University of Beira Interior
Informatics Department

N4MD
News For Mobile Devices

Marco Paulo Novais Teixeira Ramos de Oliveira

Final dissertation submitted for the
MSc in Computer Science and Engineering

Supervisor: Prof. Joel José Puga Coelho Rodrigues

August 21, 2009

This report is a dissertation submitted for the MSc in Computer Science and Engineering, presented in the Informatics Department of the University of Beira Interior, under the supervision of Prof. Joel José Puga Coelho Rodrigues.

This dissertation was written using LaTeX typesetting system, and using the book template. References type are IEEE standards [36].

Acknowledgments

A Dissertation is, normally, a result of a lonely process of intensive work, but in this case, the effort involved several persons, and, at this point I want to show them my gratitude.

Therefor, I want to thank Prof. Joel José Puga Coelho Rodrigues for his motivation and direct guidance since the first lectures.

I also want to express my appreciation to the following entities and their representatives for their support:

The LabCom – Online Communication Laboratory –, my place of work for several years now; their comprehension, understanding, recognition, and respect helped me in a lot of ways. My special thanks to Professor António Fidalgo and Prof. João Canavilhas for their esteem and collaboration in this project;

The DI – Computer Science Department – in general, my teachers and staff in particular;

The IT – Telecommunications Institute – for their support, disposal and equipment. A special thanks to the *NetGNA* – Next Generation Networks and Applications Group – for their fellowship, for letting me use their iPhone 3G, facilities, and for their literature availability.

The UBI – University of Beira Interior – for their availability and services.

Many thanks to my friends, colleagues and co-workers for their fellowship, cheerful and agreeable companion.

Thanks to the untiring support of the usual suspects: my parents and family and a special thanks' to Sofia Pastor, who has put up with too many nerd talks and too many gadgets over the year while remaining both kind and patient at the end of the day.

Abstract

Mobile devices are, today, one of the engines that pull us to an even fast, growing, inter-connected global society. Communications are discovering their seventh way. The development community is adapting today's technologies to a more ubiquitous mobile environment. We urge to adapt our information, services and applications to mobile devices, and more specifically, to, the development platform of the moment, Apple's iPhone.

By trying just that, we've come up with two different approaches: Web Applications and Native Applications. These applications use sensor and context-awareness to select, define and distribute news content, to speed-up communications efficiently, and, to setup viewing optimizations on client-side. We've studied the concepts, paradigms, characteristics and differences, in order to establish a *mobile mindset* in terms of design, effectiveness, responsiveness, power and security.

Keywords

Mobile devices, Portable devices, Mobile computing, Ubiquity, Software engineering, Web, Native, Applications, WebApp, UIKit, iPhone, iPod Touch, SDK, News, Media, Context-aware, Wireless, Internet.

Contents

List of Figures

List of Tables

Acronyms and Abreviations

2G Second-Generation Wireless Telephone Technology

3D Three Dimensional

3G Third Generation of telecommunication hardware standards and general technology for mobile networking

3GS aka 3G S. Apple's designation for it's Third Generation iPhone. Successor of the iPhone 3G. S stand's for speed

A-GPS Assisted GPS. Enhances the startup performance of a GPS satellite-based positioning system.

AAC Advanced Audio Coding

ADC Apple Developer Connection

AIFF Advanced Interchange File Format. An uncompressed Macintosh audio format, functionally identical to Windows' WAV.

AJAX Asynchronous JavaScript and XML

AKA Also Known As

API Applications Programming Interface

APP shorthand for Application

AUTO shorthand for Automatic (operation)

AV Audio and Visual

AVC Advanced Video Coding

Bluetooth Open wireless protocol for short-range radio seamless communications

CDN Content Distribution Network

CMS Content Management Systems

CPU Central Processing Unit

CSS Cascading Style Sheet

DI Informatics Department

DNS Domain Name System

DOM Document Object Model

DRAM Dynamic RAM

EDGE Enhanced Data rates for GSM Evolution

EDR Enhanced Data Rate. A Bluetooth specification which introduces bandwidth capacity of up to 2.1Mbps

EM Is a unit of measurement in typography. It's equal to the point size of the current font

GB shorthand for GigaBytes

GCC GNU Compiler Collection

GIF Graphics Interchange Format

GL Graphics Library. Aka OpenGL – Open Graphics Library

GNU stands for "GNU's not Unix". It's denomination for software distributed under the GNU Public License (GPL)

GPL aka GNU General Public License is a widely used free software license

GPS Global Positioning System

GPRS General Packet Radio Service

GSM Global System for Mobile Communications

GUI Graphical User Interface

GZIP Shorthand for GNU zip. Is a software application used for file compression

HLR Home Location Register

HTML HyperText Markup Language

HTTP HyperText Transfer Protocol

HTTPS Hypertext Transfer Protocol Secure

I/O Input/Output

IB Interface Builder

IDE Integrated Development Environment

IEEE Institute of Electrical and Electronics Engineers

INFO shorthand for Information

Internet Internet is a global system of interconnected computer networks

IP Internet Protocol

IT Telecommunications Institute

iUI iUI is an open source user interface library for iPhone web application (webapp) development

JPEG Joint Photographic Experts Group responsable for the file format commonly used for image compression

KB shorthand for KiloByte

Kb shorthand for Kilobit

LabCom Online Communication Laboratory from UBI

LAN Local Area Network

LAPP Linux (OS), Apache (Web Server), PostgreSQL (Database Management System), Programming language (PHP, Perl or Python)

LCD Liquid Crystal Display

Mac Shorthand for Macintosh. Is a series of several lines of personal computers designed, developed, and marketed by Apple Inc.

Mbps Megabits per second of data transferred

MHz MegaHertz: One million periods per second

MP3 MPEG-1 Audio Layer 3

MPEG Moving Pictures Experts Group

MPEG-4 Is a collection of methods defining compression of AV digital data

Mpx Megapixel also known as MP

N4MD News for Mobile Devices

NET aka Internet

NetGNA Next Generation Networks and Applications Group

NS NeXTStep. The original code for the Cocoa frameworks that came from the NeXT computer company formed by Steve Jobs in 1985

OO Object-Oriented

OOP Object-Oriented Programming

OS Operating System

PDA Personal Digital Assistant

POSIX Portable Operating System Interface for Unix. Is a family of open system standards based on Unix or related standards specified by the IEEE

POST Information transferred or assigned from one location to another, normally from a client to a server

PC Personal Computer

PHP Hypertext Preprocessor scripting language

PPI Pixels-per-inch

PX Pixel

RAM Random-Access Memory

RIM Research In Motion Limited. It's a Canadian wireless device company developer of the BlackBerry smartphone

RSS Really Simple Syndication

SDK Software Development Kit. Also known as devkit

SIM-Card Subscriber Identity Module Card

SMS Short Message Service

SOAP Simple Object Access Protocol

SQL Structured Query Language

SVG Scalable Vector Graphics

TCP Transmission Control Protocol

UBI University of Beira Interior

UDDI Universal Description Discovery and Integration

UI User Interface

UIKit User Interface programming kit. UI framework in the IPhoneRuntime, the equivalent of AppKit for traditional OS X apps

UK United Kingdom

UML Unified Modeling Language

URI Uniform Resource Identifier

URL Uniform Resource Locator. The address of a web page on the world wide web. Is a type of URI

USA United States of America

XHTML Extensible Hypertext Markup Language. A reformulation of HTML as an XML application

XIB Xcode Interface Builder files extension

XML Extensible Markup Language

XSLT Extensible Style-sheet Language Transformations. Is an XML-based language used for the transformation of XML documents into other XML

XP shorthand for Experience on Microsoft's Windows XP line of OS's

W3C World Wide Web Consortium. Is the main international standards organization for the WWW

WaN aka WebApp.Net. Open source Web Application framework by Chris Apers for the iPhone

WAN Wide Area Network. Is a network of LANs covering a multi site, national or even global area

WAV aka WAVE. Waveform Audio Format developed by Microsoft and IBM

WEB aka Internet

WebKit Application framework that provides a foundation upon which to build a web browser. WebKit was originally derived by Apple Inc.

WHATWG Web Hypertext Application Technology Working Group. Community of people interested in evolving HTML and related technologies

Wi-Fi Wireless Local Area Network. Wireless LAN. Wireless Fidelity. Wi-Fi is a trademark of the Wi-Fi Alliance for certified products based on the IEEE 802.11 standards

WiFi aka Wi-Fi

WML Wireless Markup Language

WWW World Wide Web

WWDC Apple Worldwide Developers Conference held annually in California by Apple Inc.

Chapter 1

Introduction

> "It has been said that the cell phones changed more the everyday life of people than the Internet." (Unknown author)

1.1 Why we choose this subject

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 his content to adjust the new needs. Despite the business model is restricted to cost-per-use, with operators maintaing a high cost and network restrictions to avoid massive usage, data porting to mobile devices hasn't really taken off.

The mobile devices fulfills the true aim of Internet to offer full connectivity anytime anywhere. The trend to go 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 that users enjoys using their mobile devices to speak everywhere without the need to log in a local wireless network is to extend to other Internet services as surf the Web, check email, read the news, listen to online radios or even watch video and television streaming.

Mobile devices have more and more embedded technology. People have more mobile devices and use them more often to view Internet content. We want to explore and take advantage of this technology, study people habits, understand their perspectives on this new world order, and, with an eye on the future to create new ways of content distribution.

1.2 Background

After the introduction of the PCs in the eighties, it was in the nineties that the PCs married the phone line and the Internet became popular as the invention of the hypertext markup language in 1990 by Timothy Berners-Lee made possible the World Wide Web (WWW) [18]. The personal computers started as very convenient typewriter and soon they became the universal office tools machines

with word processors, spreadsheets, and databases programs. When they got connected to the Internet their main use changed, people started to use them as a communication central, to send and receive e-mail, to send and receive documents as a replacement of the fax machines, to download and exchange text, music and pictures files.

Now, the laptops with wireless technology and incorporated microphone and camera go a significant step further in the process of making the Internet the universal medium. It points very clear the trend direction of the Internet: downsizing and portability of the devices and universal broadband coverage. Internet meets the mobile telephony.

More and more we get the news from the Internet. The traditional print press is fading, the television channels loose audiences, mainly among the younger, and the listening of radio is confined to the times we sit in a car. To inform and to be informed are activities that increasingly are being done online and journalism is facing one of the major revolutions in its secular history. The production of news changed; the computer assisted reporting is so obvious as the computer assisted design and no reporter can do his job without a computer connected to the Internet. The consumption of news changed; the reader can surf the Web, follow the links of a new, post a new, a comment, and email the new to a friend. And the language of the news changed; the language is leaner, the most must be said in the minimum, words join pictures and video clips to a multimedia symbiosis.

1.3 Objectives

The goal is to 'create' and distribute news and specific content for mobile devices by creating an unique ubiquitous Internet service. We want to do this by creating two types of applications – Web Applications and Native Applications. Web Applications are to be viewed in mobile devices in general. Also, we want to deploy specific Web Application versions for the Safari mobile browser.

Regardless to Native Applications we want to install them on the iPhone/iPod Touch. Also, we want to be able to run them without Internet connectivity.

We want to design better Internet services to users providing an uncompromised perspective on data usage in mobile devices. We want to use well-formed documents, Web standards, and software design best practices for user interfaces, in order to guarantee usability and accessibility. The delivery system will have sensor and context-awareness, default parameters and users pre-defined settings to evaluate how the information is showed, what, and how it will be transmitted.

Ultimately, we want to improve mobile devices, communication, server side computation processes, and, ultimately, user experience and usability. The benefits for clients are selecting what and how they want to see and Internet data transfers costs and speed. The benefits for mobile devices will afect battery, memory, processor and Internet performance. The benefits for networks and servers are bandwidth, speed and storage costs.

1.4 Main contributions

The systems created drove wireless networks, mobile devices, users and news content in an unprecedented level of flexibility, convenience and ubiquitous decisions.

The Web applications combine the portability of a Web service, the mobility of users, and overcome the limitations of small screen size, low memory and limited processing capability to conveniently distribute information. A specific version can be viewed in all mobile platforms with the same layout and without usability problems.

Our Native Application resolves some of the wireless networks limitations (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. Our solution combines ubiquitous information in both sides of the provided system – the server side (by calculating image quality, item numbers and format of text) and the mobile device side (deciding if downloads new information or images). This ubiquitous collaboration between mobile device and Web service is a new form of intelligent applications that brings-out the best of two worlds – the server-side power and the client-side context, location and sensor awareness.

We also provided new means do distribute news content to the online journal. By creating several versions and the native application we get user feedback and statistic informations about the readers and usage. This is important because user interface design is also critical for the software success.

More, for journalists, and all the news related area professionals[1], it's important to know if mobile devices are just an Web page extension or is it a new way of communication, more specifically – the seventh way of communication [1].

By developing for the iPhone/iPod Touch devices, which was a new platform to develop for in our community, we brought new levels of local knowledge and experience for future jump-start developers.

1.5 Methodology

Location-aware and context-aware techniques, as a part of ubiquitous computing, will be used to define how, when and how fast the content is delivered to the 'client'.

We embrace the mission of programming for these new mobile devices, networks and technology keeping in mind the cost and speed concerns.

The methodology followed was to create two types of software application prototypes. One for Internet browsers and one to run natively in mobile devices.

We will describe the capabilities and attributes of the iPhone and his Software Development Kit (SDK) and present our experiences. We applied for the Apple iPhone Development Program (and then, to the University Program[2]) to

[1]The Pew Research Center for the People & the Press. http://www.people-press.org

[2]The iPhone Developer University Program allows students within the same development

initiate the learning process. We've learned from the tutorial videos, to their development resources, code examples, lists and community, and, distributing them in the iTunes App Store.

We also did an extensive comparison, testing and analyses of other free applications developed from entities with the same interest, which is, delivering media content to their users.

The development of the ubiquitous applications was the most ambitious objective in this work, as it would be used and distributed worldwide, and, therefor, subjected to reviews, criticism and feadback.

1.6 Dissertation organization

This dissertation includes chapters that present the emerging applications and services, the technology foreground of ubiquitous devices, and the developed applications for this systems. In particular:

In chapter 2 we will describe the related technology and background to make ubiquitous computing for nowadays systems. How and why we choose the several devices and platforms to work with.

In chapter 3 we will talk about the technology, specifications, tools and methods we've choosed and used to create our systems. This is a superficial introduction to one of the most potentially ubiquitous devices in the market, and it's related tools and frameworks to develop for.

In chapters 4 and 5 we will show how we created systems intended to distribute news content in a optimized way. The 'optimized way' was incorporating ubiquitous technology in the process. We took advantage of the mobile devices features, and characteristics, to combine location, information and context-aware capabilities to optimize the news and media content.

Finally, the chapter 6 is dedicated to the result conclusions of our challenge to make real-world applications for ubiquitous computing. We will also talk about what can be done in the area as future work.

team to share their applications with each other through email, or by posting them to a private Website for presentation and grading purposes. In addition, higher education institutions can submit applications for distribution in the App Store.

Chapter 2

Related Work

In this chapter, we will overview the world of mobile computing. We will talk about mobile devices, their features, characteristics, Internet media content, Web services, ubiquitous systems, and related applications.

2.1 Portable devices

Portable devices, also referred as hand-held, mobile devices or wearable devices, such as mobile telephones and Personal Digital Assistant (PDA), are small and lightweight equipment than can be fit into a suit pocket, hand or briefcase. The ones we will be talking about are those with Internet connection and a small screen, such as mobile phones and PDAs. These gadgets can perform a big number of tasks, such as: Short Message Service (SMS), email, packet switching for access to the Internet, gaming, bluetooth and Wi-Fi[1] connectivity, infrared, photo camera and video recording, music player, radio and Global Positioning System (GPS) antennas, memo recording, and, more importantly, make and receive phone calls.

The Nokia Communicator, released in 1996, was the first mobile phone to enable the PDA functionality, Internet connectivity and wireless email. Today is a standard phone. [37]

There's a screen limitation in this devices but all the content available to the other kind of screens (Cinema, Television and Personal Computers) are being produced accordantly mobile characteristics. The content from other types of Media such as Print, Recordings, Radio and Internet, is also being imported to this kind of devices. The services had evolved with the phones and in 1999 the first mobile Internet service was launched in Japan by NTT DoCoMo[2] under the i-Mode service. The media services began with ringback tones. Today we have mobile content such as "mobisodes" - video content that has been produced exclusively for mobile phones.

[1] Also know as Wireless Local Area Network – Wireless LAN –, is a trademark of the Wi-Fi Alliance for certified products based on the IEEE 802.11 standards.

[2] NTT DOCOMO Global Home: http://www.nttdocomo.com/

In Portugal, the estimate mobile cellular phones was, in 2007, of 13 413 million. Not all of them has browser or media capabilities [20]. In the world, back in 2007, the number was 3.3 billion - which is half of the human population. In the third quarter of 2008, the penetration of mobile phone services was 137% [28]. 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" [53].

We can predict that it will be de device of the future, accordantly to Gartner[3] [32], the enterprise mobile phones will replace desktop phones in North America by 2011.

The fact that the number of people that accessed the Internet via their portable devices had increased by 25%, in the second and third quarters of 2008 [19], and the audience is younger predicts that the contents for these gadgets will continue to grow. In fact, for the generation of USA Internet users between 18 and 24 years, the preferred consumer electronics, in 2007, was the mobile phone (47%) over the computer laptop or desktops (38%). [27] Indeed, as a 2008 Nielsen Media Research report highlighted, mobile devices have increased traffic by an average of 13% across several popular Websites. [42]

2.1.1 iPhone's family

The iPhone is more than a gadget or an high-tech toy. It's a revolutionary mobile device that brought innovation to the cell phones market by integrating a phone, an iPod and a Internet communicator. All this in a slick, multi-touch, wide screen and friendly format, making him one of the most successfully "technolust" [40] devices thru the world.

iPhone 3GS

The iPhone 3GS is the latest offering from Apple Inc., being the third generation of the revolutionary phone. It was announced in June 8, 2009 and In Portugal it was for sale in late July 2009.

iPhone 3G

The iPhone 3G is the second-generation iPhone, was released in July 11, 2008 in 22 Countries, the first model, the iPhone, was released a year before, back in 2007 only in the United States of America (USA). According to the press releases of Apple the numbers of the sells are: During the first quarter that ended by September, 2008, 6.9 million units were sold;

[3] Gartner, Inc. Information technology research and advisory company. http://www.gartner.com

iPhone

The first-generation model, debuted in June 29, 2007, in the prior five quarters of activity sold 6.1 million; In the second quarter of activity the company sold 4.4 million units. This makes the iPhone 3G the first product achieving the 10 million mark in just a few months. [7]

iPod Touch

After the iPhone, later that year, Apple redesigned his iPod line and created a new iPod Touch, based on the iPhone, with network and browsing capabilities but without camera and Global System for Mobile Communications (GSM), which extended the classic music player and bringing him to the portable devices category.

> From now on, we will be referring to the iPhone, iPhone 3G, iPhone 3GS and iPod Touch simply as iPhone.

Overall, the iPhone package is more intuitive with better 'design'. When we say 'design', we are talking about the 'looks' of the device, and, more important, the friendly user experience or 'feeling' of the software. The interaction with the user. The 'just do it', 0-flaw and reliable philosophy. The iPhone, is a award-winning gadget, is today's benchmark, the device that other are trying to match. Some say it's a device that Apple and Steve Jobs brought from the future with it's Time Machine.

Alongside the iPhone, Apple launched a new Software Development Kit (SDK), that allows developers to create applications, and deploying them to the iPhone thru the App Store.

So, what's the fuzz about this gadget and his development kit? We will talk more about it later in the chapter 3.

Some people predicted, in 2008, that what's on the iPhone today will be on other phones in the future [30], and, looking at the mobile market today, the industry is heading that way.

2.1.2 Other devices

There are other devices on the market and with similar characteristics as the iPhone, for example, T-Mobile[4] G1 and HTC[5] Magic (Google Android[6]), Nokia phones, Palm Pré[7], and Microsoft Windows Mobile smart phones[8], that are equally capable of networking and Web browsing. They also have location-aware

[4]T-Mobile International AG. http://www.t-mobile.net/

[5]High Tech Computer Corporation (HTC). http://www.htc.com

[6]Android Developers. http://developer.android.com/

[7]Palm Developer Network. http://developer.palm.com/

[8]Windows Mobile Developer Center.
http://msdn.microsoft.com/en-us/windowsmobile/default.aspx

capabilities with GPS or GSM triangulation. They now have accelerometers and other kind of sensors. Some of them have more or better features than the iPhone. They also have their own development platforms, and, some have their code open to the community for developers, but, producers keep the codes and the operating system specifications to themselves.

Microsoft uses the Visual Studio and .NET Compact Framework. Palm now uses the JavaScript based WebOS platform (the Palm Mojo Framework) and used to be the PalmOS. Nokia uses the Symbian OS[9] and specific phones frameworks or SDK's. Blackberry[10] use the RIM[11] Wireless Handheld Software Developer's Kit. In 2008 the Google launched a phone with software stack for mobile devices that includes an operating system, middleware and key applications, and there's a movement called OpenMoko[12] [43] that intends to produce a fully customized operating system, tools and services for mobile devices. This free version of the software is based on a linux distribution called Debian. [47]

2.1.3 The Software Development Kit and the App Store

The SDK and the App Store was the the most significant announcements surrounding the iPhone in 2008 [25]. Because of the 'on-click-download-design' of App Store (July 2008), it's simple for the customers to download/purchase the applications using the iPhone using the desktop and an iTunes Account. They can also use the desktop application for Mac OS X or Microsoft Windows for synchronization and installing applications later in the iPhone.

The applications catalog is all there, at least without hacking you phone [46] and, is divided in major categories, there's customers review, top selling applications, the featured, evaluation stars, search and automatic updates.

From the developer point of view, it's equally easy to offer or sell their software by adhering to the authorized iPhone Program. The downside is that Apple takes 30 percent of the price, there's stifling competition and the you can't sell products/services that compete with Apple itself.

The predictions on App Store where to be successful, like the iTunes Store that sold more than four billion songs, and maintains a catalog of more than six million songs [23], but nobody predicted the tremendous impact. It started off with less than 500 applications, and by February 2009, it had more than 20 000 applications [24], 25 000 in April 2009, 35 000 in May 2009, and the torrent seems to be growing larger each day [21]. The customers of App Store downloaded more than 300 million applications until the end of 2008 and in April 2009. Nine months after opening, they achieved the 1 billion mark. [23] [5]

Before the App Store were launched, new programs appeared on the iPhone. But the device itself had to be *jailbreaked* using special software tools to "breaking" it up. The ordinary users of the iPhone didn't bother with it, and the

[9] Symbian Developer Network. http://developer.symbian.com
[10] BlackBerry Developer Zone. http://na.blackberry.com/eng/developers/
[11] Research In Motion (RIM). http://www.rim.com/
[12] Openmoko Developer. http://wiki.openmoko.org

operation wasn't foolproof because of the Apple software updates could render it inoperable, requiring a complete replacement.

2.2 Web Services and Media Content

The Internet is on it's second generation in terms of Web development, social, design and business revolution – Web 2.0 – as the O'Reilly team mentioned [44]. In this phase, the interconnectivity and interactivity of Web-delivered content, that were born for the desktop computers, have to move on and adapted to mobile devices and other portable devices. This adaptation is taking the best of what this devices have to offer – portability and connection everywhere –, and, the Internet companies are creating new and innovating services for this market.

Generally, the integration techniques used to combine Web services and mobile devices are: Socket communication, Web services, and, Messaging techniques.

Initially, and because of the lower computational skills of mobile devices, Web services were one of the best ways to share the computing capabilities of servers and desktops.

Web services uses Extensible Markup Language (XML) and Simple Object Access Protocol (SOAP) to provide a mechanism that facilitates the exchange of data 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 facilitate interoperability across different hardware and software implementations.

The Application Programming Interface (API) developed today such as Google, Amazon and eBay, have robust Web server integration with desktops but when it comes with mobile and lightweight applications the challenges are different: Retrieval of significantly data info; Performance; Multiple round trip communications; Memory; Security and Information display limitations. To overcome this problems mobile computation is developing techniques such as caching, large data set handling, compression, paging, filtering, just in time techniques, information on demand and speed testing.

The fuzz about media Web services can be explained by observing the chart about Mobile Media Consumption in the United States (table 2.1). [22] The number of news and media content downloaded thru Internet and portable devices will increase. So, the number of these kind of Media Web Services will increase, also.

If we look to the typical iPhone owners they are at least three times more likely than average to visit several popular social/communication/entertainment sites [52] – the so called infotainment. The major newspapers and media groups of the world have, already, the iPhone Application for delivering the news in this new format.

So, what mobile Web services we can choose? and what about the design guidelines? or the design and development best practices? This things were considered and kept in mind when plan mobile Web services.

