Engineering Hypermedia for Multi-channel Access, with the use of XML

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

[1]National Technical University of Athens, Network Management Center, Heroon Polytechniou 9, Zografou 157 80, Greece, Tel: +30-210-7722241, Fax: +30-210-7721866, dsin@noc.ntua.gr

[2]National Technical University of Athens, Software Engineering Laboratory, Heroon Polytechniou 9, Zografou 157 80, Greece, Tel: +30-210-7722487, Fax: +30-210-7722519, pavger@softlab.ntua.gr

Engineering Hypermedia for Multi-channel Access, with the use of XML

Abstract. The plethora of networked devices and platforms that continuously come to light, as well as the emergence of alternative ways to access the internet, have increased the demand for multi-channel access to hypermedia applications. Researchers and practitioners nowadays not only have to deal with the challenges that classic hypermedia applications pose, but also have to face numerous considerations with respect to multi-channel delivery of the applications. This paper presents an attempt for attacking the problem of multi-channel hypermedia application development. In specific it proposes a model that explicitly separates the hypermedia content from its presentation to the user through a document engineering perspective, by employing XML content storage and XSL transformations. Our work is based upon the empirical results of designing, developing and deploying hypermedia applications on multiple platforms and client devices, and on the practices of well-established hypermedia engineering techniques.

Keywords: multi-channel delivery, hypermedia application, content personalization.

1 Introduction

For centuries mankind has used documents for recording and distributing information. At the beginning the form in which we recorded information was necessarily the same as the form in which the documents were presented to the reader. Though the introduction of the computer introduced the potential for a more flexible handling of documents, the early digital text processing systems' purpose was only to obtain the same quality as the one produced by traditional printing techniques. The potential flexibility of the computer was mainly used to provide better support for the authoring and production process, not for the distribution and presentation process. Due to the superior quality of printed paper as compared to computer displays, the final version of a document was still disseminated to its audience in the traditional way: on paper.

With the birth of hypertext, the World Wide Web and hypermedia, researchers and practitioners were given the ability to deliver information in several alternative forms, like different versions of sites, different browsers, different client devices etc. There was a paradigm shift from 'one source – one delivery medium' to 'one source – multiple delivery media'. This new paradigm seems to expand as new technologies and new client devices emerge, like for example the wireless World Wide Web (W4), where hypermedia applications can be accessed by wireless clients, anywhere and at any time. Typical examples of alternative hypermedia access channels are NTT DoCoMo's i-mode technology [http://www.nttdocomo.com] with its millions of users in Japan and WAP [http://www.wapforum.org/what/technical.htm] that is heavily deployed in

Europe and abroad.
The anticipation of faster and cheaper W4 (Wireless World Wide Web) that the 3rd generation (3G) wireless networks [http://www.3gpp.org] 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 multi-channel access. This task though can be overwhelmingly difficult since several problems have derived in building and maintaining hypermedia applications for access by heterogeneous platforms.
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 "phonetops" and 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, different clients might need different versions of web pages as a side effect of the different level of support for client-side scripting. 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 and extension of the **WOnDA** (**W**rite **On**ce, **D**eliver **A**nywhere) model [1, 2] that extends the capabilities of the

published model by giving the ability to maintain dynamic pages. 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 analyses the proposed model and its philosophy. Section 3 presents a case study of a demo site that is created with the aid of the model and deployed in four mobile clients. Section 4 introduces a short literature review, comprised of the most significant approaches in designing hypermedia applications for several presentational domains and mobile clients. Finally section 5 wraps up with ideas for future work.

2 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 multiple 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. The inclusion of references to dynamic data sources can be done in a variety of ways starting from wizard-driven predefined queries to databases for novice users, to hard-coding these references in the actual XML code for advanced users. 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. Information that is not static can be dynamically generated, e.g. retrieved from a database, or a web service, and then be integrated into the page generation process. The final page is then published to a Web Server and served to the appropriate clients through the Internet.

