLP), students were guided by their teachers through development of native mashup applications for smartphones using mainly Java-based mobile device technologies, with Java Microedition (JME) and Android platforms. A mobile web widget was developed using HTML5 technologies and JavaScript on the Opera Widgets platform.

The projects presented are articulated with a School project related to mobile application development. For application development and testing, several mobile devices were acquired by the School. The devices used for this purpose were:

- HTC Touch HD™ T8282, with Windows Mobile® 6.1 Professional OS.
- Nokia N900 with Maemo 5 and Nitdroid (Nitroid is a free software project consisting in taking Android to Nokia Internet Tablet N900)) double booting.
- HTC Nexus One withAndroid 2.2 and 2.3.

The tested mashup applications were aimed at a mid-range/high-end device with touchscreen features.

JME-based projects were tested on the HTC Touch HD™ device with Windows Mobile 6. Because Windows Mobile does not natively support JME applications, it was necessary to install several emulators to provide a controlled environment for application execution. The HTC Touch HD™ device has a JME emulator, JBlend; however, it was impossible to emulate developed applications on JBlend. The teachers thus tested a series of emulators, which included IBM, PhoMe and JBed. All the applications were successfully emulated using these products.

Android-based projects were tested on the Nokia N900 device with double-booting and in the HTC Nexus One devices by Google, with Android 2.2 and 2.3. These devices allowed for native application installation, and there were great advantages in terms of performance and usability.

The experience in development and setup of JME applications raised some issues in terms of lack of available memory for the emulator to dynamically render maps that had to roll because of user interaction. To solve this problem, it was necessary to optimize resource allocation, which made it necessary to lose the portions of the map that were not rendered. This decision goes against efficiency, given that it makes it impossible to have a memory cache. Emulators restricted permissions as well.

Another topic relevant to mobile application development with JME are limitations in terms of user graphic interfaces. The user interface controls provided by the platform are basically text-oriented (text fields, buttons, etc) and not extensible or customizable.

With the goal of promoting collaborative work and giving the students a taste of a professional development environment, the teachers have created a wiki [http://wiki.labmovil.linti.unlp.edu.ar] that registers the experiences in the development of mobile applications. In this shared repository, the developed applications are presented by their authors, who share problems faced during development, noteworthy features, and the architecture and tools used. Screenshots and executable files are also shared, so that visitors can use and test the applications.

5.1 Mashup based on GoogleMaps

GoogleMaps is a free web service that provides interactive maps, where users can make annotations and mark locations.

Mashup applications based on maps developed in the framework of this project were aimed at the JME platform an allow displaying, consulting and adding information on a set of shared restaurant guides, which hold information on related places. These guides are shown by GoogleMaps in a map with markers in the location of the restaurants. Information can be stored about each location.

Many mobile device emulators were evaluated (JBed, PhoneMe and JBlend) to execute these applications, out of which only IBM gave the expected results.

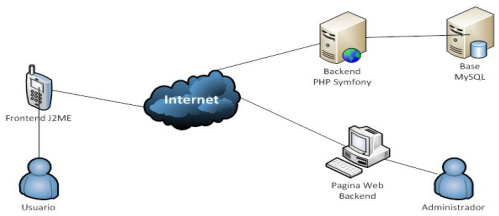

Figure 4. Architecture of a mashup based on GoogleMaps

The designed of the developed applications required a special user interface control to display the map in the screen of the device and be able to roll it in the four cardinal points (North, East, South and West), and read the information on a particular restaurant, among other features. As mentioned before, JME has very little capacity to build user interfaces, which made it necessary to develop our own user interface canvas with the required features.

The developed GoogleMaps-based mashup applications respond to a mobile architecture. As shown in Figure 4, a service was created in the web to register the information entered from the mobile application. The service implements an interface that is similar to REST to access and modify the information resources. Data are communicated through JSON.

5.2 Mashup based on Flickr

Flickr is an image storage website that allows users to organize their collection of images and share them. Using their API, content can be shared across websites, creating mashup applications.

The applications developed in this project allow managing Flickr user photostreams, creating and modifying galleries, uploading and deleting photos, modifying the information related to each photo and searching the entire site. In this project, some students chose to implement with JME and others with Android.

Flickr does not provide an API developed exclusively for mashups, so that data are obtained and processed manually.

The developed Flickr applications have a particular feature that can only be used through a mobile device, which is taking a picture with a cellphone camera and uploading it to a Flickr account from the application itself. In these development, it is worth noting the advantage of using native mobile applications with respect to desktop or browser-based applications.

Projects done on JME had some memory problems owing to the need to execute on an emulator and not natively in the device. The service that Flickr provides has authentication tokens for applications and users. Out of all the emulators evaluated, only PhoneMe allowed these tokens successfully.

Projects developed on Android had no functional problems due to native support. Likewise, very friendly and easy to program user interface were achieved.

The architecture of these applications is mobile, with a native mobile application as mashup and presentation layers, from which pictures can be taken and interaction achieved with modifications that are communicated directly to the Flickr service.