Mobile Media Consumption

Activity	Subscribers	% Mobile Subscribers	% Change
News & Info via Browser	36,185	15.9%	1.3%
News & Info via Download	13,274	5.8%	7.0%
IM	21,032	9.2%	6.1%
News & Info via SMS	18,727	8.2%	4.0%
Social Networking	14,947	6.6%	8.8%
Email (Work & Personal)	33,564	14.7%	4.5%
Purchased Ringtone	20,124	8.8%	-2.0%
Purchased Games	5,478	2.4%	-1.0%
Used Network for Photos/Video	59,877	26.3%	-1.1%
Listened to music	19,001	8.3%	-2.2%
Received SMS ads	48,943	21.5%	-0.5%

Table 2.1: Mobile Benchmark Study of Mobile Media Consumption. Three Month Average Ending August 2008 vs. Same Period in 2007. Total U.S. Mobile Subscribers. Source: comScore

2.3 Ubiquitous systems

Mobile devices are getting smaller and powered up with add-ons, speed and battery life. Combining this, with the growth of short range ad-hoc networks, the possibility of accomplish the vision of ubiquitous computing that was sketched out in the early 90s is getting closer [50] [41]. With this 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.

2.3.1 Seamless integration

The ubiquitous computation is the omnipresent relationship of things 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.

After the two great trends - the mainframe relationship and the Personal Computer (PC) relationship - Mark Weiser, the 'father' of ubiquitous computation, predicted that, the Internet would lead us through an era of "deeply imbedding computation in the world". The approach to alter the place of this technology in our lives was called "calm technology". [50]

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 a ubiquity context.

This calm technology has the ability of going from the periphery of our

attention, to the center, and back again. It's there, we know it, it's doing his job, his computations, his communications with other devices, managing the relationships, empowering our periphery but not moving to much the the center of our attention. The impact on everyday life is already huge, it's becoming so commonplace like writing or electricity.

2.3.2 Location-aware systems

Mobile devices with GPS capabilities have been around for a while now, but, location-aware with iPhone is changing our life. This device will put into the mainstream the location-aware applications [45]. From pinpointing our location on a Google Map (which came with the first version), to tracking friends, giving a heads-up on what's going on in your area. The nearest place to eat, list the shopping areas near you, where to party, and other businesses.

The first application for the iPhone that use the faux-GPS feature (which used cell tower information to triangulate your position) was Google Maps. Now, there's a big number of applications that use real GPS feature and service. Here's some examples of Location-aware Applications that can be found in Apple App Store: Loopt – Friend-finder application with virtual earth display. 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 everyday.

2.3.3 Context-aware systems

Mobile devices are equipped with wireless capabilities and users can go through several contextual changes as they move around. This changes are related to the movement of the user in his physical and social surroundings. By sensing their environment, mobile devices are capable of communicate and deliver ubiquitous services adequate to the situation.

Our ubiquitous system uses some of the device sensors to adapt, communicate, manage data and display information has the context changes.

As the device context changes, a interoperation with the server is made, and, if necessary, the system will re-do the content. 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. This can be observed in the figure 5.6.

2.3.4 Sensors and device settings in the ubiquitous system

Lots of mobile devices and operating systems already have landscape and portrait modes, but they need the user interaction to do the change, and, to do this, the user needs to change of application. Sometimes changing of application involves quitting.

The iPhone accelerometer is an example that shows how the context-awareness in the ubiquity systems can be used. Some applications like the Calculator.app, iPod.app and the Safari.app, modify their appearance or functionality as the position (landscape or portrait state) of the device changes. The state is given by the three dimensional accelerometer sensors which gives us instantaneous knowledge of movements: Rotate, Pivot, Spin and Tilt.

Some of the following integrated systems are already a form of ubiquitous systems, some of them in the iPhone. These were some of the questions and thoughts that we considered for creating our application. The list is grouped by the two well known types of ubiquitous systems.

1. Location-aware group

    - The GPS is the most commonly used. GPS location-aware capabilities can be used to document where all your photos were taken – geotag your photos – or show us the location of near by restaurants. It can be also used to determine if you are moving.
    - GSM triangulation is also a form of 'faux' GPS feature.
    - Internet Protocol (IP) location database plus Domain Name System (DNS) code area resolution mechanism is used to differentiate news by area.
    - Network connection type and speed. GPRS, EDGE, 2G/3G networks or Wi-Fi network will influence the ubiquitous system by knowing what network is the device using and what (and how much) kind of information will be delivered.
    - Bluetooth can be used to determine if network is available or what kind of connected object i can use.

2. Context-aware group

    - Context-awareness by knowing the type of nearby object. If you get near a printer the system can adapt the functions and display;
    - Movement sensor using the integrated camera device;
    - Headsets plug-in detection and sound volume of media and device;
    - Application type running in the device. Web or Native Applications;
    - Battery capacity and current status;
    - Memory capacity and availability for the application and data;
    - The accelerometer gives 3D status, motion detector and screen width size detection;
    - The proximity sensors are used to know when the phone is near objects;
    - Ambient light sensors to determine how we adapt the colors and content palette. In the iPhone it re-calibrates only once, each time you unlock the phone after waking it;

- Silence button - we use this to know if the content delivered will include embed audio or audio options;
- Other applications status. Are they active or sleeping? Are they using resources we will need? What to do if they became active? Will we adapt ubiquitously or notify the user? For example, the iPhone allow us to listen to iPod music while using another application. So, we can use this to make content without audio.

2.4 Requirements and Design

We want to work with standards and known frameworks to build seamless interfaces and services. Following the best practices in code development and design. Ultimately, to produce a good application and Web platform that users experience, follow and recommend.

The advantage of mobile is the immediacy. When it comes to instant access to weather, email or sports news the portable devices are their first choice. Knowing that, the success happens because of the small amount of content delivered. News Web sites have more visits then the search companies.

We believe that mobile experience merits its own design, customized to their needs and 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, then they go to a computer screen, so, developing for these screens and devices brings new issues, paradigms and semantics.

Nevertheless, the mobile Internet is about functionality opposed to entertainment and e-commerce on the screen based systems. [19]

2.5 Related Applications

We've selected some applications that illustrates the best practices for iPhone development. Those applications combines the power of the Internet with the simplicity of Multi-Touch technology, all on a 3.5-inch screen.

Obviously, we have a special interest in Applications related with News. In our case-studies, we've selected some examples in order to analyze, study and understand possible influences in our own application.

Web Applications

There are some portals on the Internet that links specifically to iPhone Web Applications. Apple[13] has it own portal with more than 3950 Web Applications, the iPhone Tookbox[14] site indexes more than 500. They are organized by category or by Most popular, Most recent, Alphabetical, or, like the Apple site by Staff picks.

[13][Online] Last visited on May 27, 2009: http://www.apple.com/webapps/
[14][Online] Last visited on May 27, 2009: http://iphonetoolbox.com/category/webapp/

Native Applications

We will present a full study of iPhone Native Applications on chapter 5. Those applications are 'news' related, but there are others distributed by several categories (Games, Entertainment, Music, Utilities and so on). App Store also divided the applications by cost: Payed or Free. At the time of the one billion download mark in April 2009 (nine months after opening the App Store) fourteen over twenty apps were in Games category. [5]

PointAbout Application and Technology

Their application was one the first using location-aware capabilities to let users know info related with de device location. This was done by inquiring several Web Services on the Internet. Metro, restaurants, conferences, Medical Care were among the services they offer.

This application worths mention because it uses the two different forms of iPhone development: Web Applications and Native Applications. The applications consists on a Native Application that gathers the device information and status, combines that with the location-aware capabilities, quits itself to bring-up the Safari Mobile browser application of the iPhone with a specific URL. This URL leads the user to a Web Application.

Chapter 3

Developing for the iPhone

> "Different isn't always better, but better's always different" (Jonathan Schwartz, Sun's CEO)[1]

To develop applications for the iPhone there's a few things we need to know or ware-about: The device itself, the Operative System (iPhone OS), the tools (Xcode, Instruments, Dashcode, and Simulator), the programming language (Objective-C), and, the User Interface architecture (Cocoa touch, Core Media, Core services and other frameworks). We will cover all these things later in this chapter.

Figure 3.1: The iPhone 3G

Key practices and the device features are fundamental for iPhone developing. There's application design best practices and hardware related issues like battery

[1] August 2007. http://blogs.sun.com/jonathan/entry/better_is_always_different

life time, memory management, Internet connectivity, and usability must be considered and they are extremely important.

First-things-first

We need to do two things to start developing for the iPhone. First, we need to head over to Apple Developers Site[2], register for an account and download the iPhone API[3].

Secondly, pack up with good tutorials and books about developing for the iPhone. As a newbie in Apple platforms and to the Objective-C language, it was nice to notice a lot of beginning material supporting new comers[4]. Books, movies, tutorials, podcasts, examples, sites[5], applications, forums[6] and lists[7] are available on the Internet.

Note that while the SDK is free, deploying to a real iPhone or iPod Touch is not, even if it is your own [31]. So, to do that, you have to apply to the iPhone Developer Program which is $99 (about €75), the Professional Program $299 (about €226) or the University Program and benefit the guides, examples, videos, lists and forums in the site.

3.1 The iPhone - Overview

In the February 2007, Steve Jobs in the Apple Worldwide Developers Conference (WWDC)[8], first revealed the device that "is a phone, an iPod, and an Internet navigator". But Apple innovation wasn't the concept itself – Microsoft Windows Mobile could do all this –, it was the all-around design package (look-and-feel, seamless usability, network connectivity, rock-solid operating system and applications, etc).

His killer features includes:

- Multi-touch capacitive touchscreen technology that can detect and interpret two or more simultaneous touches [40] as the input user interface. No need for a stylus, other tool, or, more than one single button. There's new ways to input thru gestures and touches:

  Bubble touch and hold;

  Flick touch and flick for scroll page;

  Flick, Two-Finger touch and flick with two fingers for scrollable elements;

[2] Apple Developers Site: http://developer.apple.com
[3] iPhone Dev Center. http://developer.apple.com/iphone/
[4] iPhone Development Central. http://www.iphonedevcentral.org
[5] iPhone Central. http://www.macworld.com/weblogs/iphonecentral.html
[6] Apple Developer Forums: http://developer.apple.com
[7] Apple Mailing Lists: http://lists.apple.com
[8] WWDC 2007 on Wikipedia. May 2009.
http://en.wikipedia.org/wiki/WWDC#WWDC_2007

Pinch for zooming in or out;

Tap a single tap for selecting;

Tap, Double a double tap permits zooming a column in Safari.

- 3.5", [16:9], 480x320-pixel reflexive screen. 163 pixel-per-inch (ppi) – one of the best in this kind of gadget;
- The portrait or landscape ubiquitous mode. Gives either 480-pixel wide or 480-pixel tall screen;
- Wi-Fi LAN capabilities using 802.11g protocol, up to 54 Mbps;
- Accelerometers that detect orientation and movement. It's a 3D space sensor with relation to gravity, providing orientation awareness;
- GPS – Although the original version doesn't support GPS, it offered "pear-to-peer" location detection [2] that gives location based on cell towers and Wi-Fi physical placement. The iPhone 3G has a true Assisted GPS (A-GPS) that enhances the location-aware of the GPS with cell tower network information;
- iPhone OS – This is ultimately the killer factor. The "just-works" philosophy of Apple in it's ultimate machine.
- The seamless synchronization capabilities with iTunes (Mac and Windows), and lately, the Apple App Store for downloading and installing applications.

For iPhone platform developing we have to know some of the difference between versions and devices. Some of them are important for the implementation of our intelligent system. The table 3.1 is a comparison between iPhone and iPod Touch versions.

Other features, also important, are the support for different formats of audio (Advanced Audio Coding – AAC; Audio Interchange File Format – AIFF; Apple Lossless; Audible, MP3, and WAV), and video (H.264[9] codec – Baseline Profile Level 3.0; QuickTime; and MPEG-4 Part 2 – simple profile), and video files with the following extensions (.mov, .mp4, .m4v, and .3gp) [49]. Note that, for network and file size optimization the movies must be encoded with the H.264 codec. It's an open standard that provides high-quality video and audio.

The networking over the Global System for Mobile Communications (GSM) uses two type of wide-area connectivity. The EDGE[10]/GPRS[11] is supposed to deliver up to 220 Kilobits per second (Kbps) [46] depending the distance from the cellular tower, was used in the first iPhone model and was used if no local-area wireless access was available. The iPhone 3G supports the third-generation standard with speeds from 384 Kbps up to several Mega bits per second (Mbps).

[9] Also know as MPEG-4 Part 10, or MPEG-4 AVC (for Advanced Video Coding)
[10] Enhanced Data rates for GSM Evolution (EDGE)
[11] General Packet Radio Service (GPRS)

Specifications	iPhone iPhone 3G	iPod Touch iPod Touch v2	iPhone 3G S
CPU Speed	412MHz ARM CPU		620MHz ARM Cortex–A8
Memory	128MB Dynamic RAM		256MB DRAM
Flash Memory	4–16GB	4–16GB	16–32GB
GSM quad-band	Yes	No	Yes
EDGE/GPRS	Yes	No	Yes
3G connectivity	No/Yes[3G]	No	Yes
Wi-Fi & bluetooth	Yes	Yes	Yes
GPS	No/Yes[3G]	No	Yes
Camera	2.0 Mpx	No	3.0 Mpx with video
External Speaker	Yes	No/Yes[v2]	Yes
Volume controls	Yes	No/Yes[v2]	Yes

Table 3.1: iPhone, iPhone 3G, iPod Touch and iPhone 3G S specifications and main differences

The priority order in this devices we're similar to the original phone: 1 - local-area wireless; 2 - 3G because it's slower than Wi-Fi but faster than EDGE/GPRS; 3 - EDGE/GPRS.

In terms of networking, and connectivity, it also have a Bluetooth wireless device. It's an Enhanced Data Rate (EDR) version of the Bluetooth 2.0 allowing for a 3.0 Mbps signaling rate data transfer of 2.1 Mbps.

The list of limitations, features available, and other characteristics such as browser of the iPhone will be referred later in this chapter.

3.1.1 Features that we can use in the iPhone

We will use some of the following features available on the iPhone 3G (figure 3.1) to develop our Web application and iPhone application.

- Data Network capabilities (GPRS/EDGE, 3G, Wi-Fi, Bluetooth);
- GPS antenna receiver;
- Screen display, 16:9 aspect, and its pixel-quality. Brightness control;
- Landscape and Portrait modes for changing between lists, reading, text-input, and some applications;
- Sound quality thru choice of speaker or Earbuds and Earbuds insertion detection;
- Vibrating and ringtones;
- Sensors luminosity/light, ear proximity and 3D accelerometers;
- Adaptable Processor Speed and power;

- Flash Storage capabilities;
- Camera for detecting motion and picture capture;
- iPod player and Mail status and activity;
- Battery consumption and capacity;
- Design software and usability;
- The SDK. Xcode Tools, documentation and code examples;
- The App Store to deploy the applications directly to the device;
- Multi-platform integration over iTunes Desktop application (Microsoft Windows and Mac OS X) and synchronization possibilities thru Internet (Mobile Me, Google applications);
- Device unique identification, security, date and time.

3.1.2 Advantages to develop for the iPhone

For lots of reasons. Despite we already talk about the iPhone competitors, see chapter 2, here we will an approach on the differences to the competition.

The iPhone stands up from his competitors in several things, but, it's not a desktop either. Physically, it's a all-screen device and buttonless design, no keys, no stylus, no trackballs or scrolling wheels, and uses finger touch-screen as an input method. But, as we already said, it's the user design-feeling and seamless usability that makes it unique.

In [2], the expression *smarterphone* is used to describe the iPhone over it's competitors. The Internet experience is a better all-around set of features: Safari usability and all-sized zoomed versions of the know Websites; Good speed networks and multiple choices; Standard Internet programs such as YouTube, Weather, Stock market program, Maps, and, Mail.

The SDK provides a good platform for application development for the iPhone. It has tools that help development and optimization, comes with an emulator for testing and uses an object oriented language, we will talk about this platform later in this chapter.

The physical form, the features, the operating system, the SDK, the usability, the use of Web standards, the emerging technologies on mobile computing since 2007, etc., are things that make us want to start developing for this cutting-edge device.

3.2 The Operating System

The iPhone Operating System (iPhone OS) is a striped version of the Mac OS X. In terms of the Operating System itself, there are things that works differently

from the other mobile platforms, and, that affects us as developers and we do programming. There are four key things that affects developers in a day-in and day-out bases.

Code Signing If we want to deliver applications for every iPhone out there, we must have a code signature for the application. This is a requirable feature for applications distribution in the App Store. It uses a certificate obtained from the developer program. When we build the application this certificate is used to signature the application. [8]

Single Application Environment Although the iPhone OS is multitasking and multi-threaded, only a few processes can be always active, and, in terms of Third-party applications only one can be active. iPhone OS is a single task single process system. If we open the 'Maps' application and from there we click in a url, the application 'quits' and opens the Safari application. When the application quits, the data and the state of the program is preserved – if the programmer want's to.

Sandbox Every iPhone application has it's own sandbox. We cannot use other applications data, or even communicate, but within the sandbox we can do almost everything with files and directories. To access some iPhone OS features like system settings, contacts database, voice mail from the phone application or the photo library, we can use frameworks, such as the 'Contacts.framework' (gives access to the contact list) and the 'UIPicker-Controller' (gives access to the photo library and the camera). So, sandbox allow us to open the file system to a minimal set of folders, hardware and network resources.

> "It's like attending an overly restrictive school with a paranoid principal" [48]

Memory There are limits here. The iPhone OS is a version of the Unix operating system. But doesn't use swap, so it has a limited memory. If we alloc to much memory or the limit is almost full an notification and a warning will pop-up and if we cross that 'line' the application will be terminated.

The iPhone OS can be viewed as a set of Layers, explored and used by his the development platform SDK.

3.3 The Software Development Kit

The SDK is Apple's object-oriented set of frameworks that help developers write the iPhone applications.

Over the following section we'll introduce some of the basic features, how to use Objective-C and the iPhone OS, and how the tools work.

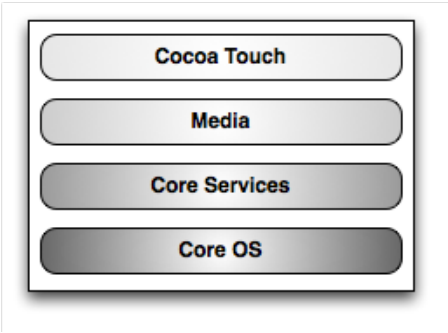


Figure 3.2: iPhone Software Development Kit Architecture

3.3.1 Tools

Before the iPhone SDK came along, Apple had several tools for Mac OS X and library development: The Xcode, Instruments and Dashcode.

Now, we have a complete set of tools for iPhone development [6]. These tools can be downloaded from Apple's developer Web site, called Apple Developer's Connection.

Xcode

The Xcode is the core of the SDK's environment, it's Apple's integrated development environment (IDE) for Mac OS X. The Xcode tools package presents a graphical workbench that tightly integrates all the following features: project management, editors, GCC compilation, logging, variable listing, debugging, optimization, documentation, inspectors, libraries , the links to the Foundation and other features. Xcode supports C, C++, Fortran, Objective-C, Objective-C++, Java, AppleScript, Python and Ruby source code with a variety of programming models, including but not limited to Cocoa [4], Cocoa Touch, Carbon, and Java.

Interface Builder

Commonly refered as the IB, the Interface Builder is the partner-in-crime to Xcode [39]. Is the tool that lets developers put together the graphical elements of the program. With Interface Builder we can create XIB files containing visual elements and their relations with the application user interface classes.

These files with the .xib extension appear at the top level of the application bundle are defined in the info.plist file [48]. The default one used in project is called 'MainWindow.xib'.

iPhone Simulator

A major part of the iPhone simulation environment is the iPhone Simulator application. This application presents the iPhone user interface in a window on your computer [14]. The application provides several ways of interacting

with it using your keyboard and mouse to simulate taps and device rotation. Allows to run the applications for testing and debugging before deploying them to the App Store. It provides the shape of the iPhone, Internet connectivity, multi-touch emulation, rotation, system preferences. Doesn't have GPS, camera, accelerometers and other sensors.

Instruments

Instrument is a program that allows developers to dynamically profile, trace and debug the application [2]. It's were we can monitor, for example, the performance and memory usage. With Web Applications we have to install browser addons, or find other Web sites, to do this sort of work, but, with Instruments that's all incorporated into this one package.

Dashcode

Dashcode it's a graphical development environment that allow the creation of Web-based applications or widgets programs. It makes it easier for a novice programmer to create Web applications with professional look. However, it has no use for native application development. The integrated workspace allows smoothly transitioning from design and layout, to editing code, to debugging JavaScript, and finally deploying the application to your server or computer.

3.3.2 Objective-C, the language

This language is used since 1984 when the Object Oriented Programming (OOP) where introduced in the C language. The base languages are C with classes (1979) and small-talk. Since then, it's taking a different path from C++ – which is fast but static [26]. The main differences are related with dynamic message dispatching, runtime decision making, runtime class extensions and, of course, notation.

Since Objective-C is build on Object Oriented (OO) architecture, the controllers, windows, views, button and sliders will be exchanging information with each other and responding to events. Here's an example in how we call/invoke a method:

```
[C++]          someObject->doSomething();
[Java]         someObject.doSomething();
[Objective-C]  [someObject doSomething];
```

Here's another example with arguments of the selected to call of a method.

```
[anEmployee setWork:work withDeadline:today];
```

The selectors are **setWork:** and **withDeadline:**. This way we have self documented code to help us understand what the method does.

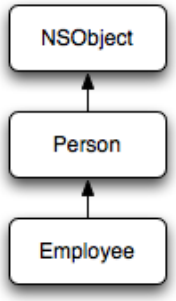


Figure 3.3: Objective-C is a single inheritance language

Objective-C is a single inheritance language but has distributive objects and message forwarding, which means, that runtime traverses class hierarchy to find a method that responds to the message.

There are some base classes for common behaviors like NSObject, and we can build subclasses like Person and sub-subclasses like Employee. The NS in NSObject stands for Next Step.

Objective-C has different ways of creating classes, methods, initialization, memory allocation and management (reference counting, retain/release), synthesize, interfaces and uses some features Java-alike such as: properties and protocols. Examples will be showed in the following subsections.

3.3.3 Cocoa Touch and Frameworks

Cocoa Touch is a collection of API and frameworks that we use in the iPhone applications and provides an abstraction layer of the iPhone OS. It has two pieces - Core Foundation and UIKit library.

Foundation

Foundation has the general object wrappers such as strings, collections, data structures and various system services like File I/O and networking. Foundation on iPhone is a subset of Foundation on the Mac so it's portable between systems.

- Strings. Example:

```
NSString *myString = @"Hello World";
```

- Arrays. There are two types of Arrays - mutable and immutable. The default one is the immutable. Example:

```
NSMutableArray *arr = [NSMutableArray array];
[arr addObject:@"randomString"];
```

- Dictionaries or Hash Tables. Like the arrays there are two types: mutable and immutable.
- File I/O

```
NSFileManager *fm = [NSFileManager defaultManager];
```

- Files and folders. An iPhone application is sand-boxed, which means that we can create everything. The system will backup/synchronize everything except the folders: tmp (cleared, not backed up) and Library/Caches (saved but not backed up);
- NSLog. Is used to view and debug the application process and state. Allow the programmer to view how the data looks and show the sequences, events and notifications;
- Foundation Bundles. An iPhone application is a set of files, it's a special folder containing the images, the executables, the Library, the Documents folder and other files. Foundation Bundles are also special read-only folders with resources like movies, Web files, metadata, etc. We can use this by invoking the NSBundle class;
- SQLlite is included in the iPhone OS and can be used with off-line and file-based databases;
- Networking. These classes like CFNetwork, CFSocket, NSXML and NSURL allow programmers to work with the POSIX protocols included in the iPhone OS. The Bonjour networking used to work with pears of devices is build on the Transfer Control Protocol/Internet Protocol (TCP/IP).

UIKit Library

It's the User Interface programming Kit. It's the heart of all GUI applications on the iPhone [54]. Provides standard interface elements. Basically, it handles everything that shows up on the screen and interacts with the user. It manages the event handling, the graphics and the windowing, the text and Web management. This is a important feature of the SDK, and if we don't fight it and understand the designs the can have hugh pace in development.

This framework includes buttons, controllers, data, image and Web views like the "UIWebView" that is used to view Web content inside the application instead of bringing-up the Safari application. Is based on the Model-View-Controller (figure 3.5) key design pattern (figure 3.4). Basically, it handles the reaction and how it reacts to the user actions. It has a object oriented approach for control (in terms of target and action) and selectors and objects.

The UIKit Framework is responsible for starting our application, and every one has a single instance of UIApplication. It is also responsible for orchestrating the life cycle of an application (figure 3.7), dispatch events, manage application icon badge and status bar.

```
@interface UIApplication
+ (UIApplication *) sharedApplication
@end
```

Notification

These are asynchronously delivered messages with information while an application is running. The iPhone OS has a Notification Centre to manage the messages and the observer/controller. It can be used while a video is being downloaded or to handle URLs.

