m-WOnDA: The "*Write Once 'n' Deliver Anywhere*" model for mobile users

Dionysios G. Synodinos[1], Paris Avgeriou[2]

[1]National Technical University of Athens, Network Management Center,
Heroon Polytechniou 9, Zografou 157 80, Greece,
dsin@noc.ntua.gr

[2]National Technical University of Athens, Software Engineering Laboratory,
Heroon Polytechniou 9, Zografou 157 80, Greece, Tel: +3010-7722487, Fax: +3010-7722519
pavger@softlab.ntua.gr

Abstract. The design and development of hypermedia applications that are deployed on the Wireless World Wide Web is a daunting task because of the various available platforms and the restrictions and capabilities imposed by the lack of established standards and the exponentially increasing number of emerging platforms. There are now justifiable research and development efforts that attempt to formalize the engineering process of such applications in order to achieve certain quality attributes like modifiability, maintainability and portability. This paper presents such an attempt for designing a conceptual model for hypermedia applications that allows for easy update and alteration of its content as well as its presentation and also allows for deployment in various mobile platforms. In specific this model explicitly separates the hypermedia content from its presentation to the user, by employing XML content storage and XSL transformations. Our work is based upon the empirical results of designing, developing and deploying hypermedia applications for mobile platforms, and on the practices of well-established hypermedia engineering techniques.

1 Introduction

During the last years there has been a lot of hype about the wireless World Wide Web (W4), where hypermedia applications can be accessed by wireless clients and especially mobile users. Emerging technologies and the penetration of mobile computing in our daily lives, have led to the need for mobile access of common web sites. The technology behind the mobile access to the Web usually draws upon a subset of HTML, like the cHTML (compact HTML) of NTT DoCoMo's i-mode with its millions of users in Japan, or is XML-derived like WML (Wireless Markup Language), the markup language of WAP, heavily deployed in Europe and abroad.
The anticipation of faster and cheaper W4 (Wireless World Wide Web) that the 3rd generation (3G) wireless networks and more sophisticated mobile devices will bring, results in a growing number of organizations that plan to deploy parts of their traditional web sites for mobile users. This task though can be overwhelmingly difficult since several problems have derived in building and maintaining hypermedia applications for mobile users.
To begin with, there is a vast demand for the adaptation of content into a growing number of presentational templates, each one of them suitable for a different device. Only in the case of

WAP-enabled phones, the developer must provide numerous templates that comply with the potential user's device capabilities and restrictions. These can be device characteristics like the screen size, e.g. a site should be deployed in a different way for a Nokia 7110 with its 96x46 pixels screen and for an Ericsson RS with its generous 360x120 pixels display. Also resolution and available bandwidth is an issue. For instance the site author should make provisions for slow GSM access, faster GPRS networks or even lightning fast 3G and beyond. Furthermore, the input peripherals bring in another degree of freedom, ranging from the standard numerical pad to Nokia 's 9210 PC-like keyboard. To make matters worse, mobile clients might need different versions of web pages as a side effect of the different level of support for WMLScript, the WAP client-side scripting language. In situations like these, updating content and performing version control can become tremendously resource-consuming if not impossible.
On top of everything else, web pages have evolved to a point of becoming too complex with all the inline client scripts and styles rules in order to facilitate the ever-growing and often conflicting demands for enhanced usability and impressive 'look and feel'. Therefore the daily task of updating content can no longer be performed by a novice in markup languages, but instead designated professionals with a solid background on web authoring and a clear understanding of the architecture of the certain site must be utilized.
The World Wide Web Consortium had an early provision to such problems with the launch of XML and the related family of technologies, in order to separate the content from the rest of the information such as presentation rules, metadata, active components etc. The question now is, given the XML technology, how can a site be engineered in order for it to achieve modifiability, maintainability and portability. In specific, the problem that hypermedia application authors have at hand, is comprised of the following secondary problems:

- How can one maintain the site content by updating it, at will, without requiring him or her to master the underlying technology of presentation style sheets, client-side scripts etc. for the target mobile platforms?
- How can one modify the layout, presentation and active components of the site without affecting the content associated with them?
- How can one port the hypermedia application to alternative versions for existing mobile platforms and still make provisions for future delivery platform versions?

