Hindawi Publishing Corporation
EURASIP Journal on Wireless Communications and Networking
Volume 2010, Article ID 689517, 11 pages
doi:10.1155/2010/689517

Research Article

New Trends on Ubiquitous Mobile Multimedia Applications

Joel J. P. C. Rodrigues,[1,2] Marco Oliveira,[2] and Binod Vaidya[1]

[1] *Instituto de Telecomunicações, UBI, Rua Marquês D'Avila e Bolama, 6201-001 Covilhã, Portugal*
[2] *Department of Informatics, University of Beira Interior, Rua Marquês D'Avila e Bolama, 6201-001 Covilhã, Portugal*

Correspondence should be addressed to Joel J. P. C. Rodrigues, joeljr@ieee.org

Received 2 March 2010; Accepted 1 July 2010

Academic Editor: Liang Zhou

Mobile devices present the opportunity to enhance our fast-growing and globally connected society, improving user-experience through novel approaches for information dissemination through mobile communication. The research community is developing new technologies, services, and applications to enable ubiquitous environments based on mobile technology. This paper tackles several important challenges such as communication cost and device limitations for development of ubiquitous multimedia applications. And we propose a system for news delivery using a set of wireless multimedia applications. For this purpose, we have performed a case study with Apple iPhone's platform, featuring two multimedia application contexts, namely, Web and native applications. The multimedia mobile applications draw on iPhone's assets, enabling context-awareness to distribute news, improving communication efficiency and setting-up viewing optimizations, thus enhancing user-experience. The proposed system is evaluated and validated through a series of real-life experiments on real devices, with online full availability. Moreover, due to the Web application availability, the system is not restrained to Apple's iPhone platform, but can also benefit users with other devices.

1. Introduction

Mobile devices have fulfilled the true aim of Internet by offering full connectivity anytime anywhere. The trend of going wireless goes beyond the walls of homes, university buildings, or hotels and reaches the open spaces of nature or the mobile spaces of trains and buses. The freedom of movements is used to speak everywhere without the need to log in a local wireless network, and to extend it to other Internet services such as Web surfing, email checking, reading news, listen to online radios, or even watching video streaming and television.

The mobile devices market is an emerging mass market with little data usage research available. Consumers are changing their habits, the Internet players are adapting their contents to adjust the new needs, and operators are maintaining a high cost and network restrictions to avoid massive usage. In this market, the business model is restricted to cost-per-use, so data porting to mobile devices have not really taken off.

With embedded technology, mobile devices have many features, so people tend to use mobile devices in order to access to Internet contents more frequently. In this paper, we want to explore and take advantage of this technology, study people habits, and understand their perspectives to this recent trends for the future to create new ways of content distribution. This paper also proposes a mobile system that tries to gather the above-mentioned characteristics for iPhone platform from Apple. We refer to iPhone assuming that mobile device incorporates iPhone, iPhone 3G, iPhone 3GS, and iPod Touch functionalities.

This work studies ubiquitous Internet services in two types of applications—Web applications and native applications. The delivery system incorporates context-awareness, industry standards, and users predefined settings to evaluate how the information is processed, transmitted and displayed. The proposed system improves mobile devices, communication, server side computation processes, user-experience, and usability. The benefits for clients are focused on what and how they want to see, speed, and Internet lower data transfers costs. The benefits for mobile devices will affect battery, memory, processor, and Internet connectivity performance [1]. Furthermore, service providers should improve network availability, bandwidth, and reduce storage costs.

The native application resolves some of the wireless networks limitations such as low-bandwidth and unreliability, understands the user definitions, and reads some mobile devices context in order to adjust the dynamic content downloaded from the Web Service. This solution combines ubiquitous information in both sides of the provided system. On the server side, it calculates image quality, number of items to download, and text format. On the mobile device (client) side, it decides what, how and how many news information or images will be downloaded. This ubiquitous collaboration between mobile device and Web Service is a new approach of intelligent applications that brings out the betterment of two worlds—the server-side power and the client-side context, location, and sensor awareness [2]. Moreover, for journalists and all the news related professionals, it is important to know if a mobile device is just a Web page extension or a new way of communication, more specifically—the seventh way of communication [3].

The rest of the paper is organized as follows. Section 2 reviews the related literature and the background for mobile and ubiquitous applications. Section 3 focuses on technology, specifications, tools, and methods used to create the proposed applications. Section 4 describes, in detail, the four proposed Web applications for news delivery and presents a comparison study between "Mobile", "News", "RSS", and "Edition" applications. Section 5 elaborates on the native application and addresses ubiquitous technology incorporated on the system. Section 6 concludes the paper and points out directions for further research works.

2. Related Work

Mobile devices, which maybe also referred to as handheld, portable devices, or wearable devices, such as mobile phones and personal digital assistants (PDAs), are small and lightweight equipments that can be fit into a suit pocket, hand, or briefcase. For this work, we have considered mobile devices such as mobile phones and PDAs with Internet connection capabilities and a small-size screen. These gadgets can provide not only regular phone calls but also other features like electronic mail (email), gaming, infrared, Bluetooth and Wi-Fi connectivity, photo camera, video recording, music player, radio, and global positioning system (GPS).

Even though mobile devices have a screen limitation in terms of size, all the content available to other kind of displays (cinema, television, and personal computers) are being produced accordantly to mobile characteristics. The content from other types of media such as recordings, print, radio, and Internet access, can also be imported to these devices.