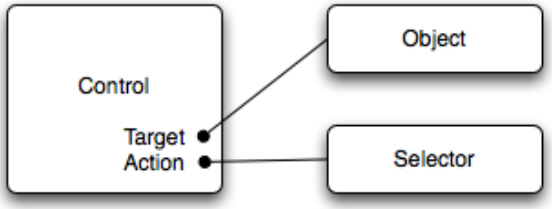


Figure 3.4: Application Design Pattern

Delegation

The iPhone OS is multi-threaded, which means, a lot is going on when launching, during and quitting an application – the Application Life Cycle (figure 3.7). For example, the UIApplication is used to load/prepare a saved state and to save data, screen resume information. It's control passed to delegate objects to perform application-specific behavior, and by using it we avoid the need to subclass complex objects. UIApplication, UITableView and UITextField are examples of classes that uses delegates.

Another example, the UIConfig is responsable for bringing up the application user interface and puts on the screen. It's a custom class, which means, has methods, and conforms to UIApplicationDelegate protocol.

```
- (void)applicationDidFinishLaunching:(UIApplication *)application;
```

Info.plist file

The Info.plist is a XML file containing the property list and some application details such as:

- Icon appearance;
- Status bar style (default, black, hidden);
- Orientation;
- Uses Wifi networking;

- System Requirements.

Model-View-Controller paradigm

This is an important concept. It's a philosophy behind the programming language and how they work. This is a architectural pattern of Objective-C and Oriented Object Programming [2]. Basically, it breaks a program in three parts, model, view and controller as you can see in figure 3.5.

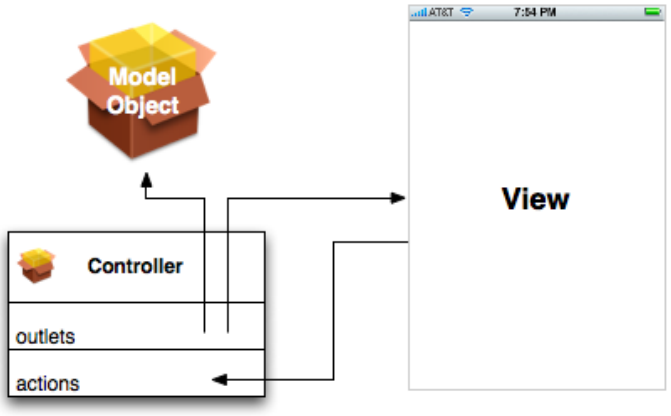


Figure 3.5: Model-View-Controller

Model Manages the application and the state. Is the heart of a program, but it doesn't have concerns with User Interface (UI) or presentation. Often persists somewhere, and, in some cases, is unchanged in different interfaces –but should be reusable.

View Presents the Model to the user, and with the controller they comprise the presentation layer of the application. Allows user to view things and manipulate data, but doesn't store any data (except the cache state). It can be configurable and reusable to display different data.

Controller The controller sits between the view and the model, i.e., it's a intermediary between the user interface and the model changes. Updates the view when the model changes and updates the model when the user manipulates the view.

3.4 Web Applications Development

Web development involves the making of Web pages that are optimized for the iPhone. They use standard technologies (such as Hyper Text Markup Language (HTML), Cascading Style Sheets[12] (CSS), JavaScript, and a server-side programming language like Hypertext Preprocessor (PHP) combined with development

[12]Web Design inspiration for the iPhone. http://cssiphone.com

kits provided by Apple or iPhone enthusiasts[13] (WebKit[14], iUI[15]), Canvas, and iPhone-specific tools like Dashcode. These Web pages run on the iPhone browser which is a mobile version of Apple's Safari.

Application development concerns the building of programs that runs natively on the iPhone. These programs use the technologies that we talk about in the above sections such as iPhone SDK and the Objective-C language.

There are advantages and disadvantages for each technology. The table 3.2 summarizes the main differences.

Web Development Advantages	Application Development Advantages
Ease of development	Complex development environment
Ease of user-access	Improved language depth
Fast deployment	Integration with iPhone frameworks
Ease updating	Access to graphics libraries
Access to dynamic data	Access to application sandbox
Offline server access	Flash Memory size
Access to other users	Native speed

Table 3.2: Some advantages for each model of development: Web Applications vs Native Application

In our work we've worked with both technologies. We will talk about that in the following chapters.

3.4.1 Designing a Web Application

In the first year of the iPhone's existence the only way we could do applications for the iPhone was thru *Web pages* tailored for the iPhone [46]. It was a "workaround" while Apple could come up with its App Store.

The advantage of developing a Web Application is that they don't occupy storage on the device (only Safari cache) and they are always updated, but, this can be a problem if you want to save data on the internal memory.

We can find these free mini-sites on the Internet. They look like desktop widgets, completely different from their main site. They are designed specifically for the iPhone characteristics and they cover all the know categories around. Some well know examples are: iPublico[16], iZoho iPhone Office[17], iActu[18], Mojits[19].

[13] iPhone Web Dev – Developers Helping Developers. http://www.iphonewebdev.com and http://groups.google.com/group/iphonewebdev
[14] WebKit. http://www.webkit.org
[15] iUI: iPhone User Interface Library. http://code.google.com/p/iui/
[16] www.publico.pt/i/
[17] http://mini.zoho.com
[18] www.iactu.mobi
[19] www.mojits.com/home

Good design practices

For developing Web Applications there's a few things we should care about in our Web[20] and Native[21] applications compiled in the following items:

- Separating HTML, CSS and JavaScript files so that each file can be cached by Safari;
- Creating valid and well-formed HTML or XHTML code. Requires less parsing and final display can be improved;
- Make size images appropriately. Don't rely on browser scaling and work for the iPhone viewport. Example, instead of using image borders, define CSS borders properties instead;
- Remove white space (tabs, spaces and transparent images) when possible;
- Remove useless tags, unnecessary styles and unused JavaScript functions;
- Create tile small images in background instead of big backgrounds;
- Clean up comments, code and use short filenames;
- Avoid complicated frame-sets;
- Divide code in specific blocks;
- Use media queries properly.

Standards

The iPhone uses Apple Webkit, which includes the following technologies, Web standards or well-formed documents:

- HTML 4.01 and HTML 5;
- XHTML 1.0;
- CSS 2.1 and some 3.x;
- Javascript 1.4;
- The Document Object Model – DOM;
- AJAX/Web 2.0;
- Web Hypertext Application Technology Working Group – WHATWG[22].

[20] W3C's Mobile Web Best Practices. http://www.w3.org/TR/mobile-bp/

[21] W3C's Device Description Repository API. http://www.w3.org/TR/DDR-Simple-API

[22] Web Hypertext Application Technology Working Group – WHATWG community. http://www.whatwg.org/

Limitations

There are some unsupported third-party technologies for the iPhone browser. The reasons for this has to do with battery life concerns and the download limitation policy of the iPhone Safari browser. But Apple's policies tend to change and is possible that the following list of unsupported technologies change with the release of future versions of the iPhone OS.

- Scalable Vector Graphics (SVG);
- Extensible Stylesheet Language Transformations (XSLT);
- Wireless Markup Language (WML) but XHTML mobile profiles documents do work at .mobi domains [2];
- Java;
- Adobe Flash.

The small screen of portable devices is the main limitation for actual Internet content. The lack of a mechanism for horizontal scrolling, slow and expensive downloads, limited memory, slow processors and battery, are other factors that make the users turn off in-line image loading.

From a design point of view, these limitations imply:

- Designing in one column and avoiding floats;
- HTML optimization by using efficient, semantic markup and CSS;
- Drop the decorative design images;
- Avoid the use of images, scripts or plug-ins for navigation;
- Set images alternative text and titles;
- Give special attention to navigation as the portable devices don't have mouse or keyboard.

Aside from these limitations, in recent history the number of full Web-capable mobile devices increased. The Apple iPhone with the Safari browser, the Google Android[23], Nokia S60, Opera Mobile and Opera Mini are examples of what technology have done to improve user experience and accessibility.

The zooming interface is an example allowing users to focus on a specific part of a Webpage [35]. Although, there's a downside, for non-familiar users with the site the viewport switching can be confusing.

[23] Android Developers. http://developer.android.com/

3.4.2 Web design and data delivery concerns

From the many Websites in the Internet, only a few percentage are standards-compliant. Among those few, the ones with style sheets adjusted to portable devices are less than the ones with printable versions. Even those which offer styling for portable devices, don't present the page without the horizontal scrolling. [29]

Although mobile browser are becoming more powerful, they still create usability challenges. Trying to solve those issues we provide a better experience to mobile users by using handheld style sheets and CSS media queries.

In fact, the World Wide Web Consortium[24] (W3C) has it's own recommendation – Mobile Web Best Practices[25] – released in July 2008.

Normally, when people goes from a layout on a computer's browser to the mobile version they loose information in the process. The best case scenario is to develop a mobile version that's not much different from their screen versions and maintaining the fundamental nature of a site's navigation.

We intend to distribute a mobile Web service by making a portable device version of the same service we've being using on the previous site. It's the same technique used by Google with it's mobile version of the site in the url m.google.com.

3.4.3 Web design for mobile devices

Here's some considerations that we have thought to build a Web Service for portable devices:

- Scaling down to smaller screens. We've avoided using pixels larger than 5px, instead, we use ems or percentages for larger sizes;
- The margins and padding for elements have been shortened. The borders have been narrowed and deleted redundant spacing;
- The titles and larger type are not bigger than twice the size of a paragraph text, and the small bits of text where close to paragraph size of the desktop;
- The text blocks are as wide as possible. "As percentages scale with the width of the display, specifying a margin in percentages instead of EMs will work well on a wide variety of screen sizes" [29];
- Loose the background-image property as they tend to be pretty big and have limited use in portable devices;
- One column design structure to ensure that the page organization makes sense. This technique has been used with non-CSS browsers, voice browsers and in terminal-window browsers like Lynx;

[24] World Wide Web Consortium. http://www.w3.org
[25] W3C?s Mobile Web Best Practices. http://www.w3.org/TR/mobile-bp/

- The header is build only with a logo and one or two small navigational elements. Leaving the complete navigational support after the main text;
- Special attention to wider objects like images or form controls by assigning the max-width to 100% or 280px;
- Eliminating the frames, pop-ups and iframe technology. Avoid the float and display CSS rules because they are non-mobile friendly properties to browsers that only read screen style sheet without CSS media queries;
- Accessibility issues with navigation. The lack of keyboard or mouse must be overwhelmed by direct on-click-design features;
- In mobile phones every kilobyte of transfer is paid off, so, to save time and money, the images were scaled down and Internet optimized for portable devices. The use of width and height attributes are necessary so that the browser pre-formats the page while downloading the image. The existence of alternative text is also necessary while the browser processes the image and if the image is purely ornamental it has the empty string;
- Use of efficient HTML and compacted CSS by removing all the unnecessary elements and choosing the right elements to structure the content.

3.4.4 Cascading Style Sheets

The way portable browsers display Web content has changed in history. Since 1998, the W3C HTML 4 specification[26] offered ways to link to different style sheets depending on the devices targeted, including handheld media. The reaction to CSS Media Types [35] lead us from the:

- "Reading only handheld style sheets" with browsers like: OpenWave browser, Nokia lite-Web browsers, Netfront (configuration dependent), Digia, BlackBerry browser[27], Opera Mini until v4, Opera Mobile until v9;
- "Reading both handheld and screen style sheets", in this area we have: Palm's Blazer, Nokia S40 browser, IEMobile 6.x and 8.x;
- "Reading only screen style sheets with Media Query support", those are recent browsers like: iPhone's Safari, Opera Mobile starting v9, Opera Mini starting v4;
- "Reading only screen style sheets without Media Query Support" are browsers that intend to view the Web as screen. Examples: Nokia S60 browser, Netfront (configuration dependent), Teleqa Q7, IEMobile 7.x.

[26] HTML 4 Specification.
http://www.w3.org/TR/1998/REC-html40-19980424/types.html#h-6.13

[27] BlackBerry Developer Zone. http://na.blackberry.com/eng/developers/

Mobile browsers that only read the handheld style sheet will never see the potentially harmful CSS properties defined in screen.css. Mobile browsers that read screen style sheets, and handheld or media queries style sheets will not be affected by the harmful properties in screen.css, since they're canceled by antiscreen.css. Finally, PC browsers will happily ignore both antiscreen.css and handheld.css.

The Safari for the iPhone supports all CSS functionalities: styling, animations, transitions and transforms. By turning on the Debug Console, which is an information strip at the top of the Safari screen [46], the user can view errors, warnings, tips, and logs for CSS, HTML and JavaScript.

3.4.5 Viewport

The browser viewport is the area in the browser through which the document content is displayed. This is not the same as the Initial Containing Block[28]. It's also a HTML meta tag used to define the viewing area on the screen and is used by a extensive list of user-agents such as Spiders, Robots, Crawler and Browsers[29] to change the document's layout when the viewport is resized. Of course, user agents may render to provide different views of the same document.

Because the Safari Mobile viewport default value is set to 980-pixel-wide, the use of the *viewport* becomes extremely important in order to correctly displays the Web application to the user.

Most of the mobile versions currently available doesn't have this feature, therefor, Safari doesn't display those Web page with the width correctly. Users need to zoom or panning to view them correctly.

3.5 Native Applications

Native Applications do what Web Applications do and more. Therefor, a native application is the best scenario for iPhone users.

In terms of components of an application, there are four distinct parts. In first place, there is the compiled code, the developer code and the frameworks of the SDK. Then we have the Nib files, which contains the UI elements and other objects (the design), and, details about how objects relate each other. We also have the resource files (images, sounds, string, an more), and, finally, the Info.plist file with details about application configuration.

3.5.1 Typical development cycle vs iPhone development cycle

Developing for the iPhone is different from developing for other platforms. Apple recommends developers to embrace the design paradigms seriously. The innovation is in 'design', and, as we already saw, 'design' is more than the looks of the 'thing', for example, seamless user experience and responsiveness. The following

[28] Definition of "Containing Block".
http://www.w3.org/TR/CSS2/visudet.html#containing-block-details

[29] List of User-Agents: http://www.user-agents.org/.

figure 3.6 shows how the time is distributed thru the development phases for typical platform versus iPhone.

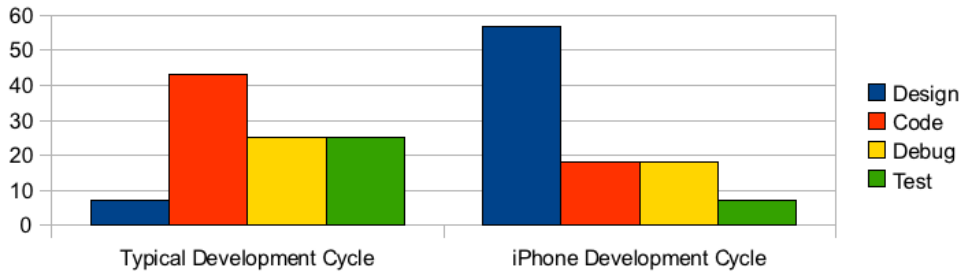


Figure 3.6: Typical vs iPhone Development Cycle

In our case, this is not far from the truth. For the user interface design we've done some homework before we start coding. We've downloaded all the news, media applications available in the App Store and perform a comparison in several aspects. That took some time, but, in the end, we've come up with some sketches for our application that gave us a good image of what we want (see figure 5.18). We can see it later in the chapter 5.

3.5.2 Application life cycle

The Application life cycle is a sequence of events that occurs while the application is active or loaded. Starts after the user launches the application by tapping its icon and end by clicking on the Home button, see figure 3.7. In the time between the application runs: the main() and the UIApplicationMain() functions (loads the main nib file and stops when 'applicationDidFinishedLaunching:' method is over); loops waiting for an event; when the system evokes the 'applicationWillTerminate:' method the application execution terminates. [13]

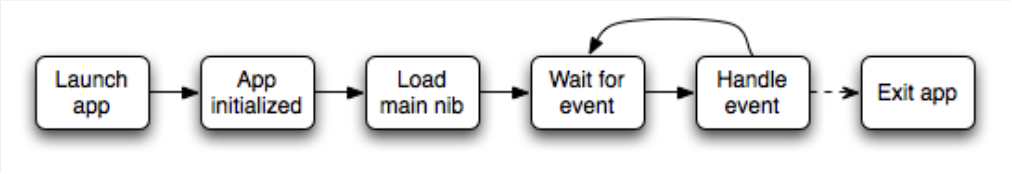


Figure 3.7: Application Lifecycle

3.6 Web Applications vs Native Applications

The development differences between a Web Application optimized for the iPhone and a 'Native' one are huge. In fact, apart from some similar tools in the SDK, there's nothing in common, see table 3.3. Some say, that Native Applications

doesn't brought anything new to the iPhone OS [3], that everything you can do with native applications it can be done with Web Applications too.

It's not only what you can or cannot do. The speed of communications and processing is also important. The wireless communications tends to drain out the battery also. Not to mention that, in some locations, we can't access networking and when that happens the Web Applications cease functioning.

As for the price and business model the differences tends to get bigger. Native Applications are sold and distributed thru the App Store. They can be downloaded directly to the iPhone or using a iTunes desktop version. Yes, Apple keeps 30% of the price for herself. That's a fee for keeping the "store clean", for transaction fees, servers, update support, quality control, and keeping costumers happy. And there is always the software license agreement with Apple and users. Yes, we didn't forgot: It's always possible to jailbreak your iPhone in order to install more applications, or applications that Apple doesn't want to sell on App Store. And you have to pay the annual subscription (a minimum of $99) to the iPhone Developer Program to deploy your applications on iTunes.

	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
Instalation	Just online access	Download or sync
'Findability'	URL/Internet	iTunes/Internet
Price	Free/Web Registration	Free/iTunes
Access	Thru Safari framework	App Store Download
Offline usage	No	Yes
User experience	Limited	Direct access

Table 3.3: Differences between Web Applications and Native Application

Web Applications doesn't have a repository store, with all the existing apps out there, for you to find them. The advantages of the App Store could be disseminated if Apple come out with a way to store Webapps localy so that we wouldn't be tied to the cloud. There's another drawback, Web Applications rely on the Safari. Safari, like any other piece of software, as it own flaws, bugs, updates, memory leaks and limitations. In future an upgrade to the Safari App could jeopardize our application.

More, to build Web Applications, like those we will talk about in future chapters, you have to choose a base framework, and that could be a big limitation to the applications itself. There's also the tools for developing. For example, debugging Native Applications with the Xcode is harder the tools and resources available for Web developers. The language itself is not limited to Objective-C or Object Oriented Programming, and this can be a major advantage for the

Web Applications. The advantage is, you always have the latest version of your Web Application available. No need to connect to iTunes and download newer versions.

For the final user, if he has a good Internet connectivity, the differences maybe not so huge. As we already saw, some of the devices features are available thru Safari for iPhone OS. If there is no wireless connectivity the Safari browser is almost worthless because you can't access Web sites. On the other hand, we can only access Native applications thru iTunes, and that can be a little tricky compared to the user friendly URL entry in the Safari.

There's also the payment procedure, to buy an application is easy with the iTunes, with Web Applications you have to insert you credit card information every time you want to purchase an item, service or payed access.

Despite the differences, all we know is that the two platforms have it own advantages and they can coexist peacefully in the mobile devices world.

Chapter 4

The Web Applications

In this chapter we will describe how we develop optimized versions of the *Urbi et Orbi* – Online University Journal –, provided by the LabCom and the students of Journalism. We use several technologies such as the Apple Web Kit Development framework, iUI and Dashcode with UIKit. We will talk about the process, the goals and needs.

Web Applications that perfectly fits in most of the mobile devices[1] available today. We will also analyze it's features, specifications and some specific iPhone Web Applications.

Urbi et Orbi

The first edition was in 1999. The *Urbi et Orbi* started to be a simple structured and static Web page, formated to be as a news repository. In 2006, received a major upgrade by implementing database capabilities, and, a back-office for content management that makes the site a dynamic one. Now, it has front-end features like: RSS, Mailing-list, Send-to a friend, You-report, Commentaries, Real-time statistics, and, on top of this, mobile versions.

The mobile versions started to be fixed width, simplified version of the main screen one (desktop). Despite it's specifications, the suggested browser client was the mobile Internet explorer. It was striped from database dynamic connectivity, i.e., it was a modified 'site dump'. No forms for users interaction, no reports, and no multimedia content whatsoever. Design was poor and difficult to implement doe to several different browser versions.

Late in 2008, we developed the *Urbi et Orbi* – Mobile/iPhone Version[2]. This version was created and tuned-up to be more than a striped version of the main one. In 2009, we built more mobile versions to test the several frameworks available for iPhone Web Developers.

In figure 4.1(a) we can see the front-page of the desktop version and on the right (figure 4.1(b)) the Mobile/iPhone version developed with the iWebKit[3],

[1] Some versions doesn't support other mobile devices.

[2] *Urbi et Orbi* – Mobile/iPhone Version. http://www.urbi.ubi.pt/i/

[3] iWebKit. http://www.iwebkit.net

viewed with an iPhone on landscape mode.

(a) *Urbi et Orbi* – 'Desktop' version

(b) Mobile/iPhone Version – Landscape mode

Figure 4.1: *Urbi et Orbi* – Web platform screenshots

4.1 Introducing the Web Applications for *Urbi et Orbi*

We developed four different Web Applications with three distinct frameworks. There are three 'news edition' with pictures, complete body info (title, date, publisher, super-lead, picture and text), divided by categories and sections. The other one is a RSS readers. It's a list of entries with title, published date, and a short brief text (super-lead).

The frameworks we've choosed are iWebKit, UIKit and the Chris Apers (Chrilith) WebApp.Net[4] framework and plugins for open-source Content Management Systems (CMS) like WordPress[5] (WPTouch).

[4]WebApp.Net – Chris Apers (Chrilith) © 2008. http://webapp-net.com
[5]Wordpress for iPhone. Available in App Store. http://iphone.wordpress.org/development/

4.1.1 The Mobile/iPhone version – iWebKit

This is a all-around mobile Web Application. Initially developed as an iPhone Web Application, soon we have realized that it works in all major mobile browsers. So, it turned out to be the all-around mobile version now because the accessibility in other devices is good. Of course, it's best viewed with Apple Safari as it uses a simplified version of the WebKit. Contains two additional iPhone features for viewing news through images: '*Urbi* in thumbnails' and '*Urbi* on images' to use the touch control with slide effect to navigate on the pages.

Sub-Site structure

This site was designed for ease to use. It has a three level structure with item/detail (left/right) organization basis. It's a simple and straightforward version for mobile users. It's guided by the UI rules that Apple uses for Web Applications: The most frequently read on top, provides a well organized workflow divided by sections, it's a clean and uncluttered layout for screens, gets user to primary content as quickly as possible, minimizes zooming and panning, and has the major CSS Transitions from Apple Webkit build for Safari on the iPhone.

The figure 4.2 shows how the site is structured. The A + A' parts are the front-page of the week edition. This part is organized in Latest, Categories, Sections, and Others (*Urbi* on Images and *Urbi* on Miniatures).

Part 1 is the detail page of the news, it can be access from the parts 1, 2, 3 and 4. Figure 4.1(b) is also this item detail view but in the landscape mode. Contains all information regarded the new item – title, *super-lead*, picture, *corpus*, journalists, and published date. From here you can access more pictures, other detailed pages, other sites, video and audio of the topic.

Part 2 is an example of the item list, in this case it's from the category 'UBI'. The list contains the news topics ordered by publishing date. Each item is linked to a detail page 1.

Part 3 is the image version of the week edition's. It has a feature for the iPhone, which is the horizontal navigation through *Flick*[6]. The detail view of the new is also available if you click in the photo. The other mobile browsers can access the items through the navigational links in the bottom of this page.

Part 4 is a Photo Library with all the images from the news edition. In the bottom of the page we can find an AJAX button that retrieves the image collection of previous editions. If we click on an image item we can access the detail view 1.

[6]Flick is how you advance to the next picture in the batch. Flick from right to left for the next (Flick from left to right to view the previous photo).