In this paper we attempt to solve the above problems by proposing an XML-based multi-tier model, which is established upon the separation of the actual content, active widgets, presentation rules and the page generation process. This model is a refinement of the **WOnDA** (**W**rite **On**ce, **D**eliver **A**nywhere) model [1] for mobile clients. The presentation rules are themselves separated into a set of rules for transforming the actual content, the various widgets and the layout of the pages for every one of the presentational domains. The proposed model is based on an XML repository that holds the actual (textual) content and utilizes the power of eXtensible Style Sheet (XSL) transformations in a hierarchical manner that is well suited for providing a rich set of formats for every page to be deployed in. It also facilitates easy administration, the ability to easily add new formats and fast/clear refactoring of old ones. Also, since there is a complete separation between data and presentation, the development of content can become a streamlined process that doesn't deal with the complexity of the underlying structure of the chosen presentational domains.

The structure of the rest of this paper is as follows: Section 2 introduces a short literature review, comprised of the most significant approaches in designing hypermedia applications for several presentational domains and mobile clients. Section 3 analyses the proposed model and its philosophy. Section 4 presents a case study of a demo site that is created with the aid of the model and deployed in four mobile clients. Finally section 5 wraps up with ideas for future work.

2 Literature Review

There is already work in progress by the World Wide Web Consortium, under the title of **Composite Capabilities/Preference Profiles** (**CC/PP)** [2] towards the direction of serving the same content to different clients according to their profile. In particular, the CC/PP framework aims to provide a common way for clients to express their capabilities and preferences to a server that originates content. The server then uses this information to adapt the content in a way appropriate for the client device. This approach is particularly valuable for the case of mobile devices where the variety of clients is overwhelming. The WOnDA model can be integrated with the CC/PP model in the manner that different presentation formats can be created in an asynchronous way to satisfy the device profiles collected from a central or various distributed profile registries. On the other hand WOnDA proposes that creation of hypermedia context is done prior to user request (static content), and has no provision for dynamic adaptation of content according to the capabilities of the device by reading the device's profile in real time. Such a scenario though will be examined in future additions of WOnDA in order to serve the needs of a database-driven website where dynamic page generation from queries to a database is a requirement.

Another important approach of serving the same content to different clients is **Several Interfaces, Single Logic (Sisl)** [3]. This is an architecture and domain-specific language for designing and implementing services with multiple user interfaces. It aims to the decoupling of interface from service logic, by employing an event-based model of services that allows service providers to support interchangeable user interfaces to a single source of service logic or data. The need to provide content in a hypermedia format to a mobile client can be satisfied using the Sisl approach, in the sense that a web site can be modeled and developed using standard Sisl architecture, so that it can later facilitate interchangeable user interfaces. In fact, Sisl targets a wider variety of services and deals in greater depth with issues like how to manage the fundamental differences in the nature of interaction across the spectrum of user interfaces. As a result it doesn't elaborate on the actual process of creating the hypermedia documents. Also the broader spectrum of applications that Sisl aims to, causes it to grow in complexity, making its use in smaller projects, an overkill.