In 2009, the estimated number of mobile phones subscriptions around the world was about 4,6 billion for a population of 6.8 billion people. And the penetration of mobile phone in the industrialized world was about 133%. In the same period, the broadband Internet access penetration (in mobile devices) was 20 times bigger than in 2005. This makes the mobile phone the most widely spread technology and the most common electronic device in the world. We can predict that it will be the device of the future. According to [4], the enterprise mobile phones will replace desktop phones in North America by 2011. As the number of people that accessed the Internet via their portable devices have increased by 25% in the second and third quarters of 2008, among which the most audience is young generation, it can be predicted that the contents for these gadgets will continue to grow in future as well. Furthermore, in 2007, for the generation of USA Internet users between 18 and 24 years, the preferred consumer electronics was the mobile phone (47%) over the computer laptop or desktops (38%). Indeed, as a 2008 Nielsen Media Research report highlighted, mobile devices have increased traffic by an average of 13% across several popular websites [5].

The emergence of Web 2.0 has transformed the web into a more dynamic and interactive environment, offering a set of tools that enhance contact and collaboration between users. Several applications including online social networks, wikis and blogs, support such Web vision [6]. Currently, the interconnectivity and interactivity of Web-delivered content, which were born for the desktop computers, have been extensively applied to mobile devices as well. This new vision of Web is gathering the best that these devices have to offer—portability and connection everywhere—, and the Internet providers are creating new and innovative services for this market [7].

In general, the integration techniques used to combine Web Services and mobile devices are Socket communication and messaging techniques. Web Services uses extensible markup language (XML) and simple object access protocol (SOAP) to provide a mechanism that facilitates the data exchange over the Internet. They are being widely developed to enable quick and cheap integration with existing services, by combining multiple services in a single workflow. This facilitates interoperability across different hardware and software implementations, as it will be discussed on next sections.

Nowadays, the application programming interfaces (APIs) created by several companies, such as Google, Amazon, and eBay, have robust Web server integration with desktops but when they migrate to mobile environment, new challenges should be addressed. The following aspects are identified: client-server data transfer optimization performance of the mobile application memory management security issues and user interface design with such display limitations. To overcome these issues, new techniques for caching, large data set handling, information on demand, data compression, paging, filtering, and performance improvement of network protocols are proposed for mobile computing.

The number of news and media content downloaded through Internet and portable devices are increasing [8], therefore, we can predict that the number of these kinds of media Web Services will also increase. We also have the perception that typical iPhone owners are bigger infotainment consumers. They are visiting, at least three times more than average, to several popular social, communication, and entertainment sites. The major newspapers and media groups of the world have already used the iPhone application for delivering news in this new format. So, at this point,

several questions may be raised. What mobile Web Services can we choose? What about the design guidelines for mobile devices? Or, there are used mobile development best practices? These issues were considered and kept in mind when planning our approach for the mobile Web Services.

Furthermore, ubiquitous computing is an omnipresent relationship in terms of connections, management, and information interaction. This technology must create calm, and act as a quiet, invisible servant. It should help humans to extend their *unconsciousness* and intuition. This calm technology has the ability of going from the periphery of our attention, to the center, and back again [8]. Computing, communications with other devices, relationship management, are empowering our periphery but not moving much to the center of our attention. The impact on everyday life is already huge and becoming so commonplace likes writing or electricity.

Mobile devices are getting smaller and powered up with add-ons, speed, and battery life-time. Combining those features and the growth of short-range ad hoc networks, the possibility of accomplishing the vision of ubiquitous computing, that was sketched out in the early nineties, is getting closer [9, 10]. With these significant improvements in the network and devices, the software is trying to catch up. Between them, the ubiquitous systems are still in their early phase.

The impact of this technological wave will alter the place of technology in our lives. By now, every mobile device has Internet connection capabilities. We can interconnect them with other devices and systems, creating a computational relationship between them in a calm and ubiquity context.

With increadingly use of mobile devices with GPS capabilities, location-awareness devices such as the iPhone have changed our daily life. We will be capable of pinpointing our location on a Google Map, tracking friends, finding the nearest place to eat and shop, finding information of the particular area and so on. The first application for the iPhone that uses the faux-GPS feature (which used cell tower information to triangulate your position) is Google Maps. Now, there are a large number of applications that use real GPS features and services. Some examples of location-aware applications that can be found in Apple App Store are the following: Loopt—Friend-finder application with virtual earth display, which allows user to share his location to the community; Whrrl—From Pelago's, is a friend-finder, business applications with browsing functionality; Urban Spoon is a restaurant picker based on your location; NearPics is a location-aware photo browser and uses Google's Panoramio service; Weatherbug is a location aware weather service with predefined cities; StreetFlow; Yelp; Twinkle (Twitter); BrightKite. The list is growing every day.

Mobile devices are equipped with wireless capabilities and users can go through several contextual changes as they move around. These changes are related to the movement of the user in his physical and social surroundings. By sensing their environment, mobile devices are capable of communicating and delivering ubiquitous services adequate to the situation. The dynamic nature of the system implies that as device context changes, delivered information can also change, due to an interoperation with the content server. The server will find the information, adapt it to the user context and format it for delivering. The history of the user is also taken in consideration, hence intelligent handling of the data.

Mobile Internet is about functionality opposed to entertainment and e-commerce on the screen-based systems. We also believe that mobile experience merits its own design, customized to their needs, having the best practices, efficiency, and accessibility. We know that a small screen size doesn't match a 22" liquid crystal display (LCD). People use the portable devices when the information or functionality they need cannot wait, so they go to a computer screen. Therefore, developing for these screens and devices also brings more new issues, paradigms, and semantics to the world of mobile devices applications.