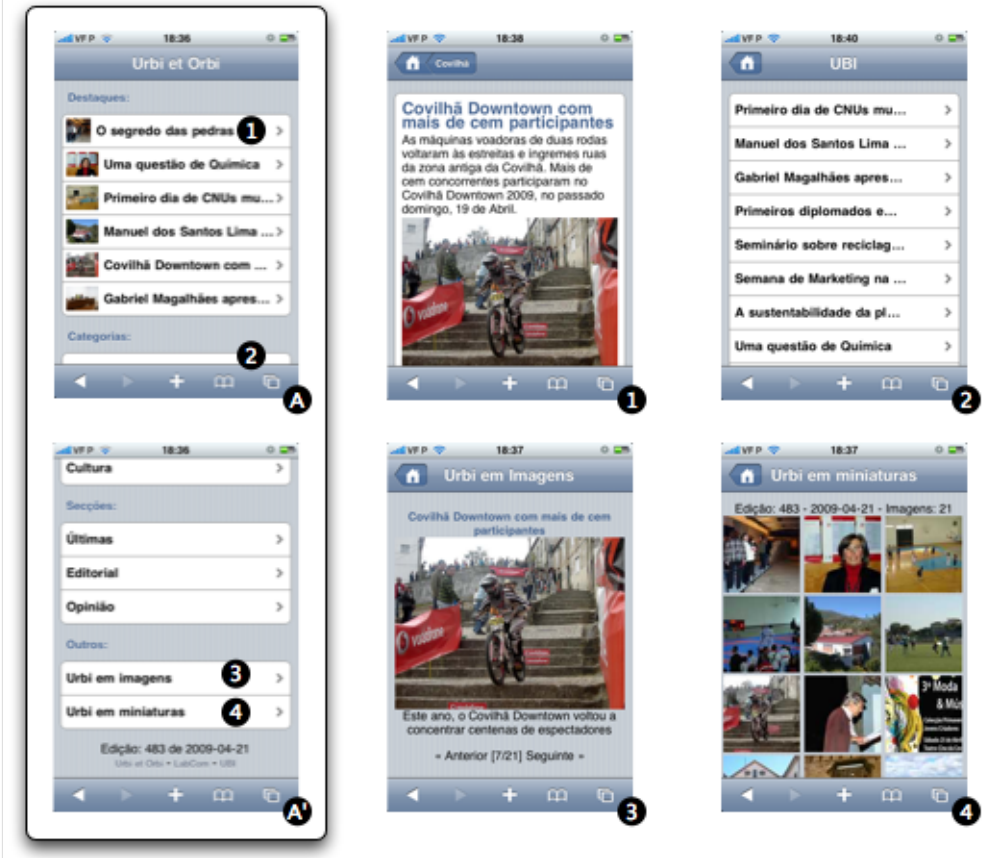


Figure 4.2: *Urbi et Orbi* – Mobile/iPhone version site structure screenshots

4.1.2 The News version – Dashcode/UIKit

This is a items list and information viewer for the latest news of our Journal. The item number depends on the edition entries.

It was built with Apple Dashcode template and uses the UIKit framework making it an iPhone platform exclusive.

The major modification to the template is the inclusion of JavaScript functions to make the content delivered through an XMLHttpRequest. This asynchronous request is made while the browser renders the page saving precious time of page loading. Our modifications in this template includes other tweaks like specific page for the version, buttons, statistics, design and content changes.

This version loads, on the first access, all the items details and relative pictures. The advantages are:

1. The detail information appears a lot faster because it's already on the Safari's memory;

2. The page is cached in the Safari and all the items and details can be viewed later – even without Internet communication.

Sub-Site structure

This application is also designed for ease to use. The site structure contains one major item list (figure 4.3(a)) to display with all the item information (figure 4.3(b)), a page detail for each item and a general version information with buttons to external links.

The first page (figure 4.3(a)) differs from the previous version (Mobile/iPhone) because the item images are bigger and with more quality. The item information include category, date, and *superlead* preview.

The detail page (figure 4.3(b)) is similar to the previous version. Doesn't include links to other pages of the same category (as we can find on Mobile/iPhone version), but, includes a button linking the viewer to the Desktop version and has more information than the Edition version.

Note that the figure 4.3 screen capture doesn't have the top and bottom navigation bars. This is an iPhone feature that makes possible to create an icon on the applications panel by clicking the plus button on the Safari for iPhone. See figure 4.4.

When the Web Applications (News and RSS) are accessed through this icon, the JavaScript included in the application removes the URL top bar and the navigation bottom bar of Safari. This makes our Web Applications almost similar to an Native iPhone Application. An easy access and personalized icon; an almost full-screen display design application with all the advantages of having a few more pixels to display content; speed; and memory because with this feature Safari doesn't have to share resources with other Web pages.

(a) News – First Page (b) News – Detail Page

Figure 4.3: *Urbi et Orbi* – News version screenshots

Figure 4.4: Our Web Applications versions icons "installed" on the iPhone applications panel. From left to right: Mobile/iPhone version (iWebKit); News version (Dashcode/UIKit); RSS version (Dashcode/UIKit); Edition version (WebApp.Net)

4.1.3 The Really Simple Syndication version – Dashcode/UIKit

This is a all-in-one-single-file-one-communication approach. The framework was developed by Apple.

It's a Really Simple Syndication (RSS) reader for the iPhone platform. This Web Application gets a RSS feed (asynchronously) from our server and deploys it in this beautiful and fast interface. We configure some buttons to allow users to read the full article in the Desktop version. Other modifications to the template includes JavaScript tweaks for time statistics, date format, item numbers changes, and design aspects.

Sub-Site structure

This application is a two page (two levels) design structured information. One for the item list and the other for the detailed information.

The first page (figure 4.5(a)) is divided in two sections, separated by commas, with the *superlead* information preview in the first section.

The detail page (figure 4.5(b)) is a simplified version of the previous versions. The information is restricted to the title, date, and *superlead*. The button to the item list and the Desktop version (similar to the News version) are build in the bottom of the page.

(a) RSS – First Page (b) RSS – Detail Page

Figure 4.5: *Urbi et Orbi* – RSS version screenshots

4.1.4 The Edition version – WebApp.Net

This is a Full Editions Web Site using the WebApp.Net framework. WebApp.Net is an open source Web application framework by Chris Apers designed to mimic the actual iPhone graphic UI.

All the content of this application is dynamically loaded through AJAX requests. Despite the XML structure and content in the returned file, cannot be considered a Web service because the information is formated specifically for this application. This means that the content depends on the request and it can be a item list or a detail information.

The framework is based on JavaScript, CSS and images. It's features include:

- Automatic handled navigation and header title;
- Compatibility with browser navigation buttons;
- Easy to use and fully integrated AJAX technology usable also with forms;
- Easy quicktime media integration;
- Custom form elements similar to those found in the Xcode library;
- Custom events to catch: slide effects, rotation, AJAX errors, and so on;
- Advanced CSS for common iPhone elements;
- Search engine compliance.

This version differs from the Mobile and News/RSS versions because it combines the access technology used in those versions. From the Mobile/iPhone version we have the first page design with just a glimpse of the details and a small image. But, when we click for the detail view a XMLHttpRequest is activated to perform a asynchronous request, as used in News/RSS versions. The Mobile version requests a new full page. We will compare this differences later in this chapter.

Sub-Site structure

This version is more complex then the previous two version (News and RSS) because it's a union of those two applications.

It has a one plus four page on a three level structure. The first page contains the item list, a item link to the weekly editorial and a button for the latest entries list.

The first page (figure 4.6(a)) is similar to the Mobile version with bigger item pictures and more items. The references to the edition number comes in the top, contrary to the Mobile version. There's also a search button on the top bar to access the search form.

The detail page (figure 4.6(b)) is similar to previous versions. Includes title, date, *superlead*, picture and body. This detail page is also used in the editorial item view.

The latest news list (figure 4.6(c)) just display the five latest items, but it has a button to retrieve more items. This is an AJAX button so the information retrieved is dynamically loaded on this page.

The latest news detail page (figure 4.6(d)) is similar to the equivalent page in the RSS version. It has the title, the *superlead*, the date and, more importantly, a Home button on the top the let user go back to level one of the application. This is the implementation of the design suggestions that we talk about in Chapter 3.

(a) Edition – First Page

(b) Edition – New Detail Page

(c) Edition – Latest Items Page

(d) Edition – Latest Detail Page

Figure 4.6: *Urbi et Orbi* – Edition version screenshots

4.2 Web Applications Size and Performance

We created a testbed for speed and size performance evaluation. The procedure consisted in loading the Web Application from different systems and platforms. We use a iPhone 3G connected through Wi-Fi; The iPhone Simulator (Mac OS Leopard) and Safari Browser (Microsoft Windows XP and Mac OS Leopard) were connected through cable network at 100 Mbps.

The measures were calculated with different methods listed in the following subsections.

4.2.1 Inspector Web results

For these tests we have a JavaScript time procedure to know the loading time of our Web Applications. It is included in the 'main.js' file, and, in some cases in the end of the parsing methods. Basically, it's the difference between a first time stamp created in the header HTML file and one other after populating the HTML fields with the database info. It's like the "ready time". Measures the time difference between the load of the page and the unload of the previous page. The events for this are 'window.onload' and 'window.onbeforeunload'.

We also use the Inspector Web from Safari (Mac) that allow us to know precisely the loading times and sizes of each element of the Web Application. The Safari cache was cleaned before each run.

The results are presented in the following table 4.1. The speed and size are average numbers resulted from several tests at different hours of May 26, 2009.

The numbers in brackets are intervals to represent the minimum and maximum value obtained in the tests. Although the numbers are low, the maximum values hapenned because it was the first time that the server produced the weekly edition. This means that the server had to create the thumbnails in different quality and sizes for each of the versions. It can be considered as the maximum time because in the previous test we had not such poor results.

Also note that, all the versions includes a 'ga.js' file, for the Google Analytics, that comes from google-analytics.com, that varies some of the speed times.

Results analysis

The Mobile version is the smallest version, but the time-to-complete-speed values are bigger because the image 'menutouched.png' is the last file requested as the page is rendered, despite we have a *preload_image* definition for this image. If this was not the case, this version would be the fastest one to download as we can see in the JavaScript runtime in table 4.1, and, in the download graphs of figure 4.7(a).

The differences between Web Applications and their model can be viewed in the Detail Page part of the table 4.1. The Mobile version is sub-page independent, therefor, the request is even bigger than the first page. News and RSS versions just have to refresh design images to put up the detail page. The Edition version request all the item detail data – information and images.

	Mobile	News	RSS	Edition
First page access				
Size	59 KB	368 KB	150 KB	90 KB
Items requested	20	51	18	24
Speed	[1.8;2.4]	[0.7;4.2]	[0.4;0.6]	[0.7;1.9]
JavaScript runtime	0.02s	[0.6;4.1]	0.35s	0.05s
Detail page				
Size	81 KB	20 KB	5 KB	23.5 KB
Speed	0.9s	0.2s	0.1s	0.2s

Table 4.1: Size and Performance of the Web Applications measured with Inspector Web tool from Safari 4.0 beta on Mac OS Leopard and Microsoft Windows XP Professional. May 26, 2009

(a) Mobile/iPhone Version

(b) News Version

(c) RSS Version

(d) Edition Version

Figure 4.7: Safari's Inspector Web partial screenshots results. May 26, 2009

The graphics shows us the weight per file type. We notice that the Scripts (in Orange) are a big percentage of the application weight – more than 75% in the 4.7(c) version.

As expected, the News version has an big percentage of pictures downloads because it downloads all the available images of the news.

The figures 4.7(b) and 4.7(c) shows us (grey area) the XMLHttpRequest's included in the first page application. In the Edition version this request is just included in detail pages, not in the first page.

We can also see the Safari Browser (and possibly the Safari Mobile) request

file priority: The HTML file is first (obviously); than the CSS files; then the images included in the HTML and JavaScript files at the same time; then the CSS images (backgrounds and buttons); the Other files (XMLHttpRequest) in the end.

4.2.2 Sitesuker size and item numbers

The numbers in table 4.2 were obtained with the SiteSucker v2.2 software for Mac[7]. It's an application that automatically downloads Web sites from the Internet. It does this by asynchronously copying the site's Web pages, images, backgrounds, movies, and other files to your local hard drive.

Although SiteSucker was configured to be viewed with the identity of Safari 2.0, his inability understand the UIKit JavaScript and perform the XMLHttpRequest's doesn't gave us the optimal result.

Again, this proves that the Mobile version is the all-browsable and platform capable version. As a comparison, we also saved the first page with Safari 4.0 beta for Mac, and, as we can see in table 4.2 the size doesn't even match. We can also compare the size of the Safari Webarchive with the First page access size from table 4.1. They are greater because the Webarchive is the size on disk, which are bigger due to the file system clustering size.

SiteSuker	Mobile	News*	RSS*	Edition*
Levels	7	3	3	3
Items	289	24	17	86
First page size	2015 KB	183 KB	148 KB	149 KB
Safari Webarchive First page size	67.7 KB	351.6 KB	164.1 KB	106.9 KB

Table 4.2: Sitesuker 2.2 and Safari Download Statistics. May 26, 2009

* Sitesuker doesn't retrieve and count the XMLHttpRequest, therefor, the detail info is missing and so are the relative images.

4.2.3 YSlow for Firebug (Firefox plugin)

We also use the YSlow[8] for Firebug plug-in of the Firefox browser that evaluates the Web Application in 22 topics filtered by Content (6 items), Cookies (2 items), CSS files (6 items), JavaScript (4 items) and Server (5 items). The results are showed in table 4.3 and table 4.4.

This test is an excellent indicator of the optimization we come up with the Web Applications. The YSlow analyzed the first page and one single click (detail view) of the application. For each page returned an overall grade and a evaluation with a 5 point scale of each 22 aspects. The following table gives us the resume of the evaluation.

[7] SiteSucker online: http://www.sitesucker.us

[8] YSlow: https://addons.mozilla.org/en-US/firefox/addon/5369

	Mobile	News	RSS	Edition
Overall Grade	A	C	B	B
Overall performance score	91%	79%	82%	86%
Total weight	25.9 KB	45.6 KB	39.2 KB	49.8 KB
HTTP Requests	16	21	17	31
Communication times	973ms	756ms	220ms	715ms
22 items evaluation				
Make fewer HTTP request	A	E	D	A
Use a Content Delivery Network (CDN)	F	F	F	F
Add expire headers	A	A	A	A
Compress with gzip	A	A	A	A
Put CSS at top	A	A	A	A
Put JavaScript at bottom	A	F	F	A
Avoid CSS expressions	A	A	A	A
Make CSS/JavaScript external	N/A	N/A	N/A	N/A
Reduce DNS lookups	A	A	A	A
Minify JavaScript and CSS	A	B	A	A
Avoid URL redirects	A	A	A	A
Remove duplicates	A	A	A	A
Configure Entity Tags (ETags)	F	F	F	F
Make AJAX cacheable	A	A	A	A
Use GET for AJAX requests	A	A	A	A
Reduce DOM elements	A	A	A	A
Avoid HTTP 404 (Not found) error	A	A	A	A
Reduce cookie size	A	A	A	A
Use cookie-free domains	A	F	F	F
Avoid AlphaImageLoader	A	A	A	A
Do not scale images	A	A	A	A
Make favicons small	A	A	A	A

Table 4.3: YSlow for Firebug (Firefox plugin) evaluation of the Web Applications on May 26, 2009

Note that, all of them scores high in the evaluation. We even could perform better, if we had one more domain server to provide the images and cookie-free domains. The ETags topic is not working despite our Apache configuration on the server.

We can see that Dashcode/UIKit failed to put JavaScript at the bottom and minifying JavaScript and CSS. Of course, we can override this the changing the primary HTML file and putting ourselves the JavaScript at the bottom, and, removing the unnecessary comment of the JavaScript and CSS files.

The Total weight values are different from those presented in the table 4.1 because this plug-in could not display correctly the version News and RSS (built by Dashcode/UIKit). This means that, YSlow doesn't count all the images on the News version included in the XMLHttpRequest. Nevertheless, we can evaluate the HTTP request per file basis, see figure 4.8.

The figure 4.8 shows us the number of JavaScript files in the application made with Dashcode/UIKit. The files were included with the Template offered by Dashcode. It's possible to reduce the amount of files by merging and minifying

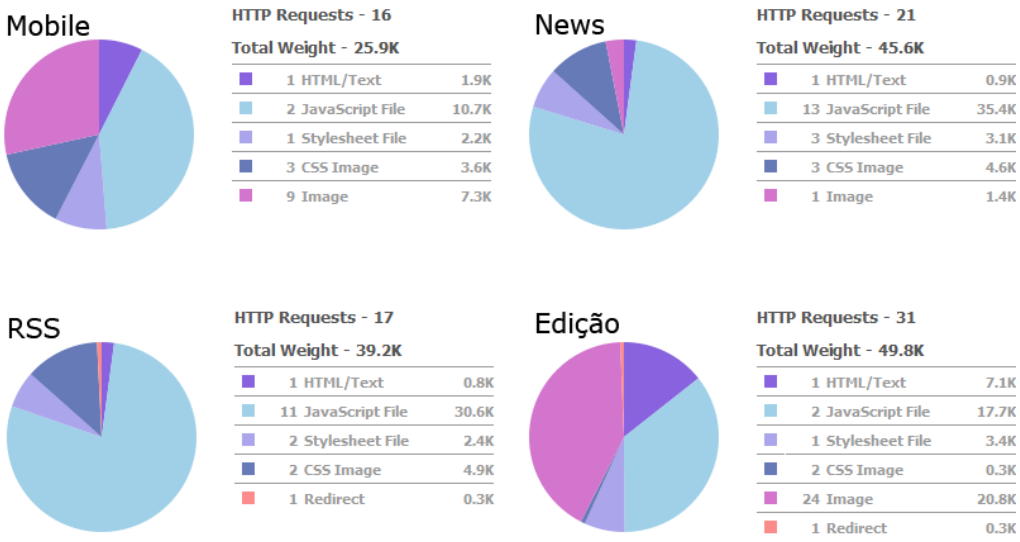


Figure 4.8: HTTP Requests per file distribution in all the Web Applications. May 26, 2009

the files, and, therefor, the application would perform better in communications.

4.2.4 Performance tweeks

If we run the tests with just seventeen items to evaluate by changing the rulesets to 'Small Site or Blog' (and configuring it to skip the rules: Use a CDN; Make AJAX cacheable; Configure ETags; Cookie-free domains; Reduce cookie size), the result shows excellent values for all the Web Applications, see table 4.4.

	Mobile	News	RSS	Edition
Overall Grade	A	B	A	A
Overall performance score	100%	89%	92%	99%

Table 4.4: YSlow for Firebug (Firefox plugin) with rulesets 'Small Site or Blog'

4.3 Functional Models

Our model is based on an standard Web-based client/server architecture. We have used Web standards and simplified models to design our architecture to improve portability and scalability. The major benefit to designing with Web standards is "design once, publish everywhere". [35]

4.3.1 Server

On the server-side, the Web Application is based on the Linux, Apache, PostgreSQL, PHP (LAPP) architecture. We have specific classes and objects to handle the service. The process consists in [1] collecting the filtered information, [2] apply the templates to the data, and [3] add the specific design and functionality for the client. In the server there's also other files, required for this model, such as, images and multimedia, and for other services.

The difference is it's phase [2] and [3] of figure 4.9, were we optimize the service for mobile devices by ubiquitously adjusting and adding specific features.

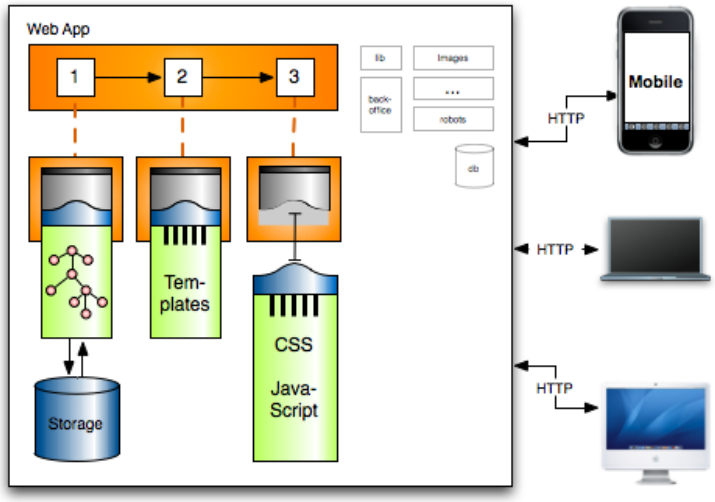


Figure 4.9: Web Application Model diagram

4.3.2 Database, Files and Images

Our system uses the information created for the Desktop platform of the online newspaper. The information comes from the *Urbi* database. The images are ubiquitously optimized for our mobile Web Applications. The information is also the same and is gathered from the database.

The figure 4.10 represents an simplified version of the database that we use to support all the Journal infrastructure. We only use 3 of the tables to create our mobile devices versions and the Web service also.

The attribute *foto* contains the image file name related to the new item. These files are stored in the *media* folder on the server. All the thumbnails created for the mobile versions are stored in a sub-folder of *media* folder called *miniaturas*.

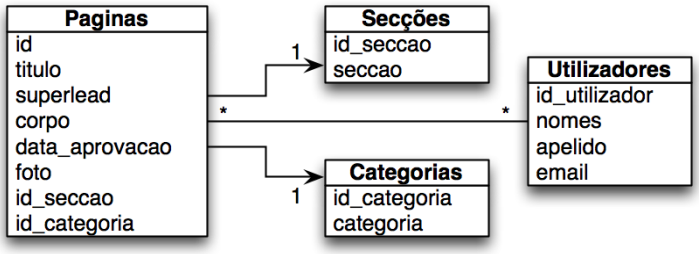


Figure 4.10: Relational Database Model of the *Urbi et Orbi*

4.3.3 Web Service

When delivering Web content to portable devices we must know the existing solutions to 'make the most' of the factors such as screen size, markup language support and image format support. In [55], the conclusions were that the cost of transmitting over the protocol TCP/IP was smaller compared to the wrapping/unwrapping of SOAP messages envelope, i.e., costs less to make successive calls than a single big one. Our applications uses the two versions of this paradigm.

The Mobile version doesn't use SOAP and it's a single page structure, while the Edition version uses it all the way. The differences between them, in terms of requests and speed, are presented in table 4.1 and in figure 4.8.

Our Web service is capable of variable content in the following variables: number of items; image quality, image width, categories/sections. We will talk about it in the following Chapter. The Web Applications are not posting device information to the Web service, so, the information retrieved is the default one.

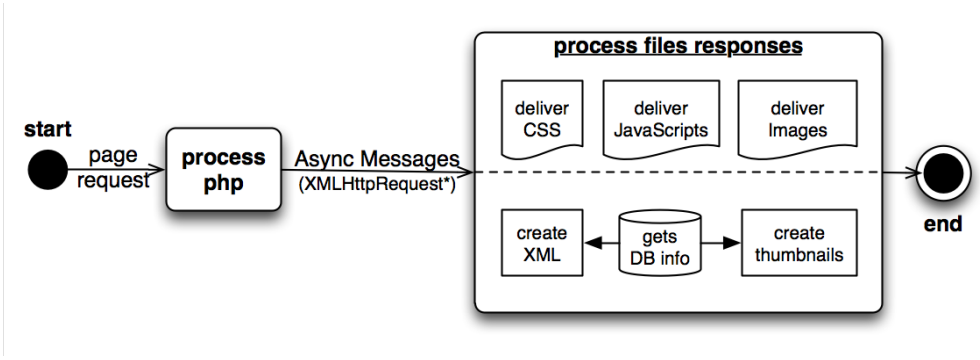


Figure 4.11: UML State diagram for the Web Server

Note that, the three versions that use XMLHttpRequest are capable of sending POST information, and this can be useful if we want to add ubiquity features to our Web Applications. The UML State diagram for the Web Server is presented in figure 4.11. It's useful because we can see when and how the Web Service works to process the XML file response and improves machine-to-machine

interaction over a network.

Our Web service is not listed in the platform-independent XML-based registry Universal Description, Discovery and Integration (UDDI).

The XML files

The Web service URL is http://www.urbi.ubi.pt/i/ws.php. The file is structured as follows the next empty example and we can see all the attributes that applications use to display the information:

```
<?xml version="1.0" encoding="UTF-8"?>
<channel>
<title /><description /><link /><language /><generator />
<items>
   <item>
      <id />
      <title />
      <description />
      <pubDate />
      <link />
      <image />
      <imageCaption />
      <thumbnail />
      <text />
      <author />
      <category />
   </item>
</items>
</channel>
```

4.3.4 Web Applications internals

The client side application is based on the browser interpretation of the following file-types HTML, XML, CSS and JavaScript frameworks. The process of delivering the news is divided in several phases. We can see part of the process in the previous figure 4.7.

Class diagram

The following figure 4.12 represents the class members, such as attributes and methods, of our News Version. But, the versions RSS and Edition uses an identical static structure.