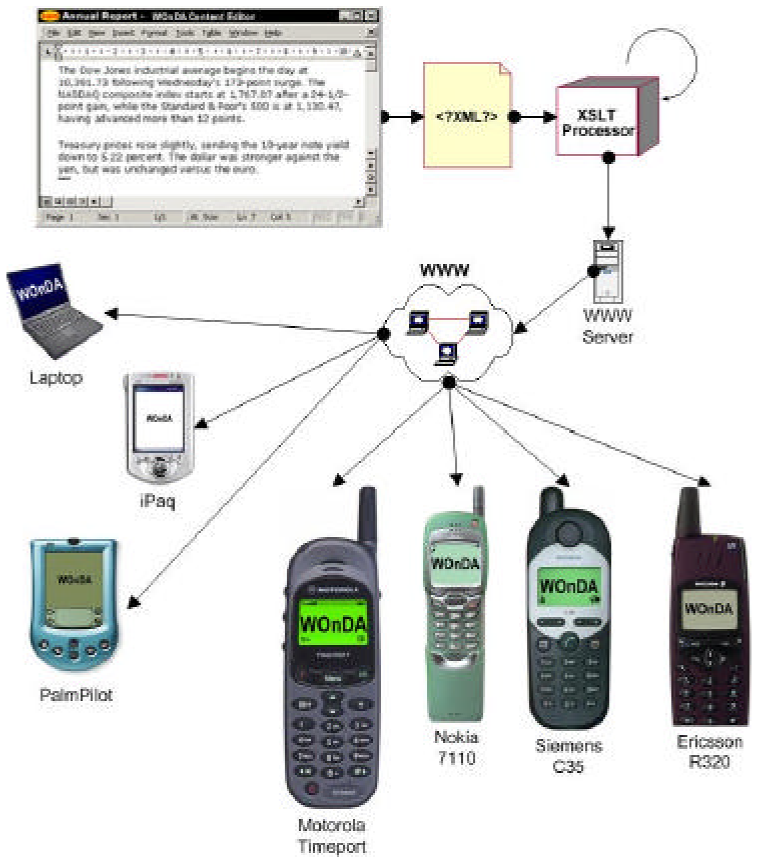
The IBM's T.J. Watson Research Centre has established a vision, named **Platform-Independent Model for Applications (PIMA)** [4]**,** http://www.research.ibm.com/PIMA/) about next generation pervasive computing that can be described by the following three dimensions: use of mobile computing devices, creation/deployment of applications to such clients, and the environment and how it is enhanced by the emergence and ubiquity of new information and functionality. WOnDA is actually oriented towards the second dimension and especially on the deployment of hypermedia applications and it elaborates on the "*design-time*" part of the proposed application model of IBM. Both approaches suggest that applications are not written with a specific device in mind. So the developer should not make any assumptions about the client's device restrictions and capabilities in a manner that restricts the applications functionality, thus resulting to the fact that the task logic should not be secondary to the user interaction. Since the model proposed by IBM requires that application description captures the purpose of the user interaction at a high level, the user interface definition should include a decomposition of the interaction that is driven by the definition and structure of the user tasks. Because the above model should also be context-aware, the developer should not make assumptions about the services available and so these services should not be explicitly named, but rather specified in an abstract manner. This is in contrast with the WOnDA model for hypermedia applications where the application is deployed asynchronously into a format desired by the target clients, where the set available services, restrictions and capabilities are well defined and available to the designer at design-time. Also in the WOnDA universe there is no need for a specific client-side adaptation of content, which means that clients with smaller footprints can be utilized

It is also important to mention the **User Interface Markup Language (UIML)** [5] that provides a high device-independent method to describe a User Interface (UI). UIML looks at any UI from six orthogonal dimensions: the parts comprising the UI, the presentation of these parts, the content, the behaviour, the mapping of UI controls in some domain (e.g. HTML) and the business logic connected to this UI. It is implemented as a declarative XML-compliant meta-language that allows for the implementation of many UIs without learning the language or API specific for that device. The same philosophy is adopted by WOnDA, which supports content creation without mastering the underlying technology of any of the supported platforms. UIML's latest version supports multi-modal UI that can be used simultaneously and kept synchronized. For example a UI might offer a voice and a screen based front-end and the user can at any time switch between the two interaction modes. It supports a variety of supported domains (not only Markup Languages) and provides a canonical representation of any UI that is suitable for mapping to existing languages. Due to its view of a UI as a tree whose parts can change dynamically, it is well suited for applications that need to be multilingual.

A final interesting approach for mobile access to hypermedia applications is presented at [6], where the authors advocate the need for an automated approach to the adaptation of the User Interface that uses artificial intelligence and statistical techniques. The adaptation model proposed is incremental and iterative and focuses on the user's interests as depicted by his actions and not by explicit content rating. In this way the system learns from the user's preferences and adapts the UI in a way that provides an easy access to a vast amount of information. This approach is still at a very theoretical level and remains to be tested in real-world applications.

3 The model

Before describing the WOnDA model in detail, it is useful to examine the way this model is used for writing content once and delivering it to multiples clients. An overview of the model 's modus operandi is illustrated in Figure 1. The content is authored in a text editor that provides XML authoring facilities in a transparent way to the author. This could be implemented as a MS Word plug-in [YAWC Pro, http://www.yawcpro.com/], an ActiveX component [XMLSpy document editor browser plug-In, http://www.xmlspy.com/download_plugin.html], or a different editor [XMLSpy IDE, http://www.xmlspy.com/products_ide.html]. This means that content can be created and updated by people with no web-authoring background, by letting them write in simple text and have it automatically converted to XML. In sequence, XSL transformations are utilized to impose style rules and presentation layout, add active objects, and generate the page in its final form, e.g. WML page, a handheld-compatible page etc. The final page is then published to a Web Server and served to the appropriate clients through the Internet.