3. Developing for Mobile Devices

Web development involves the creation of optimized Web pages for mobile devices. Standard Web programming languages, such as, hypertext markup language (HTML), cascading style sheets (CSS), JavaScript, and hypertext preprocessor (PHP) may be combined with available tools provided by companies or Web developers. These Web pages run on the mobile device browsers. iPhone uses a mobile version of Apple's Safari. Main advantages of these developing tools are the following: ease and fast development; ease of user-access; ease updating; access to dynamic data; and offline server access.

Native applications have more functionalities than Web applications. Therefore, a native application is the best choice for iPhone users. iPhone native applications have four distinct layers. The first layer includes the source code, the compiled code, and the software development kit (SDK) frameworks. The second has the *Nib* files, which contains the user interface (UI) elements and other objects (the design), and details about how objects relate each other. The third contains resource files (images, sounds, string, video), and finally, the fourth includes the "Info.plist". This file saves details about application configuration. The proposed system assumes that all applications run natively on iPhone. The programs were created using iPhone SDK and Objective-C language. These tools offer several advantages in comparison with others. They include a more complete development environment, improved language depth, integration with SDK frameworks, iPhone emulator, and software debugging.

In terms of development aspects, Native application completely differs from a Web application optimized for the iPhone excluding some similar tools in the SDK. For instance, Safari Web browser limits web applications while native applications are limited by the iPhone operating system. In terms of price and business model, the differences tend to get bigger. Native applications are sold and distributed through the App Store. They can be downloaded directly to iPhone or using iTunes desktop version. Native applications follow the Apple software license agreement, keeping 30% of the price (if payment is required). This fee is paid for keeping the "store clean" and for support

Table 1: Main differences between iPhone Web and native applications.

	Web applications	Native applications
Technology	HTML, CSS, JavaScript	Cocoa, Objective-C
Deployment	Web server	App Store
Frameworks	Safari, limited iPhone OS	iPhone OS
Limitations	Memory, JavaScript Runtime	Cache, DB
Installation	Just online access	Download or sync
Findability	URL/Web	iTunes/Web
Access	Through Safari framework	App Store download
Offline usage	No	Yes

transactions, server storage, helpdesk, and quality control. Main differences are summarized in Table 1.

Communications speed and the fast information process need to be considered since the wireless communication tends to drain out the battery. Furthermore, assuming that access networking is limited in several locations, Web applications cease functioning.

Web applications do not have a repository store like the iTunes App Store. All the existing applications are on the Internet without any common reference. The advantages of the App Store could be reduced if Apple creates a website to store and control Web applications instead of being dispersed on the Web. Web applications have another drawback relying on Safari browser. Safari, like other piece of software, owns flaws, bugs, memory leaks, and, in a future upgrade, Safari App could jeopardize the Web application. Furthermore, Xcode tool offers an easy environment to create native applications in comparison with tools and resources available for Web developers [11]. The major advantage for Web applications is the programming language because it is not limited to Objective-C or object oriented programming. Moreover, contents are always updated and synchronized with Web server. Regarding native applications, they can only be downloaded through iTunes while Web applications uses the user-friendly uniform resource locator (URL) entry in Safari. In terms of applications acquisition, buying natives is easier when compared with Web because the first uses iTunes and the later needs to use a credit card each time an item, service or paid access is purchased. iPhone SDK can be used to create both kinds of applications. Table 2 summarizes a comparison between developing "Web Apps" and "Native Apps", in the developer perspective.

Table 2: Comparison of main technical characteristics between Web and native applications in the developer perspective.

Features/access	Web applications	Native applications
Installation	"Add to main screen option"	Through App Store
Initializing	Open Safari bookmark or insert URL	Click installed icon
App Frameworks	JavaScripted	Custom
iPhone Frameworks	Limited	Full SDK
Sandbox	Safari sandbox	App sandbox
Cache	Safari cache	Sandbox files
File system	None	Sandboxed
OS Memory	Page shared	iPhone OS (128/256 MB)
Customizing	User Web login	App and iPhone settings
Sensors	Limited	Yes
Location	Yes	Yes
Accelerometers	Limited	Yes
Cocoa Touch	Limited	Yes
Network	Auto	Custom
Multimedia	Embedded	Custom
Database	No	SQLLite
Offline	No	Yes

4. Web Applications Toolkits

This section describes the four proposed Web applications such as "Mobile", "News", "RSS", and "Edition" and the corresponding Web server created to deliver news from the *Urbi et Orbi,* which is online newspaper at University of Beira Interior, Portugal. These Web applications use generic libraries for structural support of the mobile devices browsers. The "Mobile" version uses the *iWebKit* free toolkit created for anyone wanting to create iPhone websites. Versions "News" and "RSS" are based on Apple's *UIKit* that is the equivalent of *AppKit* for traditional OS X applications. The "Edition" version uses a JavaScript framework called *WebApp.Net,* which allows working with asynchronous JavaScript and XML (AJAX).

Due to screen sizes and browser limitations, the proposed ubiquitous mobile multimedia applications also addresses design concerns. The user interface is designed to avoid horizontal scroll bars, and the most important news is located on top of the screen. The content design follows a top-down approach according to the importance and a left-right disposition, per levels, according to the intended detail of the news content. Furthermore, another important concern kept in mind regarding the design of user interface is Web accessibility on mobile devices, improving usability and user experience. Then, the Web content accessibility guidelines (WCAG) 2.0 is followed [12].

This study analyses the differences between the four proposed versions of Web applications related to their frameworks in order to determine betterment for the ultimate client. The "Mobile" Web application was initially used to create software specifically for the iPhone because it has optimized code to work with this device. This version was also tested in different browsers outside iPhone and the results have shown a complete website with no information lost.