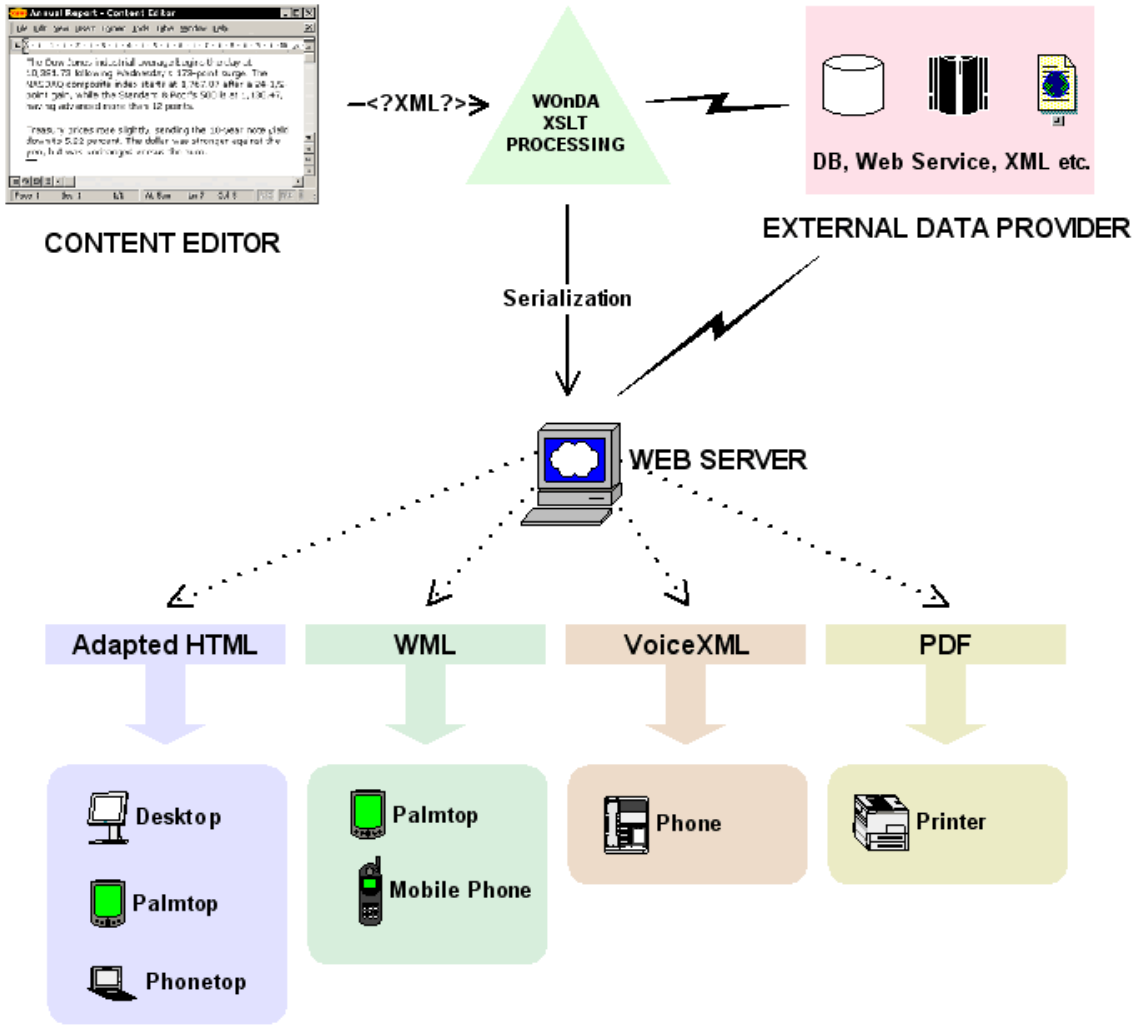

Figure 1 - A macroscopic view of the model

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 [11] (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 [12].