In figure 4.16 we can see the DOM structure for this application. We can find there similar name elements to those presented in figure 4.12. This gives us a clear view of the structure classes, the page structure, and what's their interaction. For example, the *listController* of our class diagram interacts with the *listLevel* and it's elements *rowTitle, rowImage, ...* The methods *goInfo(),*

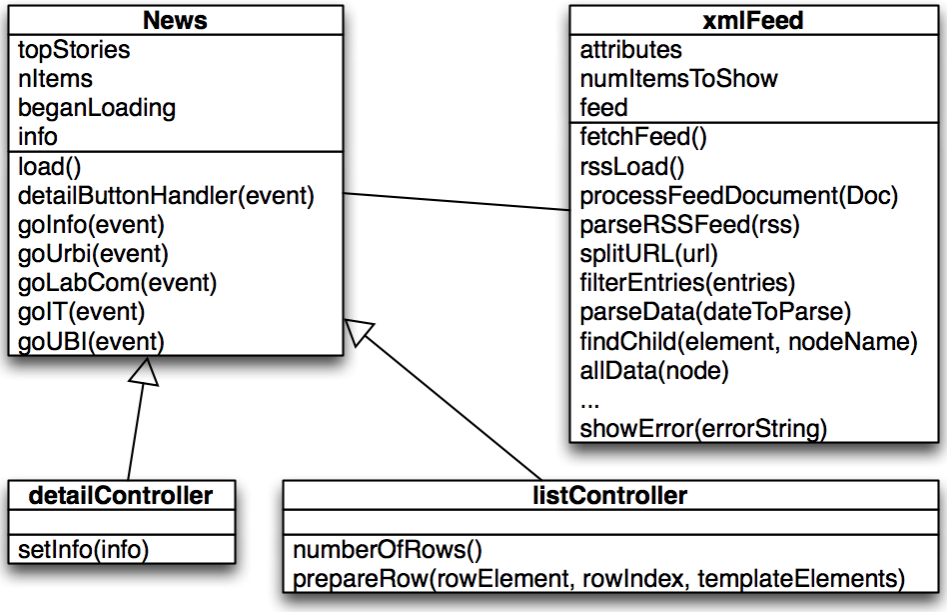


Figure 4.12: UML Class diagram for the *Urbi et Orbi* – News Edition Web Application

goUrbi(), goLabCom(), goIT(), goUBI() we're created to handle event associated with the *buttonURBI, buttonLabCom, buttonIT, buttonUBI* in the *infoN4MD* panel.

Stage sequence

The stages, in our Web Applications are very straightforward. There's an initial request for a Web page. Then, the Apache server responds through on single HTML file. Within the file there are other linked resources that forces the browser to perform more requests to two different servers. Then those server retrieves the information wanted.

The figure 4.13 shows the UML Sequence diagram with the Browser actor on the left and the two servers on the right. This diagram can represent all the different versions of our Web Applications, but note that, the asynchronous XMLHttpRequest is only performed by the News, RSS and Edition versions – represented by * in the diagram.

The versions with XMLHttpRequest object connect directly to XML data for updates without reloading the page. This process is done in the background. [12] Normally two JavaScript functions are used to provide Ajax requests: *loadXML-Doc*, *processReqChange*. These generic functions includes object creation, event handler assignment, and submission of a GET request. [12] After creating the object through an ActiveX constructor, we can use several other methods (*abort,*

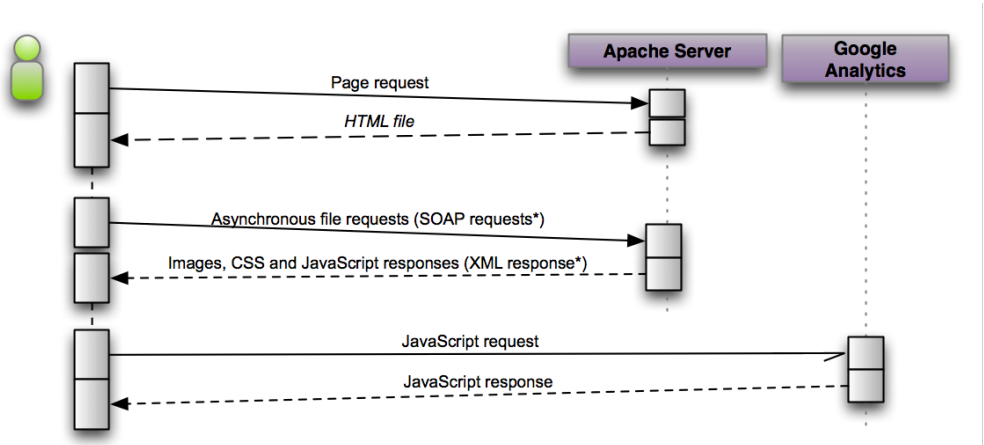


Figure 4.13: UML Sequence diagram for the Web Applications

getAllResponseHeaders, getResponseHeader, open, send, setRequestHeader) and properties (onreadystatechange, readyState, responseText, responseXML, status and statusText) to manage the connections. The XML data is then converted (parsed) into renderable HTML content.

Life cycle

As we already show, The News Web Application only uses one communication stage to deliver the information. That's why we divided the process of delivering information in three different stages, see figure 4.14.

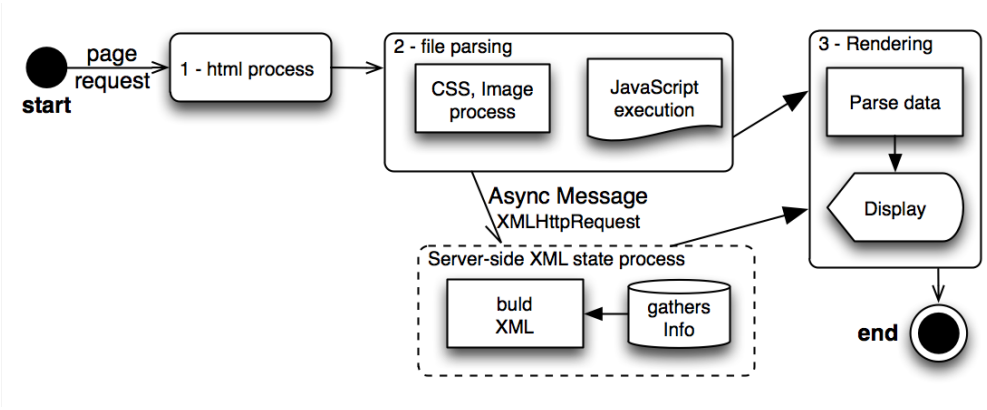


Figure 4.14: Web application life cycle diagram for the News and RSS versions

After the browser request and server responses, we have the Stage 1 at the client side – the browser. It's the interpretation of the headers and body of the HTML. Then we have more request to the server, the responses are CSS, Images and Javascript.

In Stage 2 the browser parses the information and executes the JavaScript. In

our scenario there's a new XMLHttpRequest for getting the info to populate the HTML. Despite this stage is server-side our figure 4.14 shows it in the Server-side XML state process.

The Stage 3, is when the browser parses all these response, populates the HTML, joins the images, and renders all this to the display and awaits for user further user action.

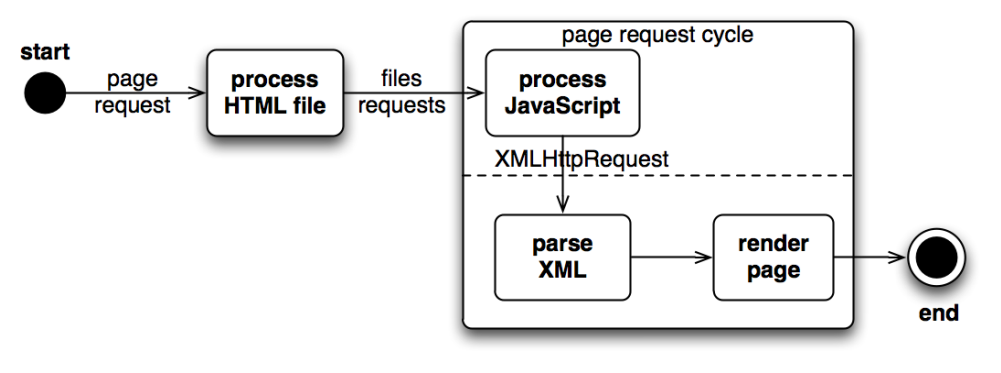


Figure 4.15: UML State diagram for the Edition version

The Edition version is different from the News and RSS versions because there's a XMLHttpRequest every time a page is requested. As already mentioned the News and RSS versions are all-in-one file content information, requested in one initial state by the browser. can see it represented in figure 4.15.

4.4 Optimizations

As we know the iPhone's mobile version of Safari has some limitations when running Web applications, like the wireless communication speed and type, the 10 MB cache per Web page or the time limit while executing integrated JavaScript.

Knowing this, we've use some tweaks to out-perform the communication process for these Web Applications by using Web design and data delivery concerns such as CSS, JavaScript and scripting design. We already talk about-it in Chapter 3.

We've also tweaked our system implementing the suggestions of Yahoo's Developer Team [51] (Host of the YSlow for Firebug) written to improve mobile to server communications. We've used this plugin to test our Web applications, result presented in the previous section.

For the specific iPhone versions we've come up with optimizations for Ajax to let users have an efficient Web experience by getting your pages updated quickly and smaller downloads. [15]

We will go through some of the items in the document and talk about the important changes applied to our Web Applications, that affected the site performance. Note that, although the document present server-side optimizations,

we didn't fulfill all of them because it required more servers and domains that, at this point, is not necessary.

Minimize HTTP request – by reducing the number of components downloaded in the Mobile and Edition version we've merged the JavaScript files in one single one. The other versions use the Templates original version presented by Apple Dashcode/UIKit.

Add an Expires or a Cache-Control Header – was included in the header tag of the HTML file. The date used is dynamically set to two years from now.

Gzip components – that allow us to compress the HTML and script files. To do the compression we call the php functions:

```
ob_start{"ob_gzhandler"}; ... ob_flush{};
```

We also configure the apache, on the server, to automatically compress and negotiate compression of text/html files with the browser through the *zlib.output∗* directives.

Put stylesheets at the top and scripts at the bottom – Only two of the version were optimized – Mobile and Edition. We also follow the **Avoid CSS Expressions** recommendation.

Make JavaScript and CSS external – All the versions were optimized with this directive except some sub-pages of the Mobile version that includes CSS and JavaScript in the HTML.

Minify JavaScript and CSS – The CSS and JavaScript files, in the versions Mobile and Edition, were re-arranged by us in the development phase. We removed the line-breaks, spaces, and code comments. This allows us to reduce the file size from 15KB to 11KB (it's about 25% savings).

Configure ETags – The server does not support identity information.

Flush the buffer early – The Edition version this optimization. This make the browser start fetching components while the server is busy with the rest of the HTML page. The Mobile version uses the GZip compression and cannot use this tweak because the flush is just applied in the end of the page.

Choose $< link >$ **over *@import*** – All the versions are optimized with this directive.

Optimize images and don't scale images – The images used in the first page have a 30% jpeg quality index. The detail page have 60% or 90% quality and the image page versions have 90% exclusively.

Make favicon.ico small and cacheable – The icons are small images of Graphics Interchange Format (GIF's), with less than 5 KB.

Keep components under 25 KB – Almost all the files have the size under the 25 KB. The *ws.php* and images with 90% quality are known-to-be files with size with more than 25 KB.

4.4.1 Ubiquity

We've implemented some server side computations to seamless optimize Web content for these versions. We've already talk about them in this chapter, but we want to add more perspectives and options with what we can do to improve our service.

Commute between different versions – We have four major version of our site. The server could choose what version to deliver regarding the client. Of course, our versions are different in design and functionality, so we have to merge them to seamless deliver the service.

For example, if the client browser is the Safari on the iPhone than the server shows the Dashcode/UIKit versions. For other browsers and operating systems the system could deliver the iWebKit or the WebApp.Net framework versions.

Location-aware – DNS reverse lookup gives us the location of the client. We can choose specific news for each region, or by offering an different text language. And, if the IP address is related to the intranet the information has no item limits.

Time-aware – The hour of the request is useful to determine if it is a Internet Congestion hour, night hour, day-light, weekend version. We could also use different CSS version for night of daylight in order to improve contrast. This has not been done also.

4.5 Web Applications Development

There are several different approaches to Web development, we took the engineering-based approach as it was developed piece-by-piece.

4.5.1 Requirements and analysis

Initially, we wanted to build a striped-out version of our Online Desktop site. By remove services such as uploading, sending to a friend and commenting we were able to simplify the application becoming a more static Web page.

Due to screen sizes, and device limitations we wanted to strip down multimedia content; limit and reduce the size and weight of images; maintain all the text and related new info (author, category, published date, section, image

captions and english version) and build a all in one column simple design and navigability.

Those requirements are still valid today, but, we now added more features like, section and category listings, navigation through thumbnails and slideshow.

The biggest problems were: the small screen size and the different interpretation of the code made by browsers. We've come around with four different versions of the site. They have small design differences and to maximize compatibility we've used different toolkits in their construction.

The speed problem was resolved by information size, compression and asynchronous communications.

4.5.2 Case studies

We did make an analysis of other mobile versions of portuguese news site. None of those were an iPhone specific version. Because of that, the majority lack on the viewport element, and that, as we know, makes it to look weird on the Safari browser on the iPhone.

Some of them are called PDA versions which kinda makes them old-fashioned. Despite of that, they all look the same. One column design with top news title on top followed by category listing options. Small amount of images or none. Titles are links to detail versions of the new.

The credits goes to three mobile versions. Two of them are journal with economics related news (Económico e Jornal de negócios), the other is an major newspaper know for it's design and simplicity (Público). The target public for these versions are, coincidently, the typical market for iPhones (gadgets, and other top mobile devices) buyers. They have design and usability concerns for their mobile versions. They are more than a stripped out desktop version.

For future references, here's the name, category and URLs of the active online studied versions in August 19, 2009:

- Público [Daily newspaper]: http://mobile.publico.pt
- Correio da Manhã [Daily newspaper]: http://www.correiodamanha.pt/pda/
- Económico [Economics newspaper]: http://mobile.economico.pt/
- Jornal de negócios [Economics newspaper]: http://www.jornaldenegocios.pt/
- PC Guia [Technology magazine]: http://m.pcguia.xl.pt/
- IOL Portugal diário [Online journal]: http://m.diario.iol.pt/
- A Bola on-line [Sports daily newspaper]: http://abola.pt/pda/
- Jornal Record [Sports daily newspaper]: http://www.record.pt/pda/
- O Jogo mobile [Sports daily newspaper]: http://m.ojogo.pt/
- Automotor [Motoring magazine]: http://m.automotor.xl.pt/

4.5.3 Process

The deployment of a Web Application can be a very simple operation or can become very complex, depending on the size of the project, its complexity, the number of team members involved, the project management process used, and many other factors. For project management we've selected major topics and tasks representing the global development, see table 4.5.

Tasks	Dec	Jan	Fev	Mar	Apr	May	Jun
Requirements	✓		✓		✓		
Mobile/iPhone version		✓	✓				
News/RSS versions			✓	✓	✓		
Edition version					✓	✓	
Design			✓	✓	✓	✓	✓
Debug, Quality and Tests						✓	✓

Table 4.5: Web Applications project management

4.5.4 Implementation

In previous sections of this chapter, we have already described the technology, the tools, features and major characteristics behind our mobile versions. We will now refer to some important aspects that, for different reasons, helped our development work.

4.5.5 Statistics and reports

Statistically, our mobile versions, in the last two months, represents more than 1.5% in total number of visits of the *Urbi et Orbi* site. Please note that the desktop version is online since 2000, and all the archives visits did count for the statistics.

An overview analytics comparison, between mobile versions and desktop version, tells us that the average time on site is bigger by 75% in the mobile, but, the downside is, the bounce rate is 50% less.

Two months is not enough for big conclusions, specially when we are in the "silly season" and, as we know, the Internet traffic slows down, and, the mobile Internet traffic is even slower. Despite all that, we believe that, with the uprising number of mobile devices, with Internet capabilities, in the world, these numbers will look better from now on.

4.5.6 Development resources

To build our Web Applications we've used several tools. In chapter 3 we've talk about some of Apple's tools, but there are others. For example, to develop the Web Service we've used Adobe Dreamweaver to code HTML, CSS and PHP Web

pages. To build the databases we've used the PG Admin tools. The Web Service itself is hosted in a SuSE Linux Operating System using the Apache Web Server for managing Internet requests. On the client side, and apart of the iPhone mobile device, we've also used other platforms. Operating systems like Linux Ubuntu, Mac OS X, Leopard and Tiger versions, Microsoft Windows XP and Vista, Microsoft Windows Mobile. On those operating systems we have installed and tested our Web applications with different versions of the following Internet browsers: Opera Desktop, Opera Mini, Microsoft Internet Explorer, Microsoft Internet Explorer Mobile, Camino Browser, Mozilla Firefox and Google Chrome.

Dashcode in Action

We have used the Apple's Dashcode tool to build the News and RSS versions, for Safari on the iPhone, that provide discrete functionality to users. [10] It's important to show the real world scenario and how we've come up with some of the Web Applications.

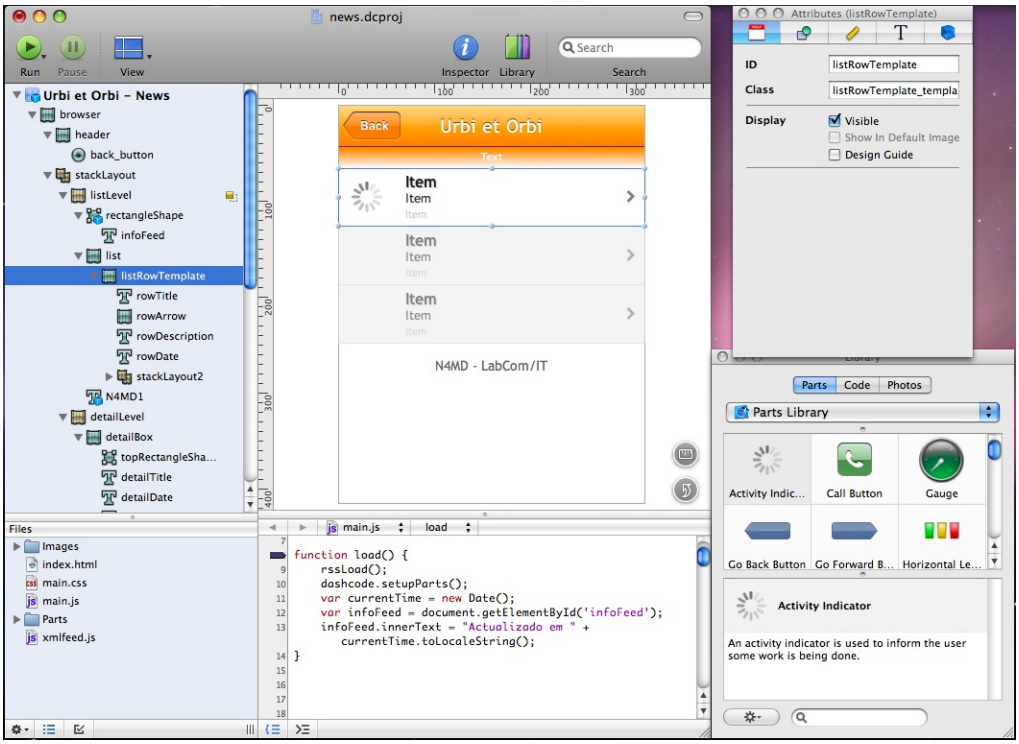


Figure 4.16: Dashcode in Action - The *Urbi et Orbi* - News Web Application in production

We can see in the workspace:

- In the left area, the DOM objects (top) and the folder/app file structure (bottom). The DOM objects properties can be viewed and changed in the Inspector panel. The select object listRowTemplate in this section is also selected in the canvas and in the inspector panel;

- At the center, the canvas with the first page item list (top), and in the bottom the code editor where we programmed all the methods and classes included in the application. The canvas has a little button for the user change the orientation between portrait and landscape;
- On the right, the Inspector and Library panels in the top and bottom respectively. The Library panel has the major objects used to create the most famous applications for Safari Mobile.

Chapter 5

The iPhone Native Application

The News For Mobile Devices (N4MD) is a simple application for viewing the weekly news produced in the *Urbi et Orbi* journal. This application run natively on iPhone devices and the primary difference from the previous Web applications is that the content is available for offline reading.

In this chapter we will describe our native application N4MD.app. First we will describe the application and its features. Then we will list requirements from partners and also the pre-requisits. In the development phase we describe the process, the application internals and the tools used. Finally, a comparison with the our previous Web applications.

5.1 The News For Mobile Devices Application

This is our first native application for the iPhone, it's our first application developed using the Apple SDK, it's even the first application coded with Objective-C, and, we still are in the fast growing learning curve of this technology. Nevertheless, it has the foundation for a World wide quality application to be distributed thru Apple App Store.

Next, we will describe the News For Mobile Devices application and present the several Frequently Asked Questions to be included in the Web page that will follows the application launch on App Store.

5.1.1 The N4MD Application user interface

This application downloads and syncs, thru the network connection, the latest news directly to your mobile device. It has a top-down list of items and a left-to-right navigational interface.

The icon

After downloading and installing the application on the iPhone we can see the icon on the program list, see figure 5.1.

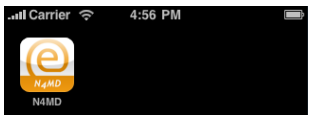


Figure 5.1: N4MD iPhone interface icon partial screenshot

The splash screen

When the N4MD icon is clicked the application is launched and the first thing we can see is the big image logo for the application (figure 5.2), this is the default splash screen.

Figure 5.2: N4MD Default image while initiating

The 'News Table View': First Tab View option

The default primary view is the 'News Table View'. This window has four distinct areas, as shown in figure 5.3. From the top to the bottom:

1. 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 icon); the clock information; and the battery status.

2. Top Bar Navigation – Table's title "Urbi et Orbi News" and reload button on the right.

3. 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).

4. The Navigation Bar – The bottom bar in black with two options: "Urbi et Orbi News" and "Settings".

Figure 5.3: N4MD Application First Tab View named 'Urbi et Orbi News'. Application 'Primary View' screenshot

The 'Detail View': 'News Table View' sub-view

The 'Detail View' (figure 5.4) appears when a cell item from the 'first window' is clicked. This is a sub-view of the previous one, so, the Top Bar ('News Table View' part number 2.) and it's content are related.

The 'Top Bar' has three items: The 'back' button with the title of the table information "Urbi et Orbi News", the item information – contains item index and total items –, the segmented buttons with navigation thru items.

The item full information appears in the white area below the orange 'Top Bar'. This is a vertical scrollable area. We can see here, from top to bottom: title, description, image, image caption, published date, category, author info and full new *corpus* text.

The 'Settings': The Second Tab View option

In this view (see figure 5.5) the user can choose if he wants to view images, or not, thru the slider button on the top.

Figure 5.4: N4MD Detail View

Next, we can find some 'control information' regarding the device itself: the device unique identification, the name, the system version and the battery information. This can be coded with the help of iPhone OS Library at iPhone Dev Center. [11]

Figure 5.5: N4MD Settings View

Navigation

The navigation actions are controlled by the usual touch screen of the iPhone. 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 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, yet, support 'Rotation View' because we primarily tried to mimic some native Apple iPhone applications like Phone.app, Clock.app, or iTunes.app. which doesn't support 'Rotation View' as well.

5.1.2 Frequently Asked Questions

In this section we try to explain the functionality and some basic features of our application in the form of answers to common questions[1].

What is the N4MD iPhone application?

The N4MD iPhone application is a new way to take the high-quality news of *Urbi et Orbi* with you, on the iPhone and iPod Touch, wherever you are. This application downloads and syncs the latest news directly to your device.

It syncs via Wi-Fi or cellular network and downloads the latest news directly to your device. It has been designed to take advantage of the advanced capabilities and rich user interface of the iPhone.

What features does the N4MD iPhone application have?

It's specifically designed to use the features and usability of the iPhone and it's compatible with operating systems version 3.0 or superior.

It has features such as:

- Offline reading: access *Urbi et Orbi* articles to read online and offline;
- Simple navigation: view pictures and articles quickly thru the horizontal navigation structure;
- Customization options: select if you want to view or download images with the articles;
- Internet connection type detection: only download images if network if 3G or Wi-Fi;
- Saved state on exit.

[1] We choose this format because it will be used in the Web page of the application

Does it cost anything to use the N4MD iPhone application?

The application itself is free to download and use. Please note, that in the absence of a Wi-Fi connection, the N4MD iPhone application uses your iPhone's cellular network connection to access and download content.

Does the N4MD iPhone application include advertisements?

No.

What devices will the N4MD iPhone application work on?

The N4MD iPhone application works on any iPhone or iPod Touch with operating system version 3.x. The N4MD application will not work on any other iPod device. To determine your device's OS, connect the device to your computer and refer to the appropriate "Summary" page in iTunes, or go to the "Settings" area on the device itself.

What is the difference between the N4MD iPhone application and the *Urbi et Orbi* mobile/iPhone Web sites?

The *Urbi et Orbi* mobile/iPhone Web sites are accessed through your phone's Web browser and requires a continuous Internet connection.

The N4MD iPhone application allows you to view news content directly on your device for offline reading after the first download. The N4MD iPhone application was also designed to take advantage of the advanced capabilities of the iPhone user interface and includes personalization options and formatting designed specifically for your iPhone.

What news can I get on the N4MD iPhone application?

The application offers the latest articles from all sections of *Urbi et Orbi*'s weekly edition.

Certain features are not available in the iPhone application, including multimedia (video), classified listings – such as latest, suggestions and archive –, and tools such as *Urbi on Images*, comments and send–to–a–friend. Please visit *Urbi et Orbi* or www.urbi.ubi.pt/i/ for those features.

If you have suggestions for enhancements to the iPhone application, please let us know at urbi@ubi.pt.

How do I personalize my N4MD iPhone application?