Two additional Web pages were created for displaying news through images ("*Urbi* in thumbnails" and "*Urbi* on images"), in order to navigate through the "touch control with slide effect" from iPhone. The Web application

structure is based on a top-to-down item table to display the most important information on the page beginning. This structure is also organized in left-to-right navigation to access detailed information on the right, as recommended by the UI rules that Apple uses for Web applications.

Figure 1 depicts the website organization. Level (1) area is the front-page of the newspaper edition. This part is structured in *Latest*, *Categories*, and *Others* sections ("*Urbi* on Images" and "*Urbi* on Miniatures"). Level (3) presents the page with news details and it can be accessed from levels (1), (2), (2'), and (2"). Specifically, the detail page on level (3) contains all information regarding the new item information—title, *super-lead*, picture, corpus, journalists, and published date. From here, users can follow more pictures, or other detailed pages, other websites, and multimedia contents. Level (2) illustrates an example of a item list of that category and it is ordered by the latest item publishing date. Each item is also linked to a detail page previously described as level (3). Part (2') is the "*Urbi* on images" Web page containing the images and captions that forms the list of news, and users can navigate through them horizontally using the *Flick* control action. Part (2") is the list of thumbnails containing all the images from the news edition. Each thumbnail also is linked to the referring detail page known as level (3).

The "News" Web application presents a table with the list of news contained in the latest edition of newspaper. It was built on Apple Dashcode tool and uses the *UIKit*, a framework specifically created for iPhone Safari. The major difference to previous framework is the inclusion of JavaScript functions to enable content delivery through *XMLHttpRequest* methods. This asynchronous request to the Web service is made while a browser renders the page in order to save time on page loading. The retrieved file of this request includes XML elements that will be used to populate the main HTML tags of the Web page.

The "RSS" is another version of the Web application and it uses the really simple syndication (RSS) Web service provided by the same newspaper. This Web application gets a RSS feed (asynchronously) containing XML elements that will be parsed with HTML elements in the main page. The buttons on the detailed information page are linked to the Desktop version allowing users to continue his reading. Authors modified the template in order to include JavaScript functions to retrieve the time required to conclude the page display and the number of items included on the feed file. The 'RSS' version does not requests any kind of multimedia files as opposed with other Web applications in order to compare how much JavaScript computation is needed to complete the information display.

When these Web applications (both "News" and "RSS") are activated through the dashboard icon, the JavaScript included in the application hides both top and bottom navigation bars of Safari, making these Web applications almost similar to a native one. Then, the table list is displayed with maximum pixel height, providing to users the same experience of a similar native application.

The last version, called "Edition", uses *WebApp.Net* framework, which is an open source Web application framework, created by Chris Apers and it was designed to mimic the current iPhone graphic UI. All content of this application is dynamically loaded through AJAX requests. On the contrary of versions "News" and "RSS", "Edition" performs an asynchronous request only when the user needs it. Each request of detailed item information creates a different AJAX request. Image files with lower quality are created for this application in order to reduce costs-per-download. The CSS file retrieved with the first page of this version is also different regarding the access timestamp. This version also includes search possibility by providing a form to input queries.

The versions "News", "RSS", and "Edition" use *XMLHttpRequest* object to connect directly to XML data for feed updates without reloading the page. Normally two JavaScript functions are used to provide AJAX requests: *loadXMLDoc*, *processReqChange*. These generic functions include object creation, event handler assignment, and submission of a GET request. After creating the object through an ActiveX constructor, several other methods (*abort*, *getAllResponseHeaders*, *getResponseHeader*, *open*, *send*, *setRequestHeader*) and properties (*onreadystatechange*, *readyState*, *responseText*, *responseXML*, *status,* and *statusText*) can be used to manage the connections. The XML data is then converted (parsed) into standard HTML content.

A system prototype (testbed) was created to test and validate the proposal and to evaluate the performance in terms of speed and size. The procedure consisted of loading the Web application from different systems and platforms. To measure the size, requests, and loading speed, the following three different clients were used: (i) iPhone 3G connected through Wi-Fi; (ii) iPhone simulator; (iii) Safari browser running both on a Microsoft Windows XP machine and on Mac OS Leopard. Those latest clients were connected through cable network at 100 Mbps.

Table 3 presents a comparison among the four versions of the proposed Web applications, evaluating the following downloads: the homepage of a given website, the corresponding first option webpage (called detail page), and the whole website. As may be seen, "Edition" version obtains best values for downloading the homepage and the website as a whole. The "Mobile" version is the smallest version when it comes to the homepage, but the time consumed to satisfy all the requests is bigger because it spends more time to download images. This version performs better on the parser time to display the homepage. Regarding the download of the detail page and the website as a whole, "RSS" performs better because does not download images. It does not show the best performance on the homepage because JavaScript files from the *UIKit* have greater size than the other version. "News" version shows the worst performance in the homepage download scenario because request all the images of the newspaper edition. It can be concluded that taking into account the main characteristics of each version and the above-mentioned considerations, "Mobile" performs better than other versions. Therefore, this version of the Web application is selected as a default version of the system when a Web application is requested.