The conceptual model described here considers both static and dynamic 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. as it is integrated by dynamic and static data sources (DBs, Web Services, XML files)
2. The dynamic elements of a page that will provide information at the time of the user ‘s request
3. A set of navigational or promotional active objects or widgets like navigation bars, search boxes, menus, logos, ads, banners etc.
4. 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.
5. 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 [13], OODHM [14], RMM [15], WebML [16] 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 described 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. They describe both static information in the form of XML segments and more dynamic information in the form of processing instructions that communicate with external modules that provide information be querying DBs, Web Services etc.
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. Again these entities are responsible for both static and dynamic information.
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**). Again this entities are responsible for both static and dynamic information.
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. The 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. Page Content, Page Widgets and Content-Specific Metadata, are comprised of a static part and a dynamic part. 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.

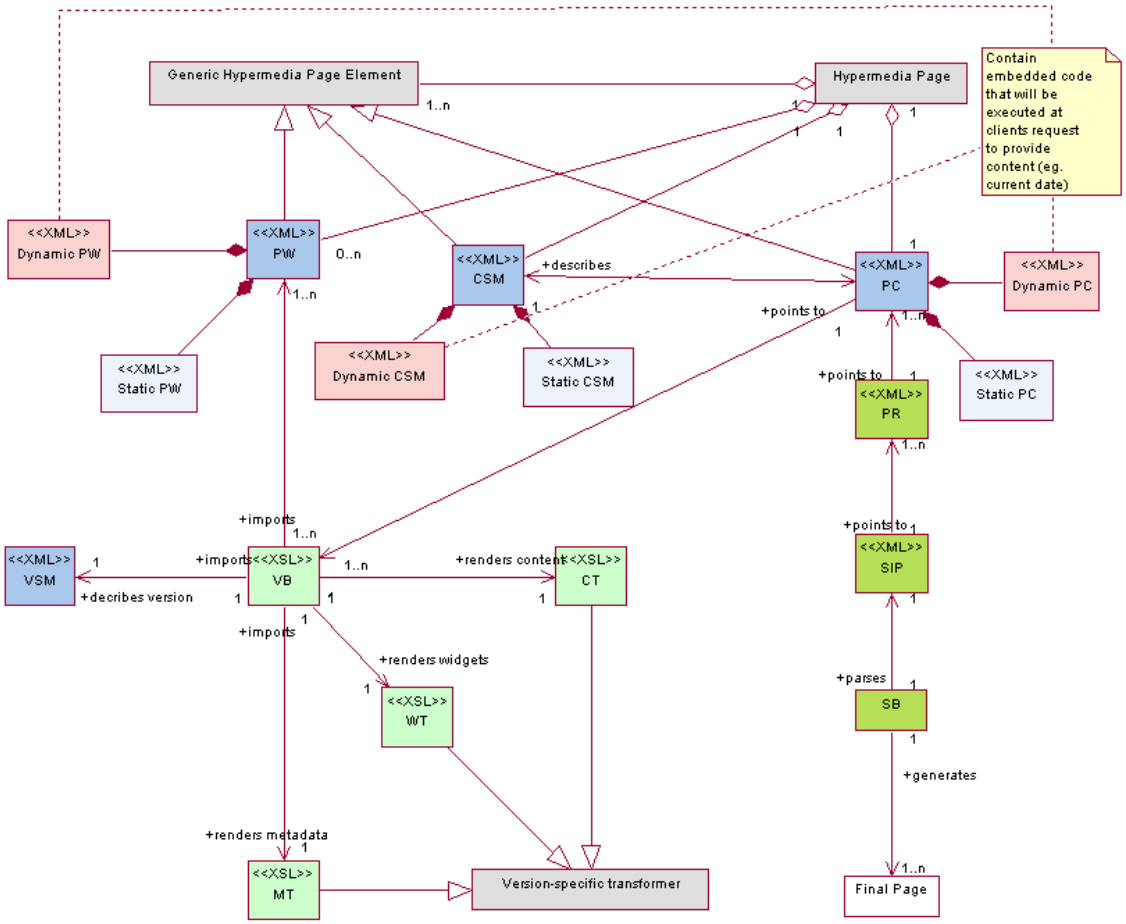

Figure 2 – the hypermedia page elements and the transformers

8. For every page there is a registry specific for it, called **Page Registry (PR),** which links 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.