The Dow Jones industrial average begins the day at 10,361.73 following Wednesday's 173-point surge. The NASDAQ composite index starts at 1,767.07 after a 24-1/2-point gain, while the Standard & Poor's 500 is at 1,130.47, having advanced more than 12 points.

Treasury prices rose slightly, sending the 10-year note yield down to 5.22 percent. The dollar was stronger against the yen, but was unchanged versus the euro.

Figure 1 - A macroscopic view of the model

What is the mechanism that deals with the XML files and XSL Transformations that translate raw content into a specific delivery platform? What are these XML files, how do the XSL transformations take place and what does the final result look like? These questions will be answered in the remainder of this section in the form of a guide for the construction of maintainable, modifiable, and portable hypermedia applications for mobile clients.

In order to describe this mechanism we propose a conceptual model by utilizing the Unified Modeling Language [7] (http://www.rational.com/uml), a widely adopted modeling language in the software industry and an Object Management Group standard [http://www.omg.org/]. Furthermore in order to define the syntax and semantics of the conceptual model we have designed a UML meta-model, i.e. a model that defines the language for expressing the conceptual model [8].

The conceptual model described here considers only static hypermedia pages and every page is comprised of the following elements:

1. The actual content of the page that consists of text, hyperlinks, images, videos, animation etc.
2. A set of navigational or promotional active objects or widgets like navigation bars, search boxes, menus, logos, ads, banners etc.
3. The general layout of the page meaning the positioning of all the above in the browser window and the rest of the markup envelope that is needed in order for the page to be syntactically valid.
4. Hyperlinks to other hypermedia pages

It is noted that this is a simplified and superficial model of a hypermedia page because the aim of WOnDA is not to model hypermedia applications in general but merely to separate content from the rest of the information and generate multiple versions of hypermedia applications. In other words the proposed model is considered to be in a lower abstraction layer than usual hypermedia design models such as HDM [9], RMM [10], WebML [11] etc.

We now move on to specify the meta-model that will define the language for expressing the conceptual model. The principles of the meta-model are the following:

1. The actual content (text, links, references to media files) of each hypermedia page is kept in one XML file. These files will be referred to as **Page Contents** (PCs). PCs represent published pages as abstract data entities without taking into account any presentation aspects derived by the desired formats.
2. The task of providing content rendering information is left up to a set of XSL files, which will be referred to as **Content Transformers** (CTs). The idea behind CTs is that if we define a set of N versions for the site under construction, every version is exactly identical to all the others in terms of textual information since this information is provided by the PCs, but the versions differ in the layout, functionality, style and the markup that they're written in. For every one of the versions we define a CT which describes the rules necessary to transform the content provided by the respective PCs.
3. In the fashion of PCs and CTs for the textual content we define **Page Widgets** (PWs) and **Widget Transformers** (WTs), which hold the necessary data and presentation rules respectively for the widgets used.
4. Metadata that are specific to the content page, as in the WML, cHTML, HTML <META>

element, used throughout a version or even throughout the entire site are kept in an XML file, which will be referred to as **Content-Specific Metadata** (**CSM**).

5. Metadata that are specific to a certain version, e.g. character encoding information, are kept in another XML file, called **Version-Specific Metadata (VSM).**
6. Both content page-specific and version-specific metadata are rendered by another transformer XSL file called **Metadata Transformer (MT).**
7. For every one of the different versions we define an XSL file, which describes the rules necessary to generate the page layout that is restricted in the context of the version. These XSL files, which will be referred to as **Version Builders (VBs)**, do not contain any information about the rules we need to render the textual content drawn from the PCs nor the widgets used. They rather define the general layout of the page meaning the positioning of all the above in the browser window and the rest of the markup envelope that is needed in order for the page to be syntactically valid for the corresponding presentational domain. The information about client-side scripts or additional client-side style rules (e.g. CSS), are referenced by the VB or included in it depending on the capabilities of the syntax of the relevant domain. For example for the HTML domain this can be accomplished by the <LINK> element.