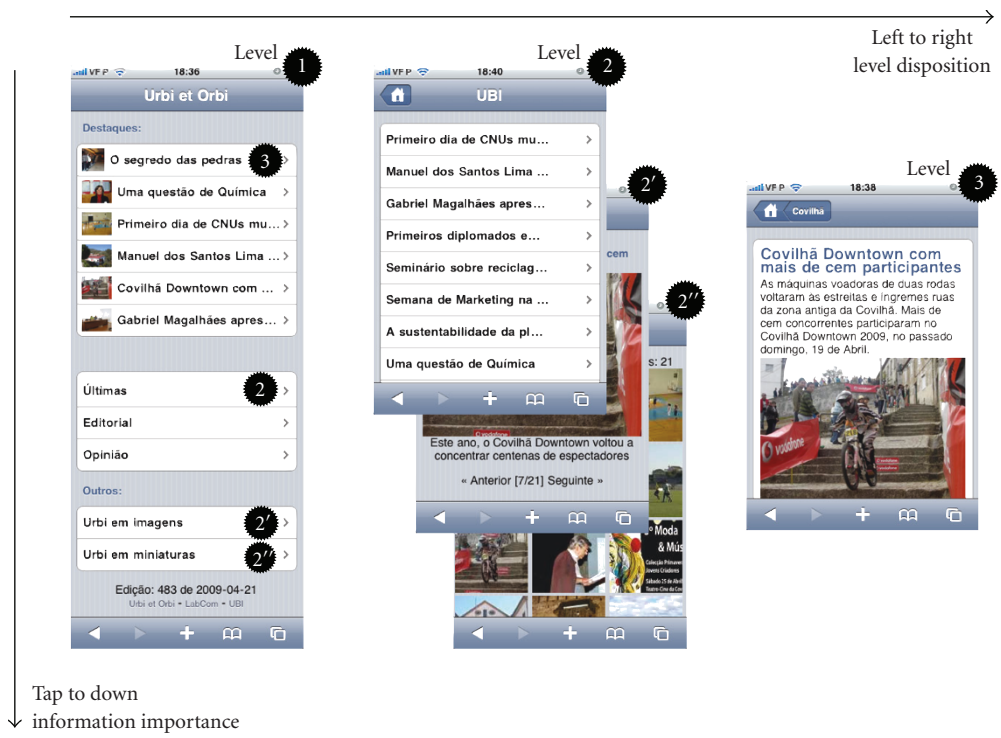

Figure 1: "Mobile" Web application structure and screenshots.

Table 3: Performance evolution of the of Web applications taking in account size and speed of the transferred content between server and client.

	Mobile	News	RSS	Edition
First page access				
Page size	59 KB	368 KB	150 KB	90 KB
Items requested	20	51	18	24
Communications (seconds)	1.8 to 2.4	0.7 to 4.2	0.4 to 0.6	0.7 to 1.9
Runtime parsing (seconds)	0.02	0.6 to 4.1	0.35	0.05
Detail page				
Page size	81 KB	20 KB	5 KB	23.5 KB
Communications (seconds)	0.9	0.2	0.1	0.2
Whole website				
Levels	7	3	3	3
Number of files	289	24	17	86
Full site size	2015 KB	183 KB	148 KB	149 KB

After describing and evaluating the Web applications, we focus on the Web server. The proposed model for server is based on standard Web-based client/server architecture. Web standards and simplified models were used to design the system architecture in order to improve portability and scalability. Designing with Web standards offers a major benefit because once designed, it can be published everywhere [13]. The Web server is based on the Linux operating system, Apache HTTP server, PostgreSQL database management system, and PHP programming language, constituting the LAPP architecture.

This server was designed in the perspective of ubiquitous computing integration on the solution. In this sense, a ubiquitous Web service was created to process and answer the client requests, illustrated in Figure 2. It can be seen that phase (1) of the process focuses on collecting and filtering information from Apache server and news database, phase (2) applies the templates to the data gathered on phase (1), and phase (3) adds the specific design files (CSS's and JavaScript) according to the ubiquitous results.

Client requests from "Edition", "RSS", and "News" versions are received and handled by three Web services, one per Web application. For the 'Mobile' version, ubiquitous computing is performed on each page request taking into account that version requests page by page. Each Web service uses specific classes and rules to handle client requests in a ubiquitous perspective. When a Web server receives a client request, ubiquitous decisions influence the images treatment in order to create and deliver images and thumbnails. A rule to change the image size and quality was created to reduce the file size and drop cost per download. This rule applies different compression algorithms according to the Web application version request. Upon reception of a request, another rule is invocated to deliver a specific CSS file, according to the client's time stamp, trying to improve contrast on the mobile device. Domain name system (DNS) reverse lookup to give the location of the client is also used. The time stamp is calculated with client location and server local time. The application can choose specific news for

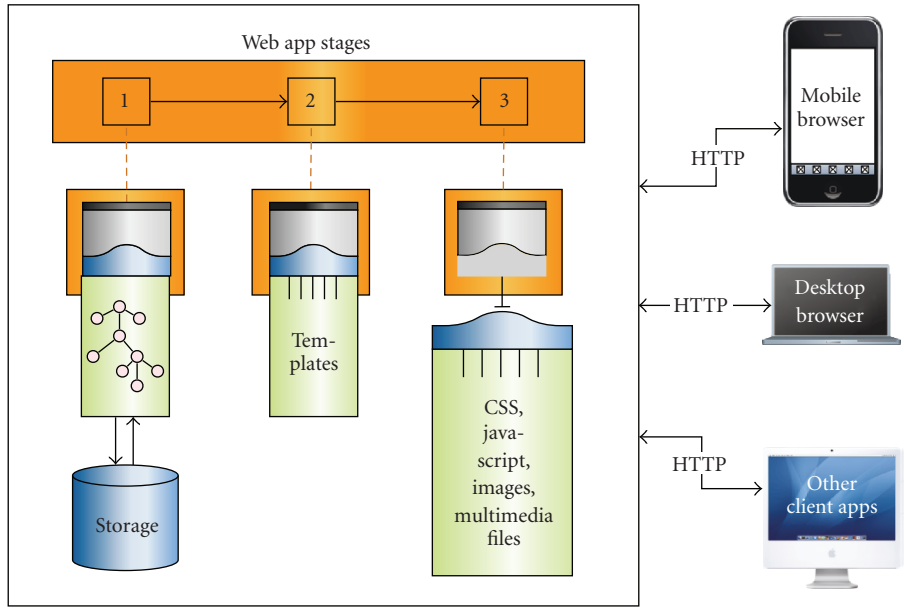

Figure 2: Ubiquitous Web service model diagram.

each region, offering the content in a different text language. Furthermore, if the IP address belongs to the user intranet, maximum quality of images is applied.