Figure 3 depicts the page creation process that takes place with the aid of the last three elements. 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.

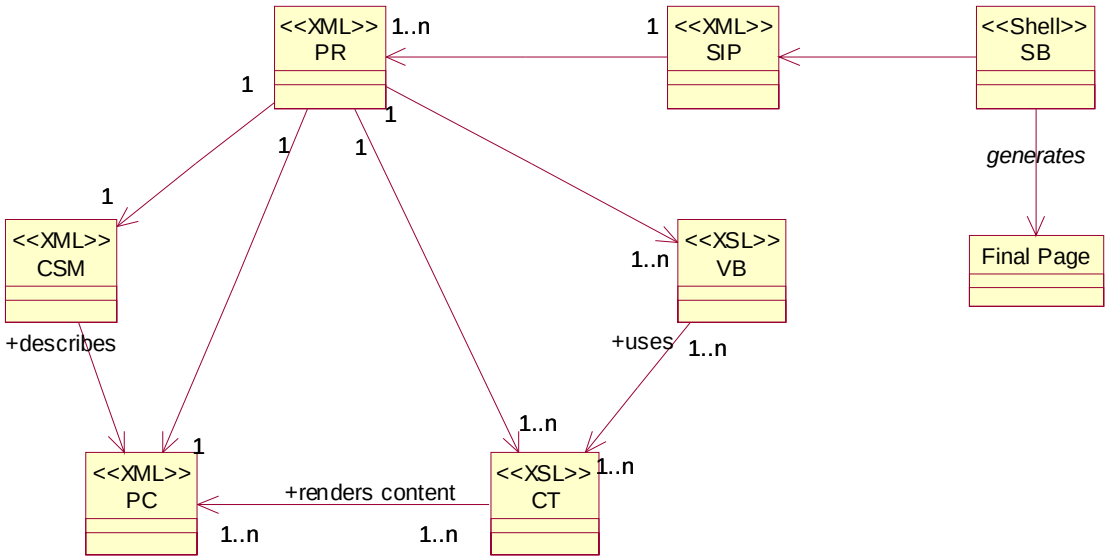

Figure 3 – The Page Generation Process

Figure 4 describes the integration of static and dynamic information to the PC entity. Initially an SB's module responsible for this integration (Integrator) parses the XML file that was developed with the aid of the Content Development Environment that was mentioned earlier. This file contains both static information in the form of XML fragments and processing instructions corresponding to inclusions of information from external sources like queries to databases, in the form of XML Processing Instructions (PI). The Integrator executes these PIs in an iterative manner and delivers the integrated PC. It is important to mention that this version of the PC still contains references to external data sources that will be executed at the time of the clients request.