Figure 2 depicts the relationship between the above model elements. Page Contents, Page Widgets and Content-Specific Metadata are XML files and are all specializations of the class "Generic Hypermedia Page Element". They are also connected with an aggregation relationship with the "Hypermedia Page" class, which means that they are all part of a hypermedia page. Content Transformer and Widget Transformer are XSL files that render the corresponding Page Contents and Page Widgets. Furthermore it is obvious that Content-Specific Metadata are related to Page Contents, in the sense that metadata describe the content. Moreover, Content-Specific Metadata and Version-Specific Metadata are rendered by the Metadata Transformer. Finally Version Builder uses all the other transformers to render the layout of the hypermedia page and insert the appropriately transformed content, widgets and metadata into the final version-specific hypermedia page. Content Transformers, Widget Transformers and the Version Builder are specializations of the "Version-Specific Transformer" class. Finally it is noted that the names of the classes "Generic Hypermedia Page Element", "Hypermedia Page", "Version-Specific Transformer" are written in italics, since they are abstract classes in this meta-model.

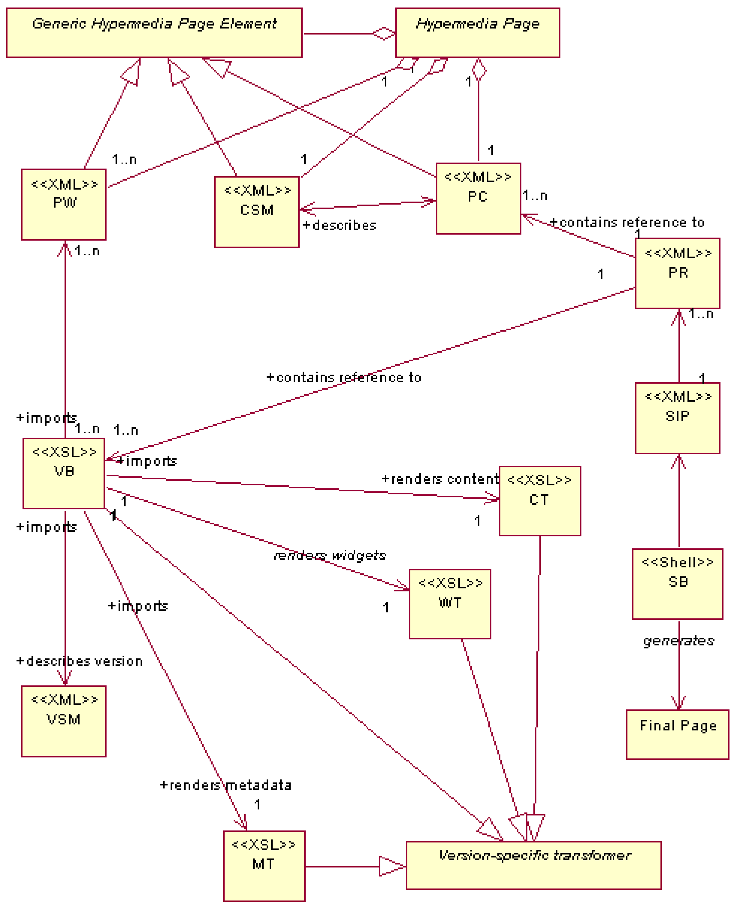


Figure 2 – the hypermedia page elements and the transformers

8. For every page there is a registry specific for it, called **Page Registry (PR),** which link together all the page elements and their transformers for the various versions.
9. For every site there is a main registry, the **Site Index Page (SIP)**, which holds all the file system (or network) paths to the Page Registries.
10. All the above are parsed by a processing shell, which will be referred to as ***Site Builder*** and provides the web site administrator with a web interface to generate or update certain pages, entire versions of the site etc.

The Page Registry gathers all the necessary data from the Page Content, the Content Transformer, the Content-Specific Metadata and the Version Builder. The Site Builder parses the Site Index Pages to look for all the Page Registries, performs the transformations and generates the hypermedia page of the appropriate format.