A transmission of a big file is more efficient than a transmission of several small files [14]. This approach is considered on "News" and "RSS" versions. The other versions use a single request for each webpage. Therefore, "News" and "RSS" have better performance in comparison with the others, as may be seen in Table 3, taking into account the number of files downloaded.

The server also stores other multimedia files not processed by ubiquitous rules, such as portable document format files (PDFs), audio and video files.

5. iPhone Native Application

The news for mobile devices (N4MD) is a simple application for viewing the weekly news produced in the *Urbi et Orbi* newspaper. This application runs natively on iPhone devices and the primary difference from the previous Web applications is that the content is available for offline reading. This section describes the native application called "*N4MD.ap*". It has been tested and validated. And it has the foundation for a worldwide quality application to be distributed through Apple App Store.

The design of the native application follows the same approach as described for Web applications in Section 4. User interface has a top-down list of news and a left-to-right navigational interface. The default primary view is the list-table-view—"*Urbi et Orbi* News". This window has four distinct areas, as shown in Figure 3. From the top to bottom, the user interface contains the following elements: (i) the status bar—iPhone's grey bar at the top with (from left to right): the cell signal and the carrier name, the network connection type (Wi-Fi, EDGE, or 3G icon), the clock information, and the battery status; (ii) top Bar Navigation—Table's title "*Urbi et Orbi* News" and *reload* button on the right; (iii) the "Table View"—List of news with thumbnails (default option) for each item, and at the bottom in orange, the table contents information (date and number of items); and (iv) the Navigation Bar—The bottom bar in black with two options: "*Urbi et Orbi* News" and "Settings".

The "Detail View" appears when a row item on the first window is clicked. This is a subview of the primary view. Therefore, the Top Bar and relative content are related, which has the following three objects: the *back* button with the title of the table information "*Urbi et Orbi* News"; the new information index and total of news; and the segmented buttons with navigation through items details. The "Detail View" appears in the white area below the orange "Top Bar". This white area is a vertical scrollable object and with similar behavior to the above described in Web applications detail page. A user can find (from top to bottom) the following news elements: title, description, image, image caption, published date, category, author info, and full new *corpus* text. On the settings tab, the user can choose (through the *slider* button on the top) if he wants to see images or not. In the same view, user can find some "control information" that may be collected and sent to the server. These data (the device unique identification, the name, the system version, and the battery information) will be used to perform ubiquity. The iPhone OS Library at iPhone Dev Center [15] was used to create the application.

The usual touch screen of iPhone controls the navigation actions. The finger movements supported are "Flick" for scrolling and "Tap" for action or selecting. The N4MD user interface uses the Apple's suggested left-to-right navigation approach to go from top level to detail levels. It also uses

FIGURE 3: Web versus native applications differences and characteristics.

a "Table View" and a "Scrollable View" to display more information in a top-down structure without zooming or panning. The N4MD application does not support "Rotation View" because we primarily tried to mimic some native Apple iPhone applications like *Phone.app*, *Clock.app*, or *iTunes.app*, which do not support "Rotation View" as well.

Normally, we wanted to use the iPhone's features and frameworks integrated in the device operating system, and available to the developer, in order to ubiquitously connect, download, manage, and display the news content. So, the content will have to be downloaded and saved in iPhone file system, allowing the application to access them offline. However, the application provides some form of user control to override the application and server ubiquitous decisions [16]. Ubiquitous computing is an important matter for mobile computing. Therefore, two types of ubiquitous decisions were created, one for the server and other for the native application.

The server considers two types of applications, the native and the above-described Web applications. Windows Mobile, iPhone or Google Android devices post different requests. Therefore, answers must be adequate and specific for each kinds of system. Users have different needs and there are other types of applications. Thus, the parameters sent to server, in order to make those ubiquitous decisions are the following: user agent, client device type, network type, screen width, battery, categories, sections, items, and edition date (as shown in Figure 4). Communications with the server are also important for the application speed and for the device battery life. So, one of the requirements for the applications should be related with network type (Wi-Fi, 3G, or EDGE) and images. To ubiquitously avoid image downloading, the user can choose an option to override images or not. The system also have more rules to define the default option for image downloading, to seamlessly create image thumbnail and asynchronous downloading. Other rules for avoiding images downloading, such as the following: if the application uses the EDGE network connection or if images are already cached. When there is no image to show, the application adapts the "Detail View" and do not present "Image Caption" and image placeholders.

The N4MD application started with the creation of a default Xcode template—The iPhone OS "Window-base Application" [11]. By doing that all the files and documents were automatic created, and some of them had a few lines of codes allowing us to "Build and Go", even without changing a line. Then we had to modify some of the existent bundle files (*png* images, visual elements on the *Xib*, and the application default *.h* and *.m* files).