5.3 Mashups based on Facebook

Facebook is a currently booming social network, where people can share interests, tastes and images, and stay connected with friends all over the world. One way to meet a Facebook user is through their profile picture.

AvatarFacedGet is an application for creating and sharing avatars through Facebook, or storing them in a mobile device. It provides thousands of combinations out of a limited set of elements.

AvatarFacedGet is a widget developed for the Opera Widget platform [8] with standard web technologies such as HTML5, CSS3 and JavaScript, executed in the widget mobile manager, with multiplatform and multidevice features (from desktop to mobile to TV applications). This widget can be executed in a number of operating systems in mobile devices that include the widget mobile manager.

Facebook offers an API for developers that is very well documented, allowing to create applications inside Facebook or communicate with it through REST.

As is the case with Twitter and Flickr, Facebook uses application authentication keys. They can be obtained by filling a form in Facebook where the characteristics of the developed application are to be specified.

The architecture of AvatarFacedGet corresponds to a mobile mashup architecture, since the mobile client can only create avatars and the Facebook service is in charge of publishing albums and profile image.

5.4 Mashups based on Moodle

Moodle is an open source Course Management System (CMS), known also as a Learning Management System, (LMS) or as a Virtual Learning Environment (VLE). It is very popular amongst educators around the world as a tool for the creation of dynamic online websites aimed at students [9].

The Computer Science School uses a Moodle-based platform for its courses as a complement of in-class activities and for communication with students, located at http://catedras.info.unlp.edu.ar. The students of the School have a user in the platform that is associated with each course they are taking.

mAvatar4Moodle is an application aimed at mobile devices and used to build avatars, dress them up, give them expression and accessories, build avatar galleries and publish them as user profiles in the platform of Computer Science courses.

Students in the Software Laboratory subject have developed different versions of *mAvatar4Moodle* for the Android platform, and each of these experiences posed a challenge. One of these challenges was communicating with the Moodle service of the School, which implied managing SSL certificates through the application. For this purpose, it was necessary to add the certificate to the application for Moodle to trust its authenticity. It was also necessary to cover the need to use multipart forms to send images through the URL. Another challenge is related to avatar editor user interface programming, so that characters could be built on the basis of images representing their parts. Android offers a very comprehensive API for the development of user interfaces that facilitated the completion of this task.

The developed *mAvatar4Moodle* applications communicate directly with the Moodle system using the device's network layer. For this, a SSL certificate was added and a direct connection with the system was implemented. Still, the mobile architecture remains dominant, since avatars are published on user profiles through the Moodle server.

6 Conclusions

The use and development of innovative technologies cosnstitute appropriate elements to be incorporated in computer science degrees. However, the main goal in university-level computing education is to acquire concepts with lasting relevance. Modern technologies should be incorporated to illustrate these fundamental concepts and place the students in a working environment that closely resembles the real one.

Mashup development for mobile devices is a motivating, innovative topic that students are very interested in. It is a challenge when developing applications for platforms with such characteristics as display and hardware limitations and new features such as movement sensors, temperature sensors, GPS and camera, among others. This poses new ways of thinking of applications.

From the experience, it is worth noting that students who chose JME as their development platform faced greater difficulties, from adapting resource allocation to limitations in the construction of friendly and custom user interfaces. Students who opted for Android as their development platform encountered simpler solutions with a consistent interaction throughout the device and features that allowed them to easily extend the functionality of the device incorporating sensors.

7 References

[1] Agnes Koschmider, et. all, "Elucidating the Mashup Hype: Definition, Challenges, Methodical Guide and Tools for Mashups", Proceeding of 2nd Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM 2009), April 2009.

[2] M. Xenos, D. Stavrinoudis and D. Christodoulakis. "The correlation between developer-oriented and user-oriented software quality measurements (a case study)". Proceedings of the 5th European Conference on Software Quality, Sponsored by European Organization for Quality − Software Committee (EOQ-SC), Dublin Ireland, pp. 267-275, 1996 .

[3] E. Michael Maximilien, "Mobile Mashups: Thoughts, Directions, and Challenges", 2008 IEEE International Conference on Semantic Computing, pp.597-600.

[4] A. Easton and G. Easton. "Demystifying Mashups". Proceedings of Informing Science & IT Education Conference (InSITE) 2010, pp. 479-486.

[5] E. Ort, S. Brydon, and M. Basler. (2007, May) Mashup styles, part 1: Server-side mashups. [Online]. Available: http://java.sun.com/developer/technicalArticles/J2EE/mashup1/index..html

[6] E. Ort, S. Brydon, and M. Basler.(2007, August) Mashup styles, part 2: Client-side mashups. [Online]. Available: http://java.sun.com/developer/technicalArticles/J2EE/mashup_2/index.html

[7] A. Brodt and D. Nicklas, “The telar mobile mashup platform for nokia internet tablet”. Proceedings of the 11th international conference on Extending Database Technology (EDBT 08), ACM, 2008, pp. 700–704.

[8] Opera Widget SDK: http://dev.opera.com/sdk/