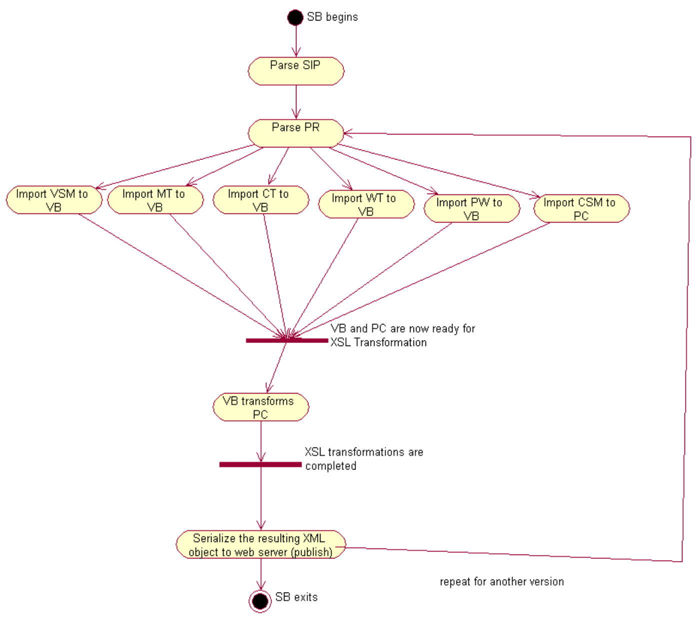

SB begins
Parse SIP
Parse PR
Import VSM to VB
Import MT to VB
Import CT to VB
Import WT to VB
Import PW to VB
Import CSM to PC
VB and PC are now ready for XSL Transformation
VB transforms PC
XSL transformations are completed
Serialize the resulting XML object to web server (publish)
repeat for another version
SB exits

Figure 3 – an activity diagram of the Site Builder functionality

The last thing to be clarified about WOnDA is the way that the model is applied. A simple process model for harnessing WOnDA is depicted in Figure 3, as a UML activity diagram, showing the discrete activities as well as the artifacts that activities produce or take as input.

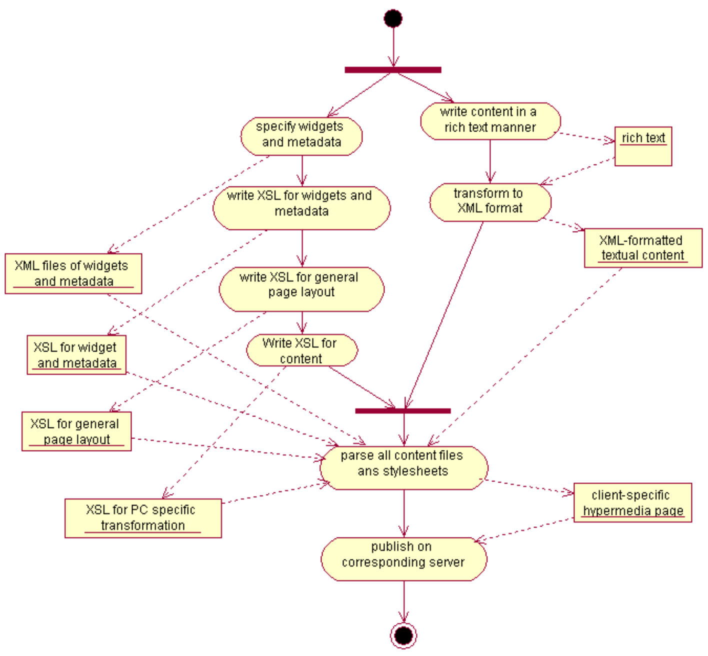

specify widgets and metadata
write content in a rich text manner
rich text
write XSL for widgets and metadata
transform to XML format
XML-formatted textual content
XML files of widgets and metadata
write XSL for general page layout
XSL for widget and metadata
Write XSL for content
XSL for general page layout
parse all content files ans stylesheets
XSL for PC specific transformation
client-specific hypermedia page
publish on corresponding server

Figure 3 - a process model for applying WOnDA

4 Mobile City Guide of Athens: A CASE STUDY

In order to examine the feasibility and effectiveness of WOnDA in a real world scenario, a demo website has been developed. The purpose of this site is to provide tourists that visit Athens for the 2004 Olympic Games and have mobile Internet access, with information about the city, the Games and the various events. The aim of our effort was to provide custom access to both textual information and images e.g. maps, for a certain number of predefined mobile devices. The scenario described by the following pictures is one where 4 users with 4 different mobile clients, access the home page of the site and choose to browse a map of the city.