In the "init" stage, while launching, the application gets user preferences and starts to build the window with components like the "Navigation Bar". Then, the application checks if it has a network connection. If it has network connection, it sends a request for server statistics. The next stage is getting the articles (the news). If there is a network connection, the application makes a XML http request, if not, it tries to load a previous saved "plist" file. In neither case, the application continues by parsing the information. Then, the display stage happens when the information is loaded and we can see the "Table View" with article title, description and thumbnail. After the first display stage, the application enters in a cycle (waiting for a user's input that change status in order to make new display changes) or quits. The "user-default" feature is used to save three state variables for user and application settings in the applications bundle directory. This information is useful as it seamless shows the last viewed window before exiting. Therefore, the user does not have to navigate through the application all over again. The data from the XML http request in a form of a "plist" file—which is also a XML format—is saved within the application sandbox. The downloaded images are saved on Documents directory. When user click on the *reload* button and the flag for Internet connectivity is *on*, the applications erase all the downloaded data from the bundle

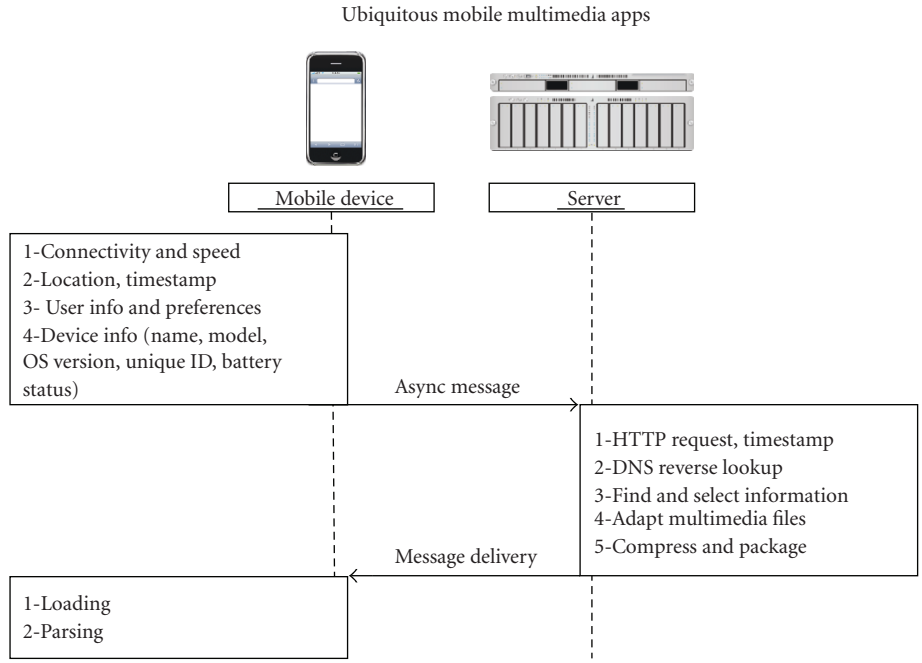

Figure 4: Sequence diagram model with context-awareness information and process variables to ubiquitously perform output results.

directory and starts the asynchronous communication with the server to get refreshed information. Moreover, screen size forces a top-down design for information display, the "Table View" is one of the most used components for the iPhone regarding its vertical characteristics. Mobile devices do not have characteristics that a desktop computer has, such as the following: all the mouse movements or click events; not all views have page zooming; text size adjustments, options for fonts sizes types, form controls or form saved information. Therefore, those characteristics cannot be used in the ubiquitous system. The iPhone OS recognizes some special links like: *http:*, *https:*, *itunes:*, *phone:*, *mailto:*, making communication between applications more dynamic.

For improvement of speed, the application depends on the network speed and number of items to be downloaded. It only occurs when threaded *url* connections are used to speed up the application, and it is provided by the *asyncimageview* class. The application contacts the server in two distinct stages. The first contains information about the device (name, model, localized model, system name, system version, and unique identifier). The second communication occurs when the application makes a request for data (news content feed), and at this point the information to server contains battery status and client type. These stages are important for the success of the ubiquitous system. Other requests are motivated by references of thumbnails and images included in the XML file. This request just happens if the user allows image loading on settings, and if the used network is 3G or Wi-Fi.

In order to create ubiquity in N4MD application, the communication between server and client was forced. The client has to collect context-aware information and send it back to the server [17]. Then, the server must determine the location-awareness of the device (so far we do not use GPS), save the user statistics, and use the information received in order to deliver the desired information content in a specific format.

6. Conclusions and Future Work

Applications for mobile devices are gaining their own market as the seventh way of communications by being less equal to small online versions of the bigger brothers, and taking advantage of the opportunities as they pop-up. Web applications and native applications must co-exist due to connectivity issues and offline reading. In terms of design, they are similar in many ways and also share architectural and structural models. But their final purposes make them unique to each other and have usefulness on their own way. Moreover, Web applications are oriented for cloud computing, peer collaboration and synchronous connectivity. On the other hand, Native applications take advantages of mobile devices characteristics and access to more frameworks.

Applications for mobile devices must use ubiquitous computing techniques in more effective way. By addressing fundamental topics, in a quest to unleash the full potential of data consumption, the usage of location and context-awareness in mobile devices are changing our life quality for the betterment. Those above described systems combine the portability of a Web Service with the mobility of users to overcome the limitations of mobile devices.

This paper proposed and described in detail a system for delivering news using wireless multimedia applications and transmission techniques to mobile devices, using proposed platform to Apple's iPhone. For study purposes, four

different versions of a University online newspaper were created. These versions were produced to specifically provide an important parameter of their target mobile device. While testing size, runtime speed, connections, design, and usability, our study shows that each version has its own advantages as we have expected. Ultimately, the server decides the best version for the specific client and delivers the corresponding application. But, the standard default (to be used with all around mobile devices) is the "Mobile" version. This version was developed with the *iWebKit*, that proved to be, the best all around accessibility platform for general mobile devices multimedia Web applications.