The default navigation bar (at the bottom of the screen) includes buttons for "Urbi et Orbi News" and "Settings". To customize the N4MD application, click "Settings" then on the "Load Images" area screen, change the option slider next to it. The images will be displayed – or not – during synching, so, if you change this setting you have to reload the 'news content' to see the changes.

Will the N4MD iPhone application be updated or add new features to?

Yes. We are actively working to expand the feature set of this iPhone application, and will be updated periodically.

How long does it take to download the news?

Under typical Internet connections, the following times have been observed while performing a full sync:

Sync Type	Wi-Fi	3G	EDGE/GPRS
Full sync without thumbnails or article images	3–10 sec.	–	–
Full sync with thumbnails and article images	5–25 sec.	–	–

Table 5.1: Typical download times of news in the N4MD iPhone application

For best results, we recommend using a Wi-Fi connection whenever possible. Please note, that the synching of article images is not recommended while using 3G or EDGE/GPRS, as the impact upon battery and data usage may be noticeable.

How do I initiate a sync?

To sync, press the arrow or "refresh" button on the main screen of the application, in the "*Urbi et Orbi* News" tab. A new sync will also be initiated each time you open the application, and synching occurs periodically while you use the application. To keep your connection active while synching, you must leave the N4MD application open. If the connection is interrupted for any reason, either to use another application or to receive a phone call, you will need to relaunch the application and restart the synching process.

Can I use the N4MD iPhone application when traveling?

Yes, but note that in the absence of a Wi-Fi connection, the application uses your iPhone's cellular network connection to access and download content. Normal network data rates apply, and you may incur roaming charges while traveling. Check with your local carrier. Alternatively, you can enable Airplane Mode to browse and read previously stored news without incurring data access charges.

Can I read the news offline or in Airplane Mode?

Yes. The N4MD iPhone application is designed to store content locally on your iPhone. Once you have completed the sync process, you will be able to access stored content even if Airplane Mode is activated, or if you do not have cellular

network coverage. Note that, to see the articles you must have fully downloaded in the first use.

Can I adjust the amount of content that is downloaded to the iPhone?

The default settings on the application is one week of news content. Nevertheless, you can change the 'Load images' option to "Off", in "Settings" tab, to avoid bigger downloads.

Why isn't my N4MD iPhone application content being updated?

If your content is not being updated correctly, first try resynching as described in, "How do I initiate a sync?" Note that there may be a slight synching lag between the article lists and the full article content. If the Internet connection or the server is offline/down you will see an error message.

You may also want to try removing and reinstalling the application.

Why don't I see article images?

Not all articles include images, but if you don't see images in any articles at all, verify that images are set to sync automatically. Go to the N4MD "Settings" tab and in the 'Load images' area check the option slider. Make sure is set to "On".

For EDGE/GPRS networks, this 'Load images' option is set to "Off" by default, to improve download speed and preserve battery life.

How do I remove the N4MD iPhone application from my iPhone?

To remove the N4MD application, follow the same procedure that you use for any other iPhone application. Press and hold the "N4MD" application icon until it begins to wiggle. Press the "X" in the corner of the icon and confirm the deletion in the window that appears.

Do I need to register for *Urbi et Orbi* to use the iPhone application?

No. The N4MD iPhone application does not require any *Urbi et Orbi* registration, though you will need to log in to your iTunes account in order to download the application from the App Store.

5.2 Pre-requisites

Generally, 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 the iPhone, to allow user to read them offline. But, the application must also provide some form of user control to override the application and server ubiquitous decisions.

We also intended to study and deploy, most of the main features already available in other applications. This study and features will be explained and listed later in this chapter.

5.2.1 Ubiquity

Ubiquitous computation is an important matter for mobile computing. For us it's a major requisite. Therefor, we needed to develop two types of ubiquitous decisions: One on the server, and, the other for the native application.

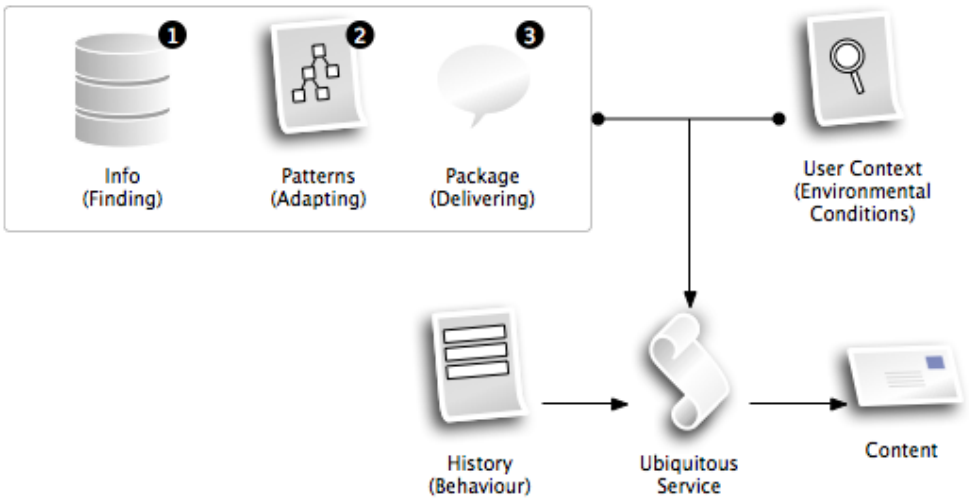


Figure 5.6: Context-aware ubiquitous system model

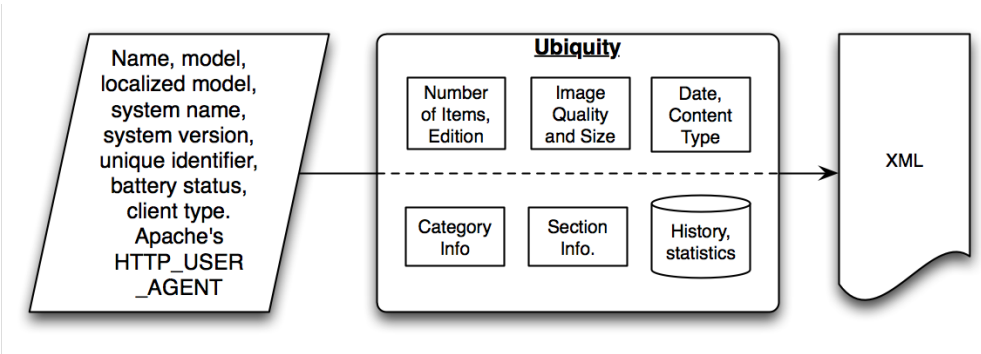
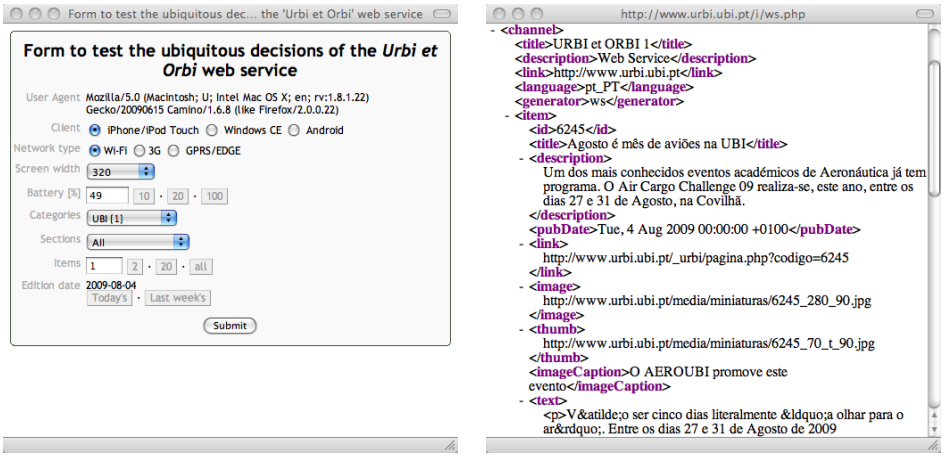


Figure 5.7: Input, process and output information model of the ubiquitous process

Server-side

In the server side we have to consider different types of clients. Devices like iPhones, Windows Mobile[2] or Google Android[3] can post requests, therefor, the answers must be adequate and specific for these kinds of systems.

Thus, users have different needs and there are other types of applications, so, the parameters for posting to the server, in order to make those ubiquitous decisions, are: User agent, Client device type, Network type, Screen width, Battery, Categories, Sections, Items and Edition date. See figure 5.7.



(a) Form test for the Web service

(b) XML file result

Figure 5.8: Form test and the respective XML Result screenshots of the Ubiquitous Web Service

The Web service is also prepared to answers to specific requests like limited number of items, specific category or section, previous news and editions. But, most of the information needed comes from the Apache server and his directive "HTTP_USER_AGENT" which has information like: Request Time stamp, Client type and version, Operating System and respective version, and client language.

The requests and answers can be explained in the following figures, and, this is important for developers testings. Note that, the N4MD iPhone native application is not the only client for the web service. A Microsoft Windows Mobile native application is also being developed.

Figure 5.8(a) shows the Web page we've created to perform some tests. We can see some of the options that the ubiquity module have to deal with.

Figure 5.8(b) is the computed answer in the XML file format. We can see the returned nodes XML structure.

[2]Windows Mobile Developer Center.
http://msdn.microsoft.com/en-us/windowsmobile/default.aspx

[3]Android Developers. http://developer.android.com/

At this point, our application just sends the information about the Battery and the Client in the request. This information will change the default settings for 'Number of items' and 'Image quality' in the Web server.

Client/Application-side

As we already saw, the communications with the server is important for the application speed and for the device battery life. So, one of the requisites for the applications would have to be related with type of network and images.

We wanted to ubiquitously avoid the image downloading, but, on the other hand, we have to had an option to override this. We also wanted to use one of the features from the framework, the network image caching.

We've implemented all of those features, and more, in our application. Here's the list:

- The default option is image downloading;
- Thumbnail creation costs valuable system resources, so, the image thumbnail have to be downloaded also;
- The application won't download images if the Network connection is the EDGE type;
- The application will look for cached images for every image request;
- The application won't download images if the setting 'Load images' is "Off";
- The application will show a default image for the item downloaded, or, in case of an item entry without an image;
- The application won't show the 'Image Caption' label if the item have no image.

5.2.2 Requests

There are several entities interested in this project. Originally, this application was born to be a part of an academic project, but, the main goal is to deploy-it and distribute a version of the N4MD application on the App Store.

Publishers request

The desirable options and features from *Urbi et Orbi* staff for this application was:

- Thumbnails for the list items;
- Bigger photo for the detail view;
- Good quality images;

- Complete edition with full article information;
- Similarity with some 'best' news related applications available today on App Store (for example: NYTimes and WSJ);
- Availability to follow external Internet links included on the text, including video and audio URLs;
- Quality for deploying and iTunes distribution, intended to be the first Portuguese News distribution system running natively in the iPhone.

Developers goals

The following general goals were defined since day one.

- It had to be error and bug free and with good memory management. Apple staff will, probably, look into the code before put it available in the App Store, so, it had to have readable and quality code;
- To use Apple's User Interface and Accessibility Guides and general recommendations;
- To code using the SDK, standard frameworks and the Objective-C language;
- Compatibility with the latest iPhone/iPod Touch devices the respective iPhone OS versions;
- Programming using the Object oriented paradigm, code quality and well documented with no obfuscation. If it has imported or foreign code, it has to be Open-Source or with a GPL v2 software license. No payed software is tolerable.

5.3 Development

Known the pre-requisits and what goals we want to achieve we've started the development phase. In this section we'll talk about the coding, decisions,

After the test-drive and analysis of the previous applications we've came up with some important notes in terms of User Interface and User Experience. We can see them in the figure 5.18.

5.3.1 Process and description

Here are the major topics while developing the native application (see table 5.2), but, some of them were mixed with the Web applications as well.

First, and before starting developing, there was a learning stage. Because learning the platform was an influential factor for the success of this project, and also a major difficulty one, here's some of the things that we have done to overcome it: Getting the literature; Signing for Apple Developer Connection (ADC);

Stages	Jan	Fev	Mar	Apr	May	Jun	Jul
Scope and Requirements	✓	✓					
Functional Design		✓	✓				
Web Service development			✓		✓		
Client application UI				✓	✓	✓	✓
Debug, Quality and Tests							✓

Table 5.2: N4MD Project Timeline

View all the videos in ADC about iPhone development; Download and install the examples; Subscribe to Stanford U class (cs143p) on iTunes; Developing cs143p exercises; Search the Internet for tips and code examples while developing.

Milestones

While developing, we had several important steps that gave us motivation and guidance. This major milestones are listed here:

- The download of the XML file and save it to disk;
- The creation of the 'Table View' with custom cell information; [17]
- The connection and appearance of the 'Detail Views' for each item;
- Customizing information with code on the 'Table View' and on the 'Detail View';
- The speed improvements thru threaded request for images;
- The optimal memory management and code quality.

5.3.2 Case studies

We've done some research on the App Store for applications from the "competitors" of our University newspaper. We downloaded, installed, studied and analyzed the following listed applications on the table 5.3 bellow in the respective major items.

The following tests and screenshots were made in an iPhone 3G with OS 2.1. The copyright of those images, and content are related to the respective application name. We did not modified or altered the images whatsoever.

The table 5.3 represents the major topics of analysis. The column Geo–localization is 'Yes' when the application asks user to use 'current location'. The 'Stars' column is our evaluation of the application. Higher values go to applications with best design and that has our features of interest or requisits.

Application	Type	Publi–city	Geo–location	User settings	Grade/score
20Minutes	News/Radio	No	No	No	4
A3 Notícias	News/TV	Yes	No	No	4
ABA Journal	News	No	No	Yes	3
AP Mobile	News	No	No*	Yes*	4
BBCReader	News	No	No	Yes	5
BGRMobile	News	Yes	Yes	Yes	5
El Universal	News/TV	No	No	No	4
ElPais	News	No	Yes	Yes	5
Estadão	News	Yes	No	No	3
Fox10tv	News/TV	Yes	No	Yes	5
ITNNews	News	Yes	No	No	4
JakartaGlobe	News	Yes	No	Yes	2
KCTV5	News/TV	Yes	No	No	3
la Republica	News	No	Yes	No	4
LANews	News	Yes	No	Yes	5
Le Monde.fr	News	Yes	No	Yes	3
Le Parisien	News	Yes	No	No	4
Marianne2.fr	News	No	No	No	2
NU	News	No	No	Yes	4
NYTimes	News	Yes	No	Yes	5
stern.de	News	Yes	No	Yes	4
Straits Times	News	No	No	Yes	4
Sydsvenskan	News	No	No	Yes	2
Telegraph	News	No	No	No	5
FT Mobile	News	Yes	No	No	4
TSRInfo	News/TV	No	Yes	No	4
USA Today	News	No	No	No	5
VOX	Magazine	Yes	Yes	No	1
WCPO.com	News/TV	No	No	No	4
WDTN	News/TV	No	No	No	4
WRAL.com	News/TV	Yes	No	No	4
WSJ	News	Yes	No	No	5

Table 5.3: Comparison chart of available App Store apps related with News

* Application requires user to select region and news categories on first usage.

The sub-views and detail views were also studied. There are some questions we thought about, things we have thought. Some of them we decided to deploy with our applications, others, we don't.

Important notes:

- The applications of type 'News/TV' have the option of video, but not in the primary view. The same for image gallery or image slideshow;
- Some applications only show publicity on detail view, others only on primary view, some of them in all views. The publicity was always at the

bottom, as an image with fixed width (even in wide screen);

- The Geo-location is 'Yes' when the application asks user to allow the use of the location service. We could not test the location-aware of the application because we only perform the tests in Portugal (Europe);
- User settings can be configured in the application itself, or, in the Preferences menu of the iPhone OS. We searched in both locations;
- Notifications in the iPhone icon (like missing call or un-readed messages), to signal new items or unread items, is not used in any of these applications;
- iPhone OS 3.0 implements a new kind of notifications, the so called "push notifications". Le Monde.fr is an example of application that uses that technology;
- All this applications have 'splash screen' while loading;
- Some applications require a license agreement on the first usage.

Note: We didn't perform speed tests of the application or the connections. At this point, we do not consider it a major topic or a quality factor.

We've also studied the application on other several aspects, but, from now on we only show the results for the applications with the higher score on the table 5.3. Note that, this 'higher score' is personal, it's for a general evaluation. The higher grades distinguishes applications with great design, user experience or options we would like to implement on our application. Or, a combined value for the previous items.

This evaluation was oriented for design characteristics. Later, in table 5.4 we will show the study for the primary view design, and table 5.5 indicates the general features and functionality of detail views.

Top Applications

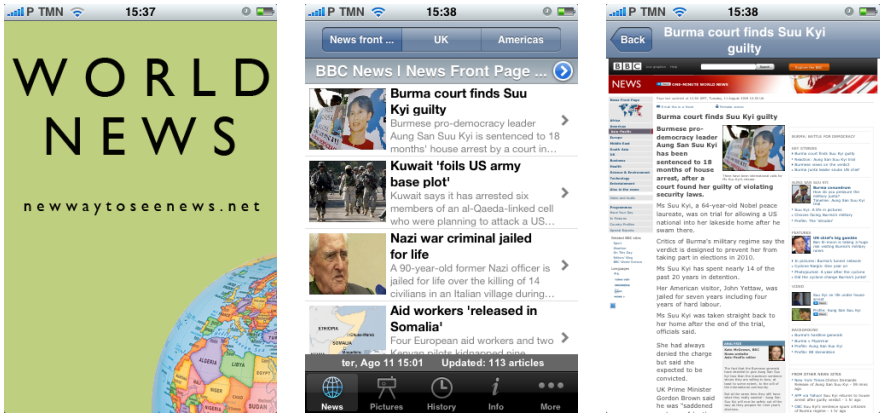
We'll continue your study, but, from now on, with just the top evaluated applications. Nine applications scored 5 in our ranking: BBCReader, BGRMobile, ElPais, Fox10tv, LANews, NYTimes, Telegraph, USA Today and WSJ.

We will study them and present the major features and characteristics. This study gave us a vision of what we want to improve in our application in terms of design and features. Our experience with our 'clients' and the experience as developers/publishers gave us a more effective design in the end.

BBCReader

Pros:

- Primary view with top bar segmented categories and latest download information;



(a) Splash screen (b) Primary view (c) Detail view

Figure 5.9: BBC Reader. British newspaper

- Big thumbnails and five lines in cell detail;
- Info. Section to display general content and application information.
- Pictures. Section with play or slideshow;
- 25 items on bottom Navigation Bar;
- Rotation view in detail view.

Cons:

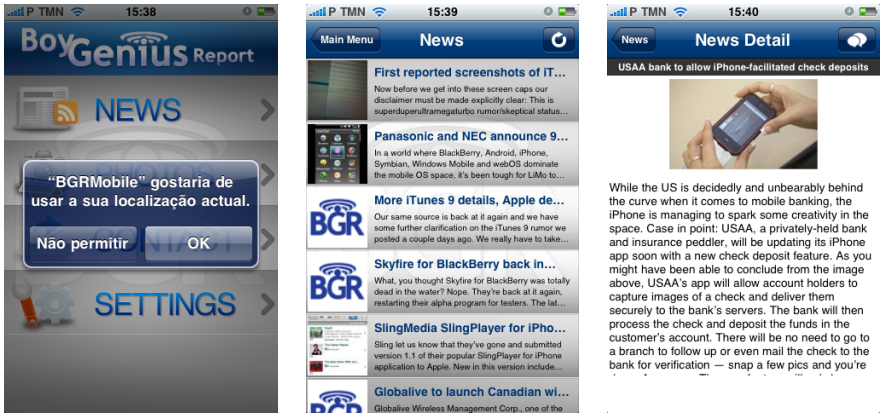
- The detail view is a full Web-page embedded and wider than 480 pixels and without options;
- Two different places to change settings;
- Can't edit Navigation Bar items;
- No last view saved.

Settings: Three segmented buttons to define pre-location for news; Offline mode [on—off];

BGRMobile

Pros:

- Design in primary view and table;
- Location-aware capabilities – showed in figure 5.10(a);
- Reload button on news items;



(a) Primary view (b) News items (c) Detail view

Figure 5.10: BGRMobile. Boy Genious Report

- Comments listings and possibility to post comment in detail view.

Cons:

- Three levels to detail view;
- Poor design in detail view – bad title design.

Settings: Auto fetch location [on—off]; Auto fill user info [on—off]; Post and comment text size [10—11—12].

ElPais

(a) Splash screen (b) Primary view (c) Detail view

Figure 5.11: ElPais. El País, Spain news journal

Pros:

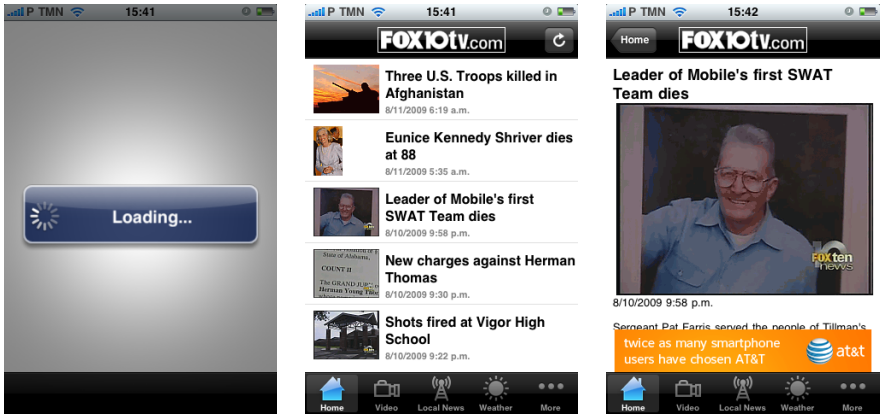
- Location-aware capabilities;
- Design – Colors on top bar and background;
- Detail view – Segmented buttons on top and send email on bottom;
- Last download information on primary view.

Cons:

- Primary view design – thumbnails on the right;
- Detail view with small thumbnail.

Settings: Edit bottom navigation bar items.

Fox10tv



(a) Splash screen (b) Primary view (c) Detail view

Figure 5.12: Fox10tv. Fox 10 channel news from USA

Pros:

- Alert message while loading;
- Simple and effective design;
- Video and Weather features; Detail view rotation available;
- Reload button on primary view.

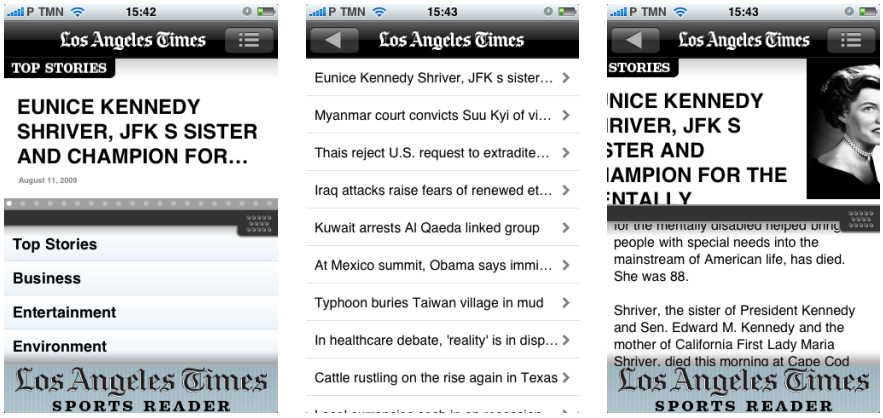
Cons:

- Publicity on top of info;
- No options on detail view.

Settings: Edit bottom navigations bar items.

Note: This application is very similar to WRAL.com, WDTN and WCPO.com applications. In fact, the Developer company is the same.

LANews



(a) Primary view (b) Category view (c) Detail view

Figure 5.13: LANews. Los Angeles Times news journal from USA

Pros:

- The interface design and user experience is different from all the other applications tested here;
- Left to right navigation in primary items;
- Button to change primary view layout;
- Category items in primary view;
- No Navigation Bar at the bottom;
- Photo slideshow in detail view.

Cons:

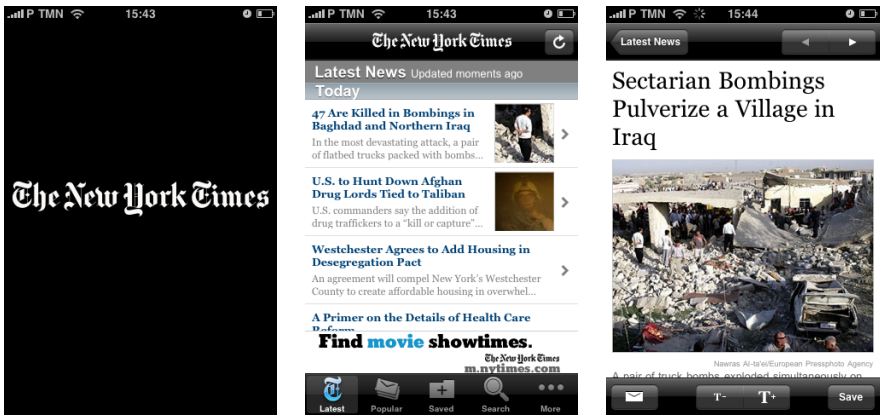
- Left to right item navigation can be tricky for less experienced users;
- The applications crashed a few times during test;
- No options for detail view.