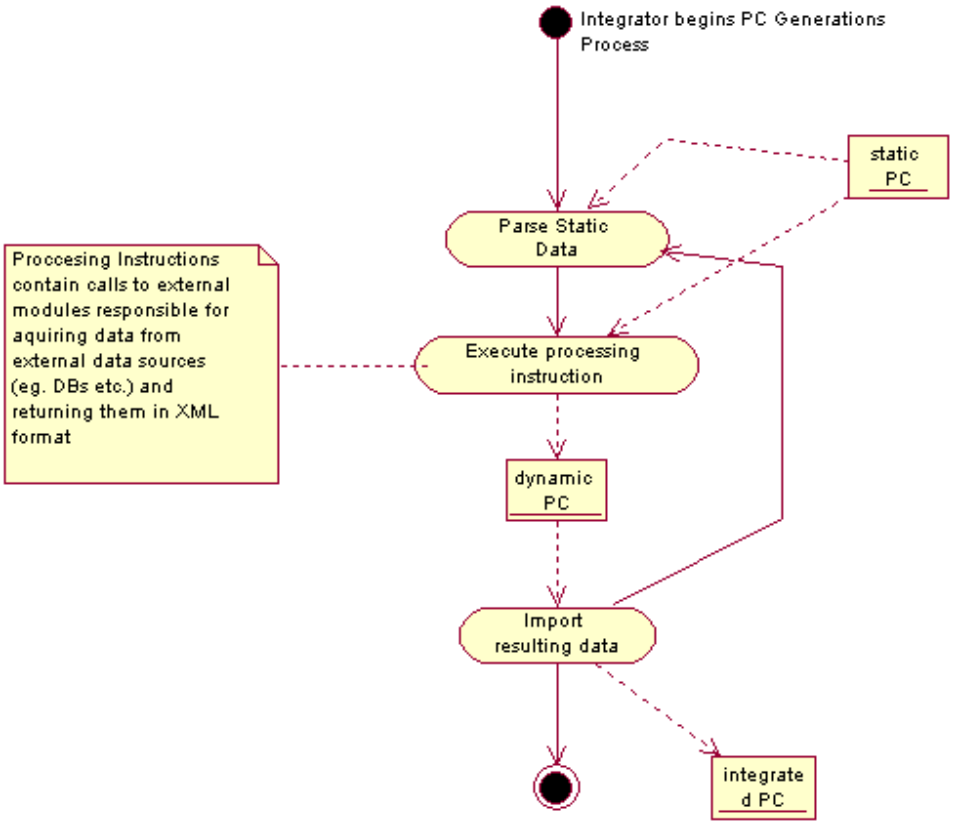

Figure 4 – integration of static and dynamic information to the PC

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 5, as a UML activity diagram, showing the discrete activities as well as the artifacts that activities produce or take as input.