The decisions for ubiquity occur in the server-side and in the application itself. They use the information of each other to decide the best for several parameters, and, ultimately, the best for the user client. This new level of ubiquitous collaboration brings out the best of two worlds—the server-side power, and, the client-side context, location and sensor awareness, making the delivering of news seamless, visually effective, communication efficient, configurable, or ubiquitously personal.

The application resolves some of the wireless networks limitations, reads the context-awareness of mobile devices, communicates with the server and understands the information received.

The proposed applications are conceptually simple as they proved to be the best way. Like the open software, the usage of standards in mobile development is a necessity ("must have"). It opens doors to new applications or services, improves compatibility with other devices, and all is under control of the programmer. The news delivery to all kinds of the readers was also studied. The proposed ubiquitous computing, software design engineering, service architecture, and news content, bring innovation and contribute to improve communication in this modern world.

In a near future, the proposed applications can be improved in several ways. The following items are suggested during tests and debugging stages and can be found in other applications studied. It seems worthy to make it for the application. In the Web applications, the following may be performed: (i) create advanced client information—client history and online statistics that would be added on the ubiquitous system to create more personal, seamless, and user-oriented news content; (ii) add multimedia capabilities to the Web application and Web server (streaming server)—enlarge compatibility with automatic conversion on codecs, containers, sizes, and file formats; (iii) add user registration—access to post comments, news, suggestions, uploading files, images, and slideshows; (iv) add options like—Send content to email; and (v) add them to Twitter, Facebook, LinkedIn, Hi5, or other social networks.

For Native applications, some features for the server side of the Web applications type may also be proposed. We can also perform the following: (i) add location-awareness given by GPS to ubiquitously choose news, language and design for a specific region; (ii) improve native application ubiquity by using more information from sensors, location services, settings, connectivity, and specific device characteristics; (iii) embeded multimedia elements such as video, audio, photo slideshow, and other Web pages without quitting from the application; and (iv) as recommended by Apple improve the software design. "Make it iPhone", by bringing innovation on design, more information on display, new features or services, usability, and accessibility.

Acknowledgments

Part of this work has been supported by *Instituto de Telecomunicações*, Next Generation Networks and Applications Group (NetGNA), Portugal, and by Online Communications Lab (LabCom), University of Beira Interior, Portugal.

References

[1] C. A. Da Costa, A. C. Yamin, and C. F. R. Geyer, "Toward a general software infrastructure for ubiquitous computing," *IEEE Pervasive Computing*, vol. 7, no. 1, pp. 64–73, 2008.

[2] G. Ortiz and A. G. De Prado, "Mobile-aware web services," in *3rd International Conference on Mobile Ubiquitous Computing, Systems, Services, and Technologies, (UBICOMM '09)*, pp. 65–70, October 2009.

[3] T. T. Ahonen and A. Moore, *Communities Dominate Brands: Business and Marketing Challenges for the 21st Century*, Futuretext, 2002.

[4] C. Pettey and H. Stevens, "Gartner Says Enterprise Mobile Phones Will Replace Desktop Phones in North America by 2011," Gartner, 2009, http://www.gartner.com/it/page.jsp?id=874012.

[5] N. Covey, "Mobile Internet Extends the Reach of Leading Internet Sites by 13%," Nielsen Media Research, 2008, http://www.cellular-news.com/story/30926.php.

[6] T. O'Relly, "What is Web 2.0," O'Reilly Media Inc., 2005, http://oreilly.com/web2/archive/what-is-web-20.html.

[7] J. Zhang, A. Hämäläinen, and J. Porras, "Addressing mobility issues in mobile environment," in *Proceedings of the 1st Workshop on Mobile Middleware: Embracing the Personal Communication Device, Co-located with ACM/IFIP/USENIX International Middleware Conference (MobMid '08)*, Leuven, Belgium, December 2008.

[8] M. Weiser and J. S. Brown, "The coming age of calm technology," in *Beyond Calculation: The Next Fifty Years of Computing*, P. J. Denning and R. M. Metcalf, Eds., Springer, New York, NY, USA, 1997.

[9] J. Steele, "comScore Reports 6.5 Million Americans Watched Mobile Video in August," comScore, Inc., 2008, http://www.comscore.com/Press_Events/Press_Releases/2008/10/Mobile_Video.

[10] S. K. Mostefaoui, Z. Maamar, and G. M. Giaglis, *Advances in Ubiquitous Computing: Future Paradigms and Directions*, IGI Publishing, New York, NY, USA, 2008.

[11] E. Sadun, *The iPhone Developer's Cookbook: Building Mobile Applications with the iPhone SDK*, Addison-Wesley, Boston, Mass, USA, 2009.

[12] B. Caldwell, M. Cooper, L. G. Reid, and G. Venderheiden, *Web Content Accessibility Guidelines (WCAG) 2.0*, W3C, 2008, http://www.w3.org/TR/WCAG/.

[13] D. Hazaël-Massieux, "Return of the mobile style sheet," *A List Apart*, no. 275, 2009, http://www.alistapart.com/articles/return-of-the-mobile-stylesheet.

[14] S. J. Zilora and S. S. Ketha, "Think inside the box! optimizing web services performance today," *IEEE Communications Magazine*, vol. 46, no. 3, pp. 112–117, 2008.

[15] D. Mark and J. LaMarche, *Beginning iPhone 3 Development: Exploring the iPhone SDK*, Apress, New York, NY, USA, 2009.

[16] P. Tarasewich, "Designing mobile commerce applications," *Communications of the ACM*, vol. 46, no. 12, pp. 57–60, 2003.

[17] G. D. Abowd, G. R. Hayes, G. Iachello et al., "Prototypes and paratypes: designing mobile and ubiquitous computing applications," *IEEE Pervasive Computing*, vol. 4, no. 4, pp. 67–73, 2005.