Settings: News item text size on iPhone Preferences panel.

NYTimes

Pros:

- News items divided per day in table view;
- Reload button in primary view;



(a) Splash screen (b) Primary view (c) Detail View

Figure 5.14: NYTimes. New York Times news journal from USA

- Download bar to see progress (now show in figures);
- Save to disk and search options;
- Detail view with segmented buttons at top and four options at the bottom;
- Big Detail image in the detail view;
- Last view saved in user;
- Last download date information.

Cons:

- Design – Thumbnail item in right.

Settings: EDGE & 3G or Wi-Fi synchronization (Automatic sync [on—off], Auto sync every [x] minutes, Sync thumbnails [on—off], Sync articles images [on—off]); Save news for [x] days, Article font size [x pt], Landscape orientation [on—off], Accessibility mode [on—off], Reset NYTimes.
Note: The publicity image is on primary view and not on the detail view.

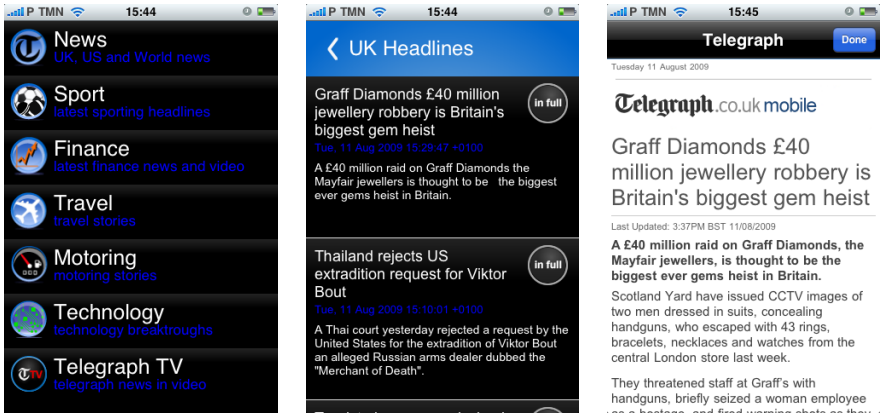
Telegraph

Pros:

- Items table view with full full title and full description (like category view figure 5.15(b)).

Cons:

- Four level approach to get detail view;



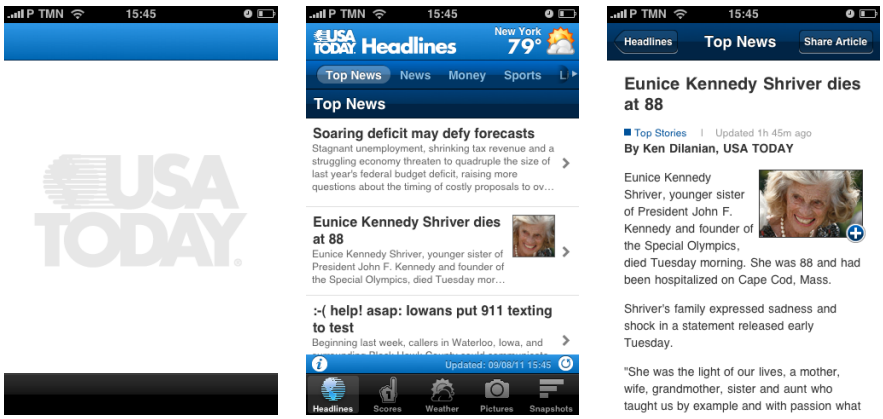
(a) Primary view (b) Category view (c) Detail view

Figure 5.15: Telegraph. Daily Telegraph news journal from UK

- Design – black background on primary view and category view;
- No thumbnail on items table view;
- No preferences for user.

Settings: None.

USA Today



(a) Splash screen (b) Primary view (c) Detail view

Figure 5.16: USA Today. News journal from USA

Pros:

- Weather info on top bar of primary view;

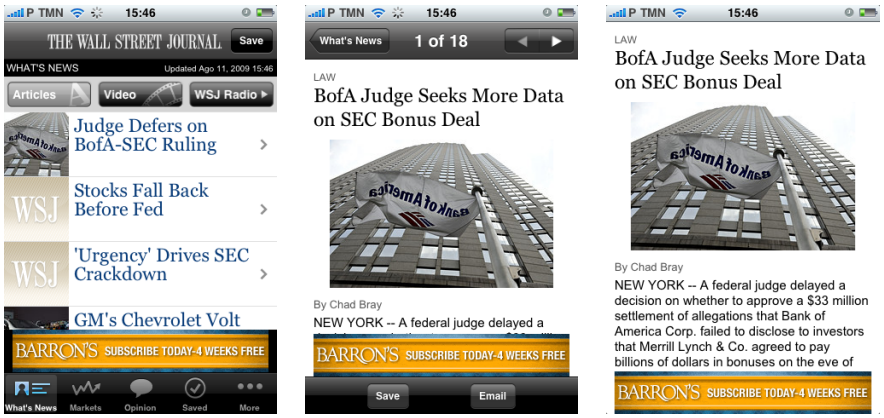
- Items on table list with five rows;
- Settings on primary view in the info icon.
- Last download information on primary view;
- Top bar option item on detail view;
- Category list on top of table;
- Picture slideshow;
- Saves last viewed view and makes it active when application finished loading;
- Zoomed image with caption in detail view.

Cons:

- Small thumbnail in primary view and detail view.

Settings: Twitter client [Twitterrific, Tweetie, TwitterFon, Twittelator]; Facebook login; Wheather locations and measure unit;

WSJ



(a) Primary view (b) Detail view (c) Detail view with hidden bars

Figure 5.17: WSJ. Wall Street Journal news journal from USA

Pros:

- Detail view automatic hides top and bottom bar after a few seconds showing more info on screen, show in figures 5.17(b) and 5.17(c);
- Segmented buttons on top bar in detail view;

- Last downloaded information on primary view;
- Category options on top bar of primary view showing radio and video;
- Saved last viewed window.

Cons:

- Publicity in all views;
- Just title and thumbnail on table view information of primary view.

Settings: Log In to WSJ.com; Update feeds every [x] minutes; Keep articles for [one day . . . two weeks]; Font size; Delete all saved items; Download new sections; Restore WSJ to initial state.

Top Applications primary and detail views

Application	Top Bar Buttons	Nav Bar Buttons	Table items	Thumbnails
BBCReader	4	25	30+	Yes
BGRMobile	0	4	4	Yes
ElPais	0	12	10+	Yes
Fox10tv	1	8	5	Yes
LANews	1	0	10+	Yes
NYTimes	1	16	10+	Yes
Telegraph	0	0	7	No
USA Today	7	5	10	Yes
WSJ	1	26	10+	Yes

Table 5.4: Comparison chart of primary view characteristics

The table comparison-chart-primary-view reflects the importance of the top bar and the navigational bar at the bottom. Some of the applications listed on table 5.4 doesn't have buttons on top bar, but there was always a top bar.

The bottom navigational bar is generally used for categories or special features like images, weather, and settings. That's why, in some cases, the number of items is bigger than twenty. The applications 'LATimes' and 'Telegraph' doesn't have bottom navigational bar because these options are listed in the form of a 'table list' in the primary view.

The number of table items depends on the number of available news, or, is limited by the selected category.

The importance of the visual thumbnail is demonstrated here, because, almost every applications uses it.

Application	Top Bar Options	Bottom Options	Images	Levels	Rotation
BBCReader	0	0	5+	2	Yes
BGRMobile	1	0	1	2	Yes
ElPais	2	1	1	2	No
Fox10tv	0	0	1	2	Yes
LANews	1	0	1+	1/2	No
NYTimes	2	4	1	2	Yes
Telegraph	1	0	0/1	3	No
USA Today	1	0	1	2	No
WSJ	2	2	1	2	Yes

Table 5.5: Comparison chart of detail view characteristics

Notes: This table 5.5 is used to justify our detail view design choices.

We also analyze how applications behave when no Internet connection is available. The table 5.6 lists our empiric observation. In the same table we've included information about interesting features related with connectivity or cached information.

Application	Cached Articles	Connectivity message	Reload button	Search feature	Save/ Share
BBCReader	Yes	No	No	No	No
BGRMobile	No	Error: Internet	No	No	Email
ElPais	Yes	Alert: No update	No	No	No
Fox10tv	Yes (no images)	No	Yes	No	No
LANews	No	No	No	No	No
NYTimes	Yes	Alert: No update	Yes	Yes	Save
Telegraph	No	Error: Internet	No	No	No
USA Today	Yes	Alert: Server	Yes	No	Share
WSJ	Saved	Error: Feed	No	No	Save

Table 5.6: Comparison on offline mode and special features

Note that, BGRMobile, LANews and Telegraph doesn't cache articles and needs Internet connection to see content that makes the application unusable. WSJ doesn't cache information but it can display saved content.

5.3.3 N4MD project design

The conceptual design involves three steps: defining our solution, refining our future set, and determining our users' mental model. The first two steps evolved from previous platforms (Desktop and mobile Web applications). The last step were based on our previous case-studies conclusions and notes.

We also know that the design of a 'new' on a Online platform is not just the text itself, there are other things to consider, like the overall graphic project, the photo and the structure of the elements. The small amount of different colors in our application due to the fact that, with fewer colors, the viewer doesn't loose time comprehending the chromatic universe of other cultures, societies or other different viewers. [33]

From the case studies we concluded that we like the bottom navigation bar, its used in several Apple iPhone applications, and provides great sense of location in a multi-level application. It's also used by the majority of the studied applications, so, users should be familiar with this element.

Other elements were also used for the same reasons, the central table view, the top bar, the reload button.

The table view were customized to have left side thumbnails, title with full text and two lines for the description. [9] If there's no image for the item the application should load a default image like the Wall Street Journal application (figure 5.17(b)). At the bottom of the table there's the status with information of file date and number of items.

The colors used on the top and status bar are the same we can find in the primary Web site, and in the Web Application iPhone/News. The orange is also the color of energy. [34]

Primary view

For this view, we had two choices to decide from. The detail information as a Web page, like we can find in the BBC Reader application, see figure 5.9(c), or, basically, all the other studied applications.

We've come up with the second option, a customized detail view with full item information. At this point there's no option for email or comment the new, but is being considered.

Figure 5.18 shows what we've come up with.

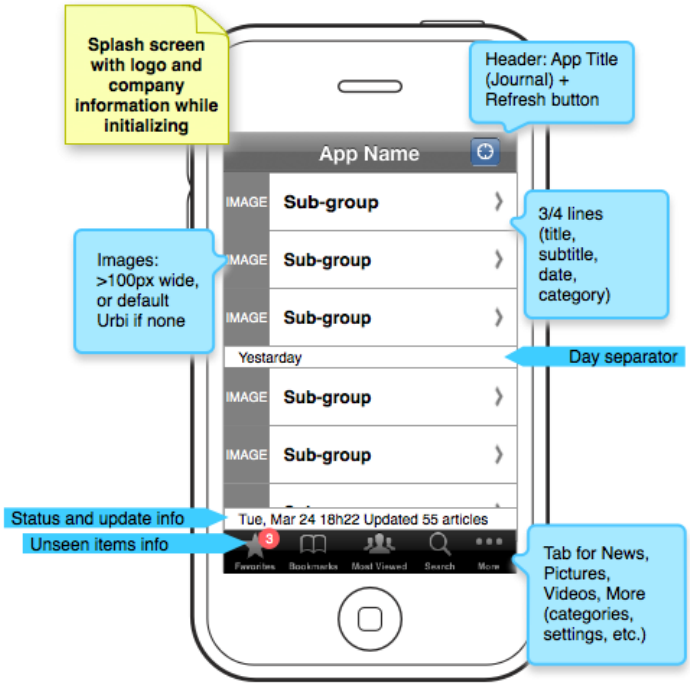


Figure 5.18: Design for the primary view

Detail view

For the detail view there was a lot of issues to decide: How many images? What's the pictures size and quality? What about text size and lettering? Should we include additional features with some navigational and options buttons (ie, send to a friend, email, save, favorite, etc.)? Where can we put those buttons? How about the links and references? These items are presented alone or with small icon with element?

Figure 5.19 shows our decisions and notes for this view.

Settings view

This view was intended to be a project description – info view –, like we have in our Web applications. But, in some earlier development stages it was used for

Figure 5.19: Design for the detail view

device and control information. Some of this information remains there.

We've considered to build a new navigation item to put our user settings, but, as it turned out to be just one option – a slider button –, we decided to put it in the info view and rename it to Settings.

Other views to consider

Some applications have a 'picture view' that shows a slideshow presentation with time, effects and music.

Questions: should the picture be presented with title and text information? What about the all-in-one picture view like a similar page we have in our Web Application (Urbi et Orbi - Mobile/iPhone version)? How many columns, three or four?

Should we present: The most-viewed? Or the most-emailed? Or the most popular thru a five-star-evaluation feature? Can the user define the new as favorite?

Should it have a search button? If yes, should it respond to stored items, Internet items, or both?

Should it save the pictures in cache, or just the new info?

Should it have the commentaries that the online version has? Can the application let users to upload comments?

5.3.4 Development resources

To develop our application we use several tools, provided by Apple's SDK, and information available on the Internet.

In figure 5.20 we can see our Xcode work space with the application on the left, and, on the right the class diagram automatically created by the SDK. With the Xcode comes the debugging tool and the log viewer, which we've used intensively.

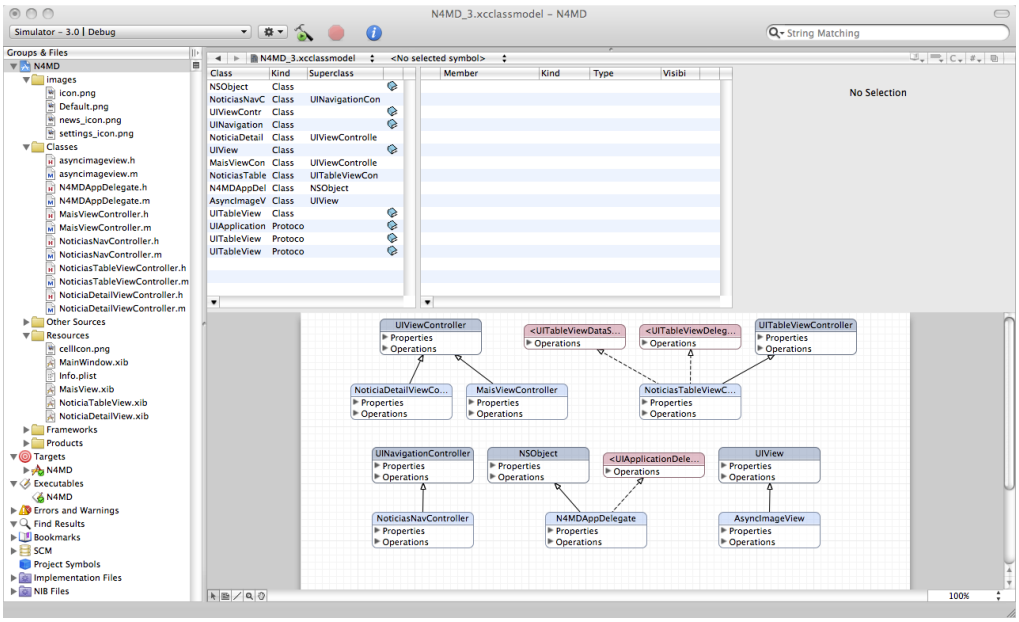


Figure 5.20: Xcode screenshot loaded with the N4MD Application project bundle on the left and UML class model on the right

We've also used the Interface Builder for designing the Views, the iPhone Simulator for testing and running the application, and, the Instruments for real-time monitoring performance and memory.

These tools we're mentioned in previous chapters, but we have to talk about some Web sites that are very useful in this stage. Here's our top sites for iPhone development:

- Apple Development Connection. Documentations and useful examples to download. Forum and online mailing-lists with more specific code. Online: http://developer.apple.com/iPhone;
- Stack overflow follows the question and answer posted by the communities. Allows users to rate the answers. Useful for every language. Online: http://stackoverflow.com;
- Google code has examples, questions and code from other developers. Online: http://groups.google.com/group/iphonewebdev;

- YouTube videos, searching for iPhone or Xcode development did return a lot of good videos for first time developers.

5.3.5 N4MD Application internals

The N4MD application started with the creation of a default Xcode template – The iPhone OS 'Window-base Application'. By doing that all the files and documents were automatic created, and some of them had a few lines of code to 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). Of course, we also had to add more .h and .m class files (NoticiasNavController, NoticiasTableViewController, NoticiaDetailViewController, asyncimageview, MaisViewController), more images for the layout (news_icon.png, settings_icon.png, cellIcon.png) and more Xib files for the Detail View, Settings View and Table View, respectively NoticiaDetailView.xib, MaisView.xib, NoticiaTableView.xib. We can see the bundle structure on the left side of the Xcode image in figure 5.20.

Simplified Functional Model

Figure 5.21 represents the application organization. In this figure we have a representation of all the objects/classes the application needs and the respective views. It's a representation of the model, view, controller paradigm that comes with Apple SDK development approach we've talk about in previous chapters.

Note that figure 5.20 has similar objects but it has a just a model vs controller approach.

Our N4MD Application has six main classes as we can see in figure 5.21. The bottom right class N4MDAppDelegate is responsible for initialization, the 'Navigation Bar', the 'Top Bar', delegation, general attributes and settings (like the user defaults and the network reachability flags). [16]

The grey area called 'View' will be populated by the 'Table View' that we can see associated with NoticiasTableViewController.

As we know, the 'Navigation Bar' in our application has two buttons. *Urbi et Orbi* News and Settings. The correspondent objects for these views are NoticiasTableViewController and MaisViewController.

The object NoticiasTableViewController (Table View Controller subclass) takes responsibility for delegation while this branch of the 'Navigation Bar' is active. This object also manages the sub-view NoticiaDetailViewController objects dynamically created for each 'Table View' item. Although we can see the correspondent elements of this 'Detail View' they are configured dynamically within the code. The visibility, height, position and number of lines are examples of what to change while loading.

In figure 5.22 we can see the correspondent class 'xib' file.

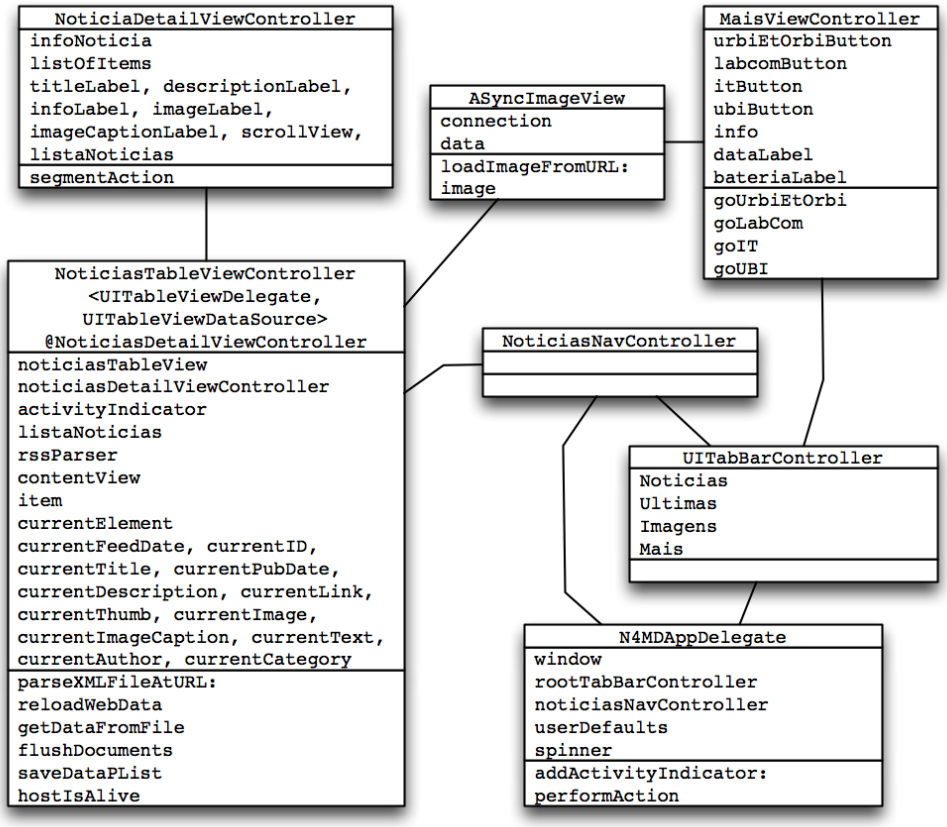


Figure 5.21: N4MD Application class diagram

App life cycle and Initialization

While launching the application, in the 'init' stage gets user preferences and starts to build the window with it's components like the 'Navigation Bar'.

Then, checks if it has a network connection and what type is it. If it has network connection, sends a request for server statistics.

The next stage is about getting the articles. If there is a network connection the application make a XML http request, if not, try to load it from a previous saved [plist] file. Neither case, the application continues by parsing the information.

The display stage happens when the information is loaded. It's when we 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 input that changes state. Then it quits or make display changes again. The next figure 5.23 show a visual representation of the life cycle.

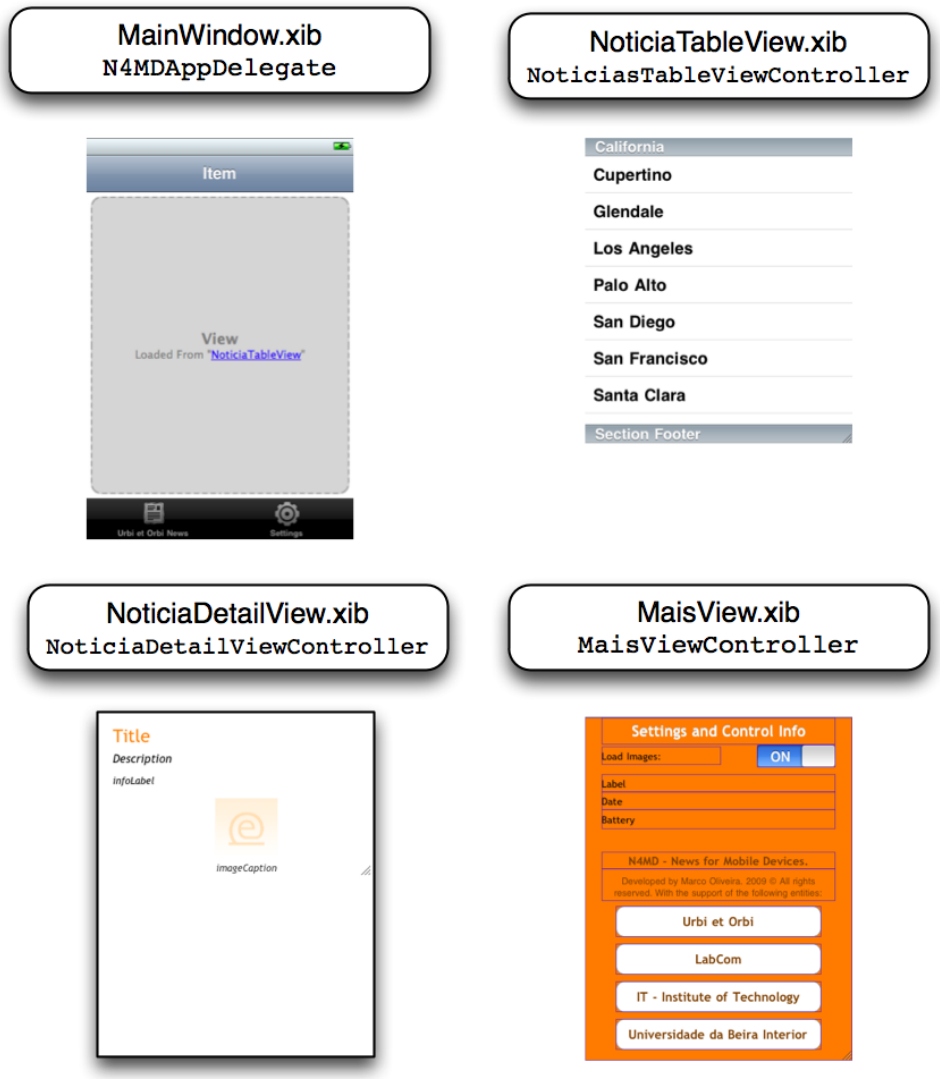


Figure 5.22: N4MD Application 'xib' files, with class association

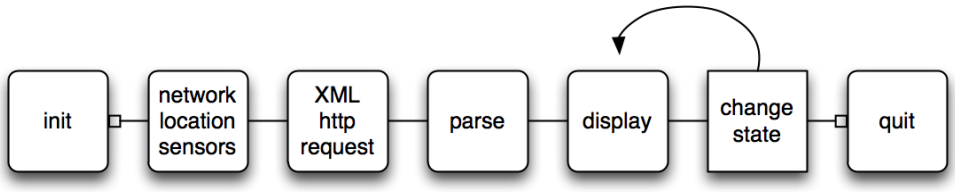


Figure 5.23: Application stages

Requests to server

There's two server communications with information from the application to the server. The first contains information about the device: name, model, localized

model, system name, system version, unique identifier. The second communications occurs when the application makes the request for data. At this point the information to server contains battery status and client type like iPhone.