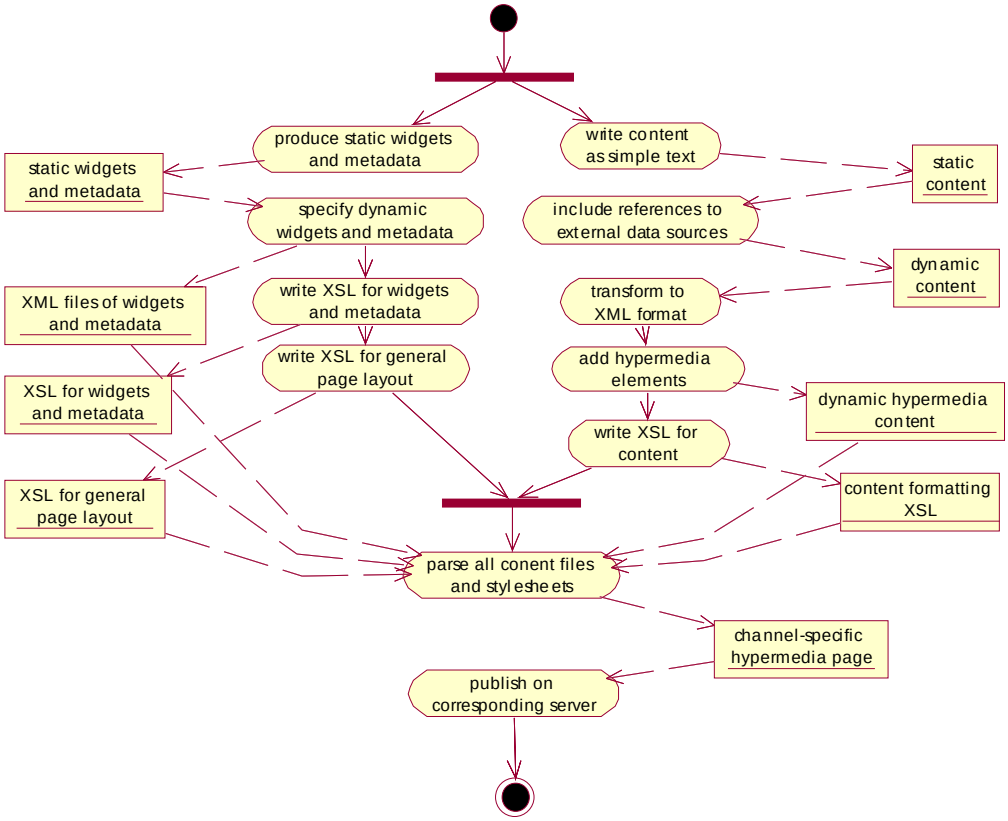

Figure 5 - a process model for applying WOnDA

3 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. It is an infotainment site that provides the citizens of Athens with information regarding entertainment like movies, music, restaurants etc. 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 picture is one where a MS WindowsCE client with wireless internet access, visits the home page of the site and through a certain number of page transitions finds where the movie theater of his choice is situated on the city map. Also the final page with the map is demonstrated on a variety of

other devices in order to visually demonstrate the diversity of such a mobile environment.

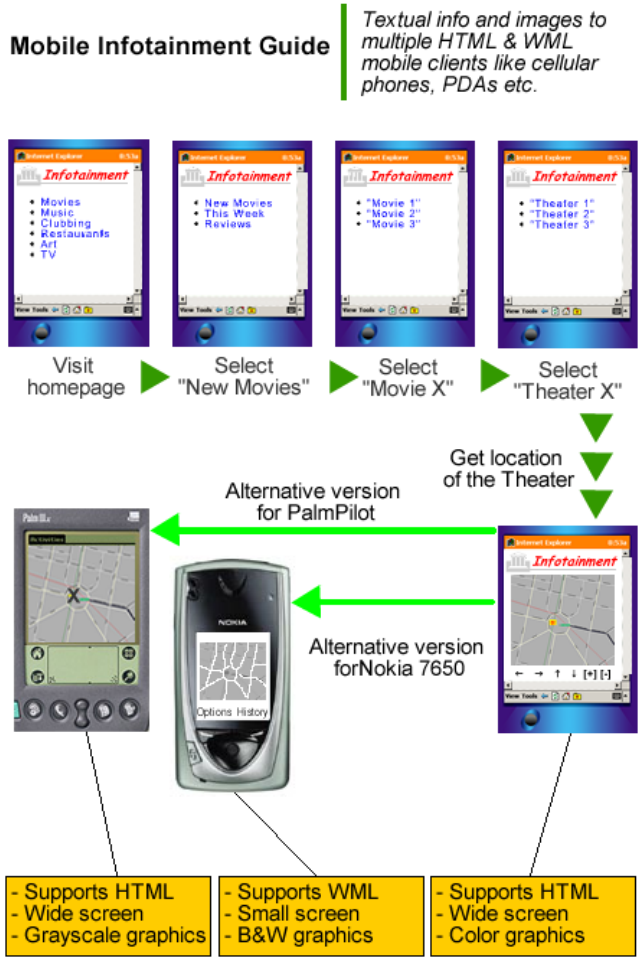

Figure 6 – screenshots of the four different mobile clients

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.infotainmentsite.gr), to the part of the site that corresponds to the device used (e.g. www. infotainmentsite.gr/nokia7650/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 [17]. 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 7 depicts the implementation of the meta-model described earlier in the case of the aforementioned demo site, focusing on the page with the Theater location Map in order not to overload the diagram. This page consists of a picture with the map 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. 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 guillemets in this diagram.

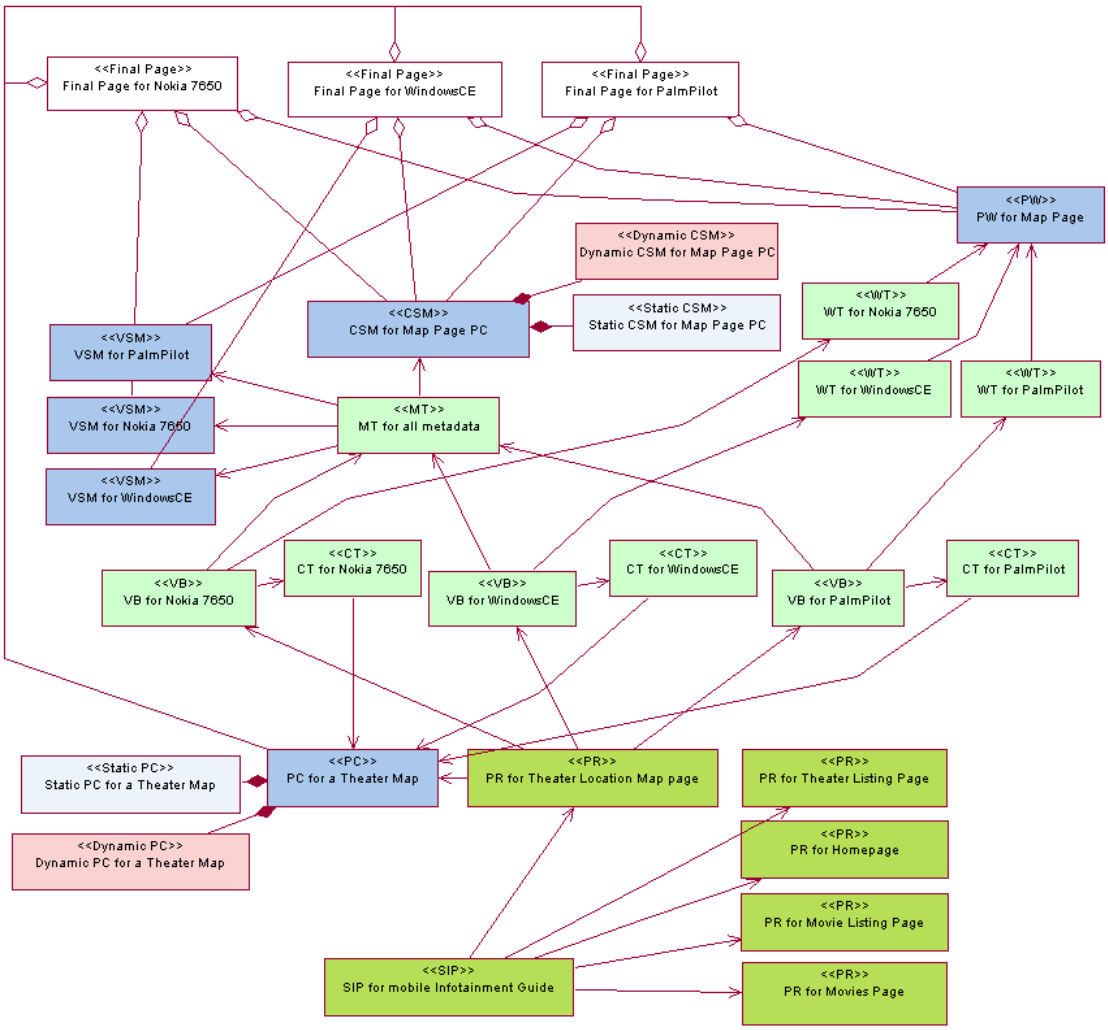

Figure 7 - the model for the mobile 2004 site

4 Literature Review

There is already work in progress by the World Wide Web Consortium (W3C), under the title of **Composite Capabilities/Preference Profiles** (**CC/PP)** [3] 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 generates 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 devices' profiles collected from a central or various distributed profile registries. On the other hand WOnDA proposes that creation of the presentation rules is done prior to user request, and has no provision for dynamic adaptation of content transformation rules 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, so that it is not required for the developer to know the specifics of a new device in order to make the necessary provisions for it.

Another effort by the W3C is the **XForms** [4], which is an XML vocabulary intended to be integrated to XHTML [5] or SVG [6] standards. XForms provide a platform-independent markup language for online interaction between a person (through an XForms Processor) and a remote agent. XForms are viewed as the successor of HTML forms that aim at separating the data being collected from the markup of the controls collecting the individual values thus resulting to platform independency and reusability of forms. As this effort receives the necessary support from industry and compatible clients emerge, we aim to integrate it in a future version of our model.

Curious Networks is developing **Multi-channel Access XML (MAXML)** [http://www.curiousnetworks.com/] which is designed especially for the growing need to deploy applications on not just one, but many access channels. Unlike many traditional development processes based on defining presentation, MAXML focuses on Interaction Oriented Development. Developers define their applications within MAXML in terms of the interactions that users may have with these applications. Through extensive usability tests, Curious Networks have created a Human-Information Interaction Model that defines standard ways of interaction between the users and information, regardless of the access channel. Although this approach is valuable in the sense that it provides a layer of transparency from the underlying delivery markup, it restricts the developers option within its core (Human-Information Interaction Model).

Another important approach of serving the same content to different clients is **Several Interfaces, Single Logic (Sisl)** [7]. 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, overkill.

The IBM's T.J. Watson Research Center has established a vision, named **Platform-Independent Model for Applications (PIMA)** [8]**,** (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 of 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.

Another approach worth mentioning is the **User Interface Markup Language (UIML)** [9] 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 very recent approach by Daniel Billsus et. al. for mobile access to hypermedia applications is presented at [10], 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.
As far as the graphics that are encountered in hypermedia applications, they are usually static bitmaps in GIF, PNG or JPEG format. The use of this kind of technology in a mobile context proves problematic due to the varying capabilities of mobile devices to render and display graphics. Traditionally the developer must adapt the image file in a number of different file formats, sizes, resolutions and depths of detail in order to accommodate the different needs of the application's target audience. During the last years, the convergence of computer graphics and other technologies such as artificial intelligence is leading to the development of **Smart Graphics** [23]**,** which recognize application requirements, user characteristics, host-machine capabilities, and target usage, and adapt themselves accordingly. The smart graphics differ from conventional computer graphics in that they can be programmed to recognize the platforms on which they are working, the tasks they must perform, and network capabilities such as bandwidth. If this adaptation is performed at the clients' side, this approach is based on the assumption that the device can handle such an intensive procedure of adapting the source graphics file. At this point WOnDA's traditional approach of before-hand server side adaptation according to target devices profiles proves more efficient for clients with smaller processing capabilities. This is the case for most mobile internet appliances.
SUN Microsystems has transferred knowledge, gained by working with larger, more powerful systems, to the field of mobile internet-enabled consumer products with **Java 2 Micro Edition (J2ME)** [http://java.sun.com/j2me/]. One of the key elements of this architecture is the Mobile Information Device Profile (MIDP), which provides the developer with a set of high and low-level APIs depending on the desired portability across a range of different devices. The high-level APIs are more portable but don't provide much control. One the other hand, the low-level APIs provide more control over the final outcome but require further effort to become portable. Although J2ME aims to provide applications to mobile devices and not hypermedia content as WOnDA does, both efforts face many similar problems that span the areas of HCI, UI design and associated methodologies.
An interesting approach in delivering hypermedia in multiple channels is the one proposed in [24], where the authors suggest the development of a **Smart Style** layer for the WWW by employing Semantic Web technologies. They propose a framework that is theoretically capable of providing intelligent stylesheets by utilizing three elements: a) common vocabularies for describing delivery contexts like CC/PP; b) intelligent transformation methods that take into account a wide variety of delivery contexts; c) the use of explicit metadata and design

knowledge. It is obvious of course that in practice these stylesheets would have to contain a large amount of design and domain knowledge encoded in RDF Schema [21] and DAML+OIL [22].
Hypermedia content is and probably always will be developed primarily for the desktop PC taking into account the capabilities and restrictions of such a