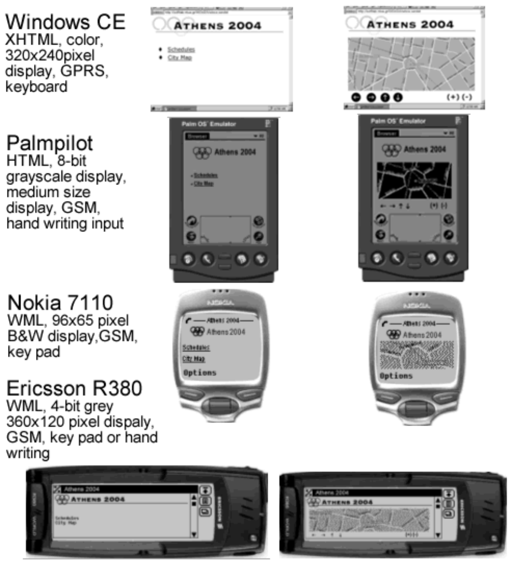

Windows CE
XHTML, color, 320x240pixel display, GPRS, keyboard
Palmpilot
HTML, 8-bit grayscale display, medium size display, GSM, hand writing input
Nokia 7110
WML, 96x65 pixel B&W display,GSM, key pad
Ericsson R380
WML, 4-bit grey 360x120 pixel dispaly, GSM, key pad or hand writing
ATHENS 2004
Athens 2004
Schedules
City Map
Options

Figure 4 – screenshots of mobile clients

It is important to mention that since there are multiple versions of the site, one for every version available, there is a need for a mechanism that redirects any device that visits the top-level URL (e.g. www.mobile2004.gr), to the part of the site that corresponds to the device used (e.g. www.mobile2004.gr/nokia7110/index.wml). This can be easily implemented by parsing the HTTP headers of the client's original request and identifying his/her browser by the "User-Agent" header [12]. Even in the case that the information exposed by this header doesn't match a client known to the system then the user can be redirected to a generic version of the site, which proves functional for his device, although not customized.
Figure 5 depicts the implementation of the meta-model described earlier in the case of the aforementioned demo site, focusing on the page with the city map. This page consists of a picture with the map, a header with the Athens 2004 logo and a footer with navigational widgets. This is an instance of the meta-model, i.e. a model per se that is described using all the necessary meta-model elements. The diagram shows only the elements for Nokia7110 and PalmPilot in order not to overload the diagram, even though the other two clients are represented in a similar way. Also with the intention of keeping the diagram simple, the relationships between the elements that were defined in the meta-model are note repeated here. Every model element in this diagram is distinguished by having a stereotype defined for it.

Stereotypes are a UML mechanism for extending the core of the language and are denoted by brackets in this diagram.

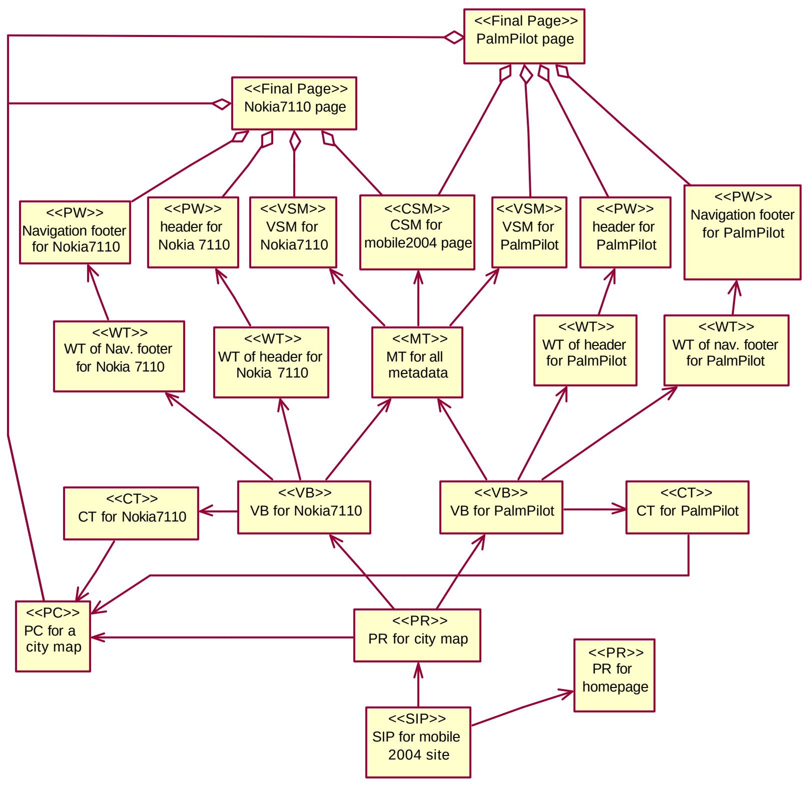

<<Final Page>> PalmPilot page
<<Final Page>> Nokia7110 page
<<PW>> Navigation footer for Nokia7110
<<PW>> header for Nokia 7110
<<VSM>> VSM for Nokia7110
<<CSM>> CSM for mobile2004 page
<<VSM>> VSM for PalmPilot
<<PW>> header for PalmPilot
<<PW>> Navigation footer for PalmPilot
<<WT>> WT of Nav. footer for Nokia 7110
<<WT>> WT of header for Nokia 7110
<<MT>> MT for all metadata
<<WT>> WT of header for PalmPilot
<<WT>> WT of nav. footer for PalmPilot
<<CT>> CT for Nokia7110
<<VB>> VB for Nokia7110
<<VB>> VB for PalmPilot
<<CT>> CT for PalmPilot
<<PC>> PC for a city map
<<PR>> PR for city map
<<PR>> PR for homepage
<<SIP>> SIP for mobile 2004 site

Figure 5 - the model for the mobile 2004 site

5 Future Work

Although the principles described above provide a solid foundation, they do not deal with the entirety of the diverse scenarios that are popular in contemporary hypermedia applications for mobile clients. Therefore we plan to extend this model in various ways in order to accommodate certain features that are currently missing. First of all dynamic content must be provided to the user in situations where the user needs to systematically draw information from a database or merely execute HTML-embedded server side scripts. Furthermore the model needs to take provisions about multiple languages used for the same content, a strongly emerging need especially for sites like the one illustrated in the case study section.
Another issue that concerns the model per se is the use of the Object Constraint Language

(OCL) [13] to define the model more formally with the use of well-specified constraints. OCL is the UML's recommended language for specifying constraints and can help in formalizing the various model elements and the relationships between them.
A formal evaluation of the model's effectiveness is another step thaw we plan to take in order to measure the quality of this work. This is already under way with the use of the model in new hypermedia applications for mobile users.
Finally, an interesting issue that is under research is the way that WOnDA can benefit from all the work recently done in the database area in order to deal with XML documents. This is caused by the variety of proposed models for storing and accessing XML information both natively and on object-relational databases.

References:

1. D. Synodinos and P. Avgeriou, "WOnDA: an Extensible Multi-platform Hypermedia Design Model", in proceedings of Efficient Web-based Information Systems (EWIS), September 2nd, 2002, Montpellier, France, Lecture Notes in Computer Science series, Springer-Verlag.
2. World Wide Web Consortium (W3C), "Composite Capabilities/Preference Profiles: Requirements and Architecture", W3C Working Draft, 28 February, 2000.
3. T. Ballz, C. Colby, P. Danielseny, L. Jategaonkar Jagadeesany, R. Jagadeesan, K. L'aufer, P. Matagay, K. Rehory, "Sisl: Several Interfaces, Single Logic", Journal of Speech Technology, Kluwer Academic Publishers, 2000.
4. Guruduth Banavar, James Becky, Eugene Gluzbergy, Jonathan Munson_, Jeremy Sussman, and Deborra Zukowski, "Challenges: An Application Model for Pervasive Computing," Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom 2000), 2000. http://www.research.ibm.com/PIMA/
5. MarcAbrams and Constantinos Phanouriou, "UIML: An XML Language for Building Device-Independent User Interfaces", in proceedings of XML '99, Dec. 1999, Philadelphia.
6. Daniel Billsus, Clifford A. Brunk, Craig Evans, Brian Gladish, and Michael Pazzani, "Adaptive Interfaces for Ubiquitous Web Access", Communications of the ACM, May 2002, Volume 45, Number 5.
7. G. Booch, J. Rumbaugh, and I. Jacobson, The UML User Guide, Addison-Wesley, 1999.
8. The Rational Unified Process, 2000, v. 2001.03.00.23, Rational Software Corporation, part of the Rational Solutions for Windows suite.
9. F. Garzotto, D. Schwabe, P. Paolini, "HDM- A Model Based Approach to Hypermedia Application Design", ACM Transactions on Information Systems, Vol. 11, #1, Jan. 1993, pp. 1-26.
10. T. Isakowitz; E. Stohr; P. Balasubramaniam, "RMM, A methodology for structured hypermedia design", *Communications of the ACM*, August 1995, pp 34-48
11. S. Ceri. P. Fraternali, A. Bongio, "Web Modeling Language: a modeling language for designing Web sites", proceedings of WWW9 Conference, Amsterdam, May 2000.
12. IETF RFC 1945, Hypertext Transfer Protocol -- HTTP/1.0
13. OMG, Object Constraint Language Specification, version 1.3. Framingham, MA, 1999.