The other request are motivated by thumbnails and images contained in the XML information. Note that, this request just happens if the user permits image loading on settings, or, if the network type is 3G or Wi-Fi.

Memory, Size, Resources

Apple does insist in optimal memory management and optimized resources. We've tried to do that. Every object allocated has a release instruction in the end, or, it was initiated with an auto-release method.

We do save three variables for user and applications settings along with the applications bundle directory.

We also save the data from the XML http request in a form of a 'plist' file – which is also a XML format. The downloaded images are also saved on the Documents directory.

When we click on the reload button and the flag for Internet connectivity is on, the applications erases all the downloaded data from the bundle directory.

Limitations

When creating the application we've realized some of the limitations that mobile devices have and we have to understand them:

- Screen size forces a top-down design for information display. The 'Table View' is one of the most used components for the iPhone because of it's vertical characteristics;
- Finger vs Mouse limitations. On mobile devices we don't have all the mouse events or buttons that a desktop computer has, for example, 'onMouseOver' or scrollable events;
- Page Zooming isn't available for all views;
- Text size adjustments, options and font types;
- Form controls and form saved information;
- The iPhone recognized some special links like: http:, https:, itunes:, phone:, mailto:. But, some are missing and other mobile devices can't interpret this.

5.3.6 Difficulties

Next, we will explain some of the problems, solutions and notes that we've encountered while developing for this platform.

The main difficulty is this new iPhone platform. It was new for all of us in the team, and we didn't have experienced folks to learn with. We have to learn

it from scratch and in a relatively short period of time. We had to learn the language, the frameworks, the tools, watch the videos, read the manuals and understand the examples.

The Objective-C language. It has very different notation, wile working with objects, when compared with other major languages like C++, Java, PHP.

The iPhone framework documentation that comes with Xcode lacks on examples. Yes, it's available in ADC, but, even so, in the examples provided, it was hard to find a complete and documented one. We have to go to the Internet resources listed before, many times, to find a suitable example to work with a specific object and it's methods. For example, the major and decisive objects in our application were: The 'Navigation Bar' at the bottom, the 'Top Bar' personalization and the XML request and parsing.

Speed improvements. Our application depends of the network speed and number of items to download. It was only when we use threaded url connections, provided by the asyncimageview class, that we've speed up the application. Previously, we were using a personalized caching system that we have develop.

Cache and file management. The turning point here was when we discovered were to save files, but there's some issues like file time stamp or type that we didn't surpass.

To ubiquitously program our application we have to make objects communicate between them. Apple uses a different approach on this subject and wasn't easy to understand at first.

5.4 Differences Between Web Applications and Native Applications

Although we already talk about this in the previous chapter 3. After the real development of these kinds of applications, we would like to add a few things. We can use the iPhone SDK to deploy both kind of these applications, the tools, features or resources we can use – and access –, are different. As we've said the Web Applications was the first way to develop for the iPhone, until Apple released the SDK, and now developers tend to prefer Native Applications because of speed and offline usage, among other reasons.

Table 5.7 shows a more extensible comparison between developing 'Web Apps' and 'Native Apps'. These are technicalities for the developer and not for the final user point of view.

5.4.1 The hybrid alternative

There's an alternative solution, combining an Web Applications and iPhone technologies, to achieve the goal of ubiquitously deliver news to mobile devices.

We though of an iPhone application that could gather the initial device info, then sends a request to a Web Service that creates a Web Application. Here's how it could operate:

Features/Access	Web Applications	iPhone Applications
Installation	"Add to main screen"*	Thru App Store
Access to content	Click on App icon, Open Safari bookmark or insert URL	Click on installed App icon
App Frameworks	Javascripted	Custom
iPhone Frameworks	Limited	Full SDK
Sandbox	Safari sandbox	App sandbox
Cache	Safari cache	Sandbox files
Filesystem	None	Sandboxed (8/16/32GB)
OS Memory	Page shared	iPhone OS (128/256MB)
Customizing	User Web login	App and iPhone Settings
Sensors	Limited	Yes
Location	Yes	Yes
Accelerometer	Limited	Yes
Cocoa Touch	Limited	Yes
Network	Auto	Custom
Multimedia	Embedded	Custom
Other features	Bookmark access	Offline usage; Quit status, SQLLite

Table 5.7: Web Applications vs iPhone Native Application technological differences

* Requires Safari and all the first access procedure. Web page can be configured to disable status and address bar (bottom and top bar).

1. Existence of a Web Service for the delivering of content in the form of a Web Application. Capable of collecting history logs, produce content in a general or dynamic format, and learning skills to intelligently adapt content.

2. Existence of the iPhone application capable of collect location-aware (GPS location, GSM 'fake' location) and context-aware (the mobile device sensor data, characteristics, and user/mobile information – this could be done with the Subscriber Identity Module card (SIM-card or GSM-SIM card) or Home Location Register (HLR) of GSM-networks [38]) information.

3. The application creates a post message (or a XML SOAP messagem with the collected data) and sends it to the Web server. Included in the XMLHttpRequest process.

4. The application quits itself by doing the request because the HTTP/HTTPS request will launch the Safari.app. Note that it is possible to include the safari framework in an application, but, in this case we want to keep it light.

5. The Web server produces the news content accordantly to the information that he received from the post.

6. The Web service delivers the information in a form of Web Application specifically created for the iPhone Safari.app.

7. The browser processes the information, displays de content to the user and wait's for further controller activity.

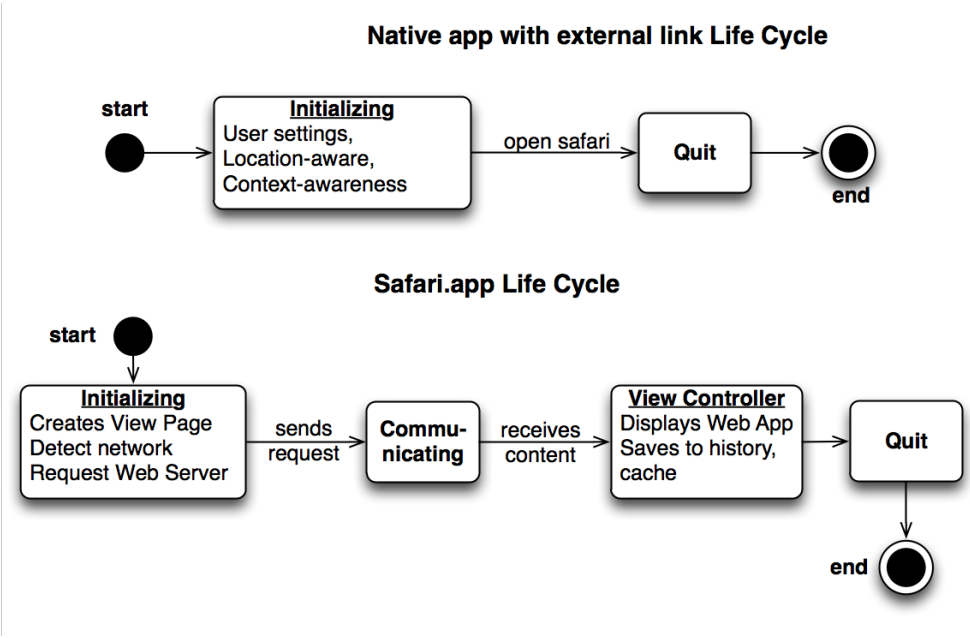


Figure 5.24: iPhone.app and Safari.app UML state in the hybrid solution

Chapter 6

Conclusions and future work

This projects let us realize that we've change our way of doing applications. By doing that, we now use ubiquitous computing techniques in a more effective way.

By addressing fundamental topics, in a quest to unleash the full potential of data consumption, the usage of context-awareness in mobile devices are changing our life quality for the better.

Our systems combines the portability of a Web services, the mobility of users and the limitations of mobile devices to conveniently distribute news information.

Ubiquity

Using resources such as location-aware, context-aware and sensor-aware capabilities, we've improved communications, speed and usability of applications and devices.

Again, the ubiquitous decisions we've come up with native applications are extended to a new level. They also 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.

Partners

The applications we've developed are a significant improvement for the news distribution. We've make the delivering of news seamless, visually effective, configurable (if or when applied) or ubiquitously personal, and, efficient with communications.

The mobile devices is a new way of communications – the seventh way –, it's gaining it's own market, being less equal to small online versions of the bigger brothers, and taking advantage of the opportunities as they pop-up.

The changes for journalist are equally visible, they have to realize that when writing a new, it's going to be readable in big desktops browsers and tiny mobile devices.

The client has a choice to make: to choose were he want's to view the information. Our job, for now, was to make the change friendly – by minimize the user experience shock –, and, reducing the costs in communications, battery life, memory, and speed.

Web and Native Applications

The Web Applications and Native Applications must co-exists due to connectivity issues and offline reading. They are similar in design in many ways, and, they also share architectural and structural models, but, the final purposes of each one, makes them unique and useful on it's own way. The market is still valid and alive for both of them.

For Web Applications we have designed better Internet services. Suited for Web and remote native applications. The Web applications versions now can be viewed in any mobile device without special tweaks, and thats a major leap forward.

We've developed four different versions of our news journal for study purposes. These versions we're created to specifically out stand an important parameter of their target mobile device. The study conducted tested size, connections, design and usability, and, as we expected, each version has it's own advantages. Ultimately, we've decided to let the user choose his preferred version, but, the standard default, to be used with all around mobile devices is the Mobile/iPhone version. This version was developed with the iWebKit, that proved to be, the best all around accessibility platform for general mobile devices Web applications.

The developers goals for the native applications and Web applications were achieved. The application has a solid foundation and ready to grow with the needs of our partners and users.

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

Like the open software, the usage of open code in mobile development is a 'must have'. It opens doors to new applications or services – like the 'open' television without costs – and improves compatibility with other devices. It's all under control of the developer. The application is conceptually simple, it proved to be the best way. The design, directions and the "make it iPhone" philosophy was accomplished by following Apple's iPhone essential guidelines, and, the studied cases of the "competition". We believe that our N4MD application already has public quality.

iPhone

The iPhone has amazing features and we can, practically, do everything we do on a Desktop. But, due to it's mobile characteristics, the iPhone is not a Desktop OS, nor, a near future replacement of it. Mobile devices will be the future of

personal computing, but, for now, the issues like, battery life, user interaction and Internet connectivity, makes it harder to live with on a daily basis.

We've familiarized with the device and the developing technology, lived with it, worked with it and worked for it. We proved to be possible to learn, use and develop in this platform in a few months time. Of course, it would be harder without the developers community and their tips. Other mobile devices have, equally, an interesting mobile mindset, but, the iPhone platform has a solid ground.

The iPhone Software Development Kit is an amazing tool for developers. It's also friendly to new comers with it's intuitive design and philosophy. Of course, Apple users oriented marketing model provides documentation and some examples to get you started, but, it's also easy to get help on the Internet: outside documentation, code examples, books, videos, tutorials, and, even attend to american university classes for science computer students. All this makes the iPhone platform learning experience less damaging.

We are eager to test the acceptance of our N4MD Application after deploying it in App Store. It's a world wide market with millions of experienced and exigent users. We are eager to be a part of this quiet revolution in the mobile market. We still have a few things to improve in our application and there are copyright license issues to resolve before putting it available. For now, we are in the second line of the battle, but, we sense, soon we will be in the front line, and, with an eye on the future of mobile devices.

The world needs news and we studied how to deliver them. With our ubiquitous computing, our software design engineering, our service architecture, our news content, our parters and clients, we brought innovation, with this project, to this new world order. The world where everybody is going to be one day. The world of mobile devices. Therefor, News For Mobile Devices – N4MD.

Future Work

Despite the our we've put up in our applications, there's a lot we can do in the future to improve them. Some of the following items we're suggested during tests and debugging stages. Other topics are already implemented in other applications we've studied, and, it seems worthy to make it for our application.

The following topics are divide by type of application we've developed for this project: 'Web Applications' or 'Native Applications'.

Web Applications

1. Advanced client information, client history, and online statistics on the Web Applications and on Web Services. By integrating the ubiquitous system with this new features, we will be able to create more personal, seamless and user oriented news content;

2. Add multimedia capabilities to the Web application and Web server (streaming server). Add video and audio with automatic conversion on codecs, containers, sizes and compatible formats for the iPhone and other mobile devices.

3. Client 'login' for specific content; Capabilities to post comments, news, suggestions, and images; Slideshows and other Safari for iPhone Web features;

4. Mobile version of the back-office for journalists or groups with privileges. Forms for posting, uploading video, audio, images and info-graphics. Or, simply, allowing to embed external files like YouTube[1] or Flickr[2];

5. Add options like: Send content to email; Add to Twitter[3], Facebook[4], Hi5[5] or other social networks.

Native Applications

Some of the items listed before, in Web Applications group, for example items 1 and 5, can also be applied, on the server side of the N4MD Native application, for future improvements.

1. Location-awareness given by GPS or DNS reverse lookup to ubiquitously choose news, language and design for the specific region;

2. Time-awareness could be a configurable feature to improve application design (by altering contrast regarding the hour of the day), news content (change the size of text in late hours when the eyes are tired, the amount of text is smaller during work hours, or display images if the last update was minutes ago), and, cached items (an automatic cache cleaner that measures the time life of the file and swaps it);

3. Improve native application ubiquity by using more information from sensors, location services, settings, connectivity and iPhone specific model device characteristics;

4. Add more content, application information and status, categories, services and multimedia capabilities, to the native application and also to the server side.

5. Embed video, audio, photo slideshow and other Web pages without quitting from the application;

[1] YouTube: http://www.youtube.com/
[2] Flickr: http://www.flickr.com/
[3] Twitter: http://twitter.com
[4] Facebook: http://www.facebook.com
[5] Hi5: http://www.hi5.com

6. Improve software design. “Make it iPhone”, by bringing innovation on design, more information on display, new features or services, usability or accessibility.

7. Optimize: Application speed by threading processes; Wireless connections and connectivity by compressing packets; Memory management by cacheing more items; Battery life time by testing with different ubiquitous configurations; ubiquitous computation by using artificial intelligence and statistical information from others users usage.

References

[1] Ahonen, Tomi T., and Moore, Alan, "Deeper insights into the 7th Mass Media channel, mobile is to the internet, what TV is to radio", in *Communities Dominate Brands*, 2009, pp. 1-12. [Online] Retrieved on May 11, 2009: http://communities-dominate.blogs.com/brands/2008/05/deeper-insights.html

[2] Allen, Christopher, and Appelcline, Shannon, "iPhone in Action: Introduction to Web and SDK Development", Manning Publications Co., Greenwich, 2009.

[3] Allsopp, John, "iPhone native Apps – the great leap backwards?", *Web Direction*, July 20, 2008. [Online] Retrieved on July 17, 2009: http://www.webdirections.org/blog/iphone-native-apps-the-great-leap-backwards/

[4] Altenberg, Bert, and Clarke, Alex, and Mougin, Philippe, "Become and Xcoder: Start Programming the Mac Using Objective-C", CocoaLab, 2008.

[5] Apple Inc., "Apple - iTunes - Thanks a Billion", 2009. [Online] Retrieved on April 25, 2009: http://www.apple.com/itunes/billion-app-countdown

[6] Apple Inc., "Apple Developers Site", 2009. [Online] Last visited on August 2009: http://developer.apple.com

[7] Apple Inc., "Apple Reports First Quarter Results", January 21, 2009. [Online] Retrieved on February 12, 2009: http://www.apple.com/pr/library/2009/01/21results.html

[8] Apple Inc., "Code Signing Guide: Security", Apple Inc., California USA, November 19, 2008.

[9] Apple Inc., "Core Data Tutorial for iPhone OS: Data Management", Apple Inc., California USA, June 6, 2009.

[10] Apple Inc., "Dashcode User Guide", Apple Inc., California USA, March 4, 2009.

[11] Apple Inc., "Data Management Coding How-To's", May 7, 2008. [Online] Retrieved on March 27, 2009: https://developer.apple.com/iphone/library/codinghowtos/DataManagement/

[12] Apple Inc., "Dynamic HTML and XML: The XMLHttpRequest Object", June 24, 2006. [Online] Retrieved on May 9, 2009: http://developer.apple.com/internet/webcontent/xmlhttpreq.html

[13] Apple Inc., "iPhone Application Programming Guide", Apple Inc., California USA, January 6, 2009.

[14] Apple Inc., "iPhone Development Guide", Apple Inc., California USA, May 28, 2009.

[15] Apple Inc., "Optimizing AJAX for iPhone", February 3, 2008. [Online] Retrieved on May 09, 2009: http://developer.apple.com/safari/articles/optimizingajax.html

[16] Apple Inc., "SCNetworkReachability Reference", Apple Inc., California USA, March 5, 2009.

[17] Apple Inc., "Table View Programming Guide for iPhone OS", Apple Inc., California USA, October 15, 2008.

[18] Berners-Lee, Tim, "The World Wide Web: Past, Present and Future", *IEEE Computer*, August, 1996. [Online] Retrieved on May 20, 2009: http://www.w3.org/People/Berners-Lee/1996/ppf.html

[19] BBC News, "Mobile internet usage on the rise", *BBC News Techonology*, November 25, 2008. [Online] Retrieved on February 6, 2009: http://news.bbc.co.uk/2/hi/technology/7748372.stm

[20] CIA - Central Intelligence Agency, "Portugal", *The World Factbook*, 2009. [Online] Retrieved on February 2009: http://www.cia.gov/library/publications/the-world-factbook/geos/po.html

[21] Cohen, Peter, "2008 in review - Mac developers who made news", *Macworld.com*, December 31, 2008. [Online] Retrieved on January 21, 2009: http://www.macworld.com/article/137835/2008/12/2008developers.html

[22] comScore Inc., "comScore Reports 6.5 Million Americans Watched Mobile Video in August", October 31, 2008, [Online] Retrieved on May 20, 2009: http://www.comscore.com/press/release.asp?press=2558

[23] Dalrymple, Jim, "2008 in review - Apple", *Macworld.com*, December 31, 2008. [Online] Retrieved on January 21, 2009: http://www.macworld.com/article/137821/2008/12/apple.html

[24] Dalrymple, Jim, "App Store tally: 20,000 apps in seven months", *Macworld.com*, Feb 10, 2009. [Online] Retrieved on February 12, 2009: http://www.macworld.com/article/138753/appstore.html

[25] Dalrymple, Jim, "Apple unveils iPhone SDK", *Macworld.com*, March 6, 2008. [Online] Retrieved on January 21, 2009: http://www.macworld.com/article/132400/2008/03/iphonesdk.html

[26] Drance, Matt, and Jurewitz, Michael, "Introduction to Objective-C and Cocoa Touch", Apple Developer Connection Video, *iPhone Tech Talk World Tour*, 2009.

[27] eMarketer Inc., "How Do Boomers Use Their Mobiles?", December 9, 2008. [Online] Retrieved on February 12, 2009: http://www.emarketer.com/Article.aspx?id=1006792

[28] INE – Instituto Nacional de Estatística, "A Sociedade de Informação em Portugal – 2008", *Sociedade de Informação em Portugal*, 2009. [Online] Retrieved on August 2009: http://tinyurl.com/m4cvhn

[29] Etemad, Elika, and Newth, Jorunn D., "Pocket-Sized Design: Taking Your Website to the Small Screen", in *A List Apart*, issue 187, August 31, 2004. [Online] Retrieved on January 9, 2009: http://www.alistapart.com/articles/pocket/

[30] Forrest, Brady, "iPhone's Location-Aware Apps", *O'Reilly Radar*, July 14, 2008. [Online] Retrieved on January 22, 2009: http://radar.oreilly.com/2008/07/iphone-location-aware-apps.html

[31] Foy, Core, "Review of: Beginning iPhone Development", *Slashdot.org*, January 19, 2009. [Online] Retrieved on January 22, 2009: http://books.slashdot.org/article.pl?sid=09%2F01%2F19%2F1451236

[32] Gartner, Inc., "Gartner Says Enterprise Mobile Phones Will Replace Desktop Phones in North America by 2011", February 4, 2009. [Online] Retrieved on May 20, 2009: http://www.gartner.com/it/page.jsp?id=874012

[33] Guimarães, Luciano, "Duas ações negativas da cor-informação no design de notícias", In *Communicare*, Vol. 5, Nº 1 - 1º sem., 2005.

[34] Guimarães, Luciano, "O repertório dinâmico das cores na mídia: Produção de sentido no jornalismo visual", Grupo de Trabalho *Produção de sentido nas mídias*, XV Encontro da Compós, na Unesp, Bauru, SP, Junho 2006.

[35] Hazaël-Massieux, Dominique, "Return of the mobile style sheet". *A List Apart*, issue 275, January 6, 2009. [Online] Retrieved on January 9, 2009: http://www.alistapart.com/articles/returnofthemobilestylesheet

[36] IEEE Standards Activities Department, "2009 IEEE Standards Style Manual", IEEE Standards, 2009.

[37] Innovative Phones, "Mobile (Cellular) Phones", 2009. [Online] Retrieved on May 20, 2009: http://www.innovativephones.com/

[38] Johansen, Tor Anders, and Jørstad, Ivar, and Thanh, Do van, "Identity management in mobile ubiquitous environments", *Internet Monitoring and Protection*, 2008 - ICIMP '08, IEEE Computer Society, Pages 178-183.

[39] Mark, Dave, and LaMarche, Jeff, "Beginning iPhone Development: Exploring the iPhone SDK", Apress, New York, 2009.

[40] McFedries, Paul, and Pabian, David, "iPhone 3G Portable Genious", Wiley Publishing, Inc., Indianapolis Indiana, USA, 2008.

[41] Mostefaoui, Soraya Kouadri, and Maamar, Zakaria, and Giaglis, George M., "Advances in Ubiquitous Computing: Future Paradigms and Directions", IGI Publishing, New York, 2008.

[42] Nielsen Company, The, "Mobile Internet Extends the Reach of Leading Internet Sites by 13%", Nielsen Media Research, May 1, 2008. [Online] Retrieved on February 12 2009: http://tinyurl.com/rdam5m

[43] Openmoko, "Open. Mobile. Free. – Main Page", *Openmokowiki*, 2009. [Online] Retrieved on February 2009:
http://wiki.openmoko.org/wiki/Main_Page

[44] O'Relly, Tim, "What is Web 2.0", September 30, 2005. [Online] Retrieved on May 20, 2009:
http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html

[45] Pash, Adam, "How your location-aware iPhone will change your life", *Lifehacker.com*, June 5, 2008. [Online] Retrieved on February 12, 2009:
http://www.lifehacker.com/395171

[46] Pogue, David, "iPhone: The missing manual", O'Reilly Media, Inc., Second Edition, 2008.

[47] Romero, Álvaro, "El 'software' libre permitirá elegir la manera de usar el móvil", *Elpais.com*, January 21, 2009. [Online] Retrieved on January 22, 2009: http://tinyurl.com/9yljas

[48] Sadun, Erica, "The iPhone developer's cookbook: Building mobile applications with the iPhone SDK", Addison-Wesley/Pearson Education, Inc., Boston USA, 2009.

[49] Wagner, Richard, "Professional iPhone™and iPod®Touch Programming: Building Applications for Mobile Safari™", Wiley Publishing, Inc., Indianapolis, Indiana, 2008.

[50] Weiser, Mark, and Brown, John Seely, "The coming age of calm technology", Xerox PARC, October 5, 1996. [Online] Retrieved on February 05, 2009: http://www.ubiq.com/hypertext/weiser/acmfuture2endnote.htm

[51] Yahoo Developer Team, "Best Practices for Speeding Up Your Web Site", Yahoo Developer Network, 2009. [Online] Retrieved on May 22, 2009: http://developer.yahoo.com/performance/rules.html

[52] Watershed Publishing, "Top 25 iPhone Apps Dominated by Games, SocNets", Watershed Publishing/comScore, February 2009. [Online] Retrieved on May 25, 2009: http://www.marketingcharts.com/interactive/top-25-iphone-apps-dominated-by-games-socnets-8629/comscore-apple-apps-iphone-top-site-categories-users-february-2009jpg/

[53] Wikipedia Foundation Inc., "Mobile phone", February 6, 2009. [Online] Retrieved on February 6, 2009: http://en.wikipedia.org/wiki/Mobile_phone

[54] Zdziarski, Jonathan A., "iPhone Open Application Development", O'Reilly & Associates, Inc., California, USA, March 15, 2008. by

[55] Zilona, Stephen J., and Ketha, Sai Sanjay, "Think inside the Box! Optimizing Web Services Performance Today", *IEEE Communications Magazine*, 2008, Pages 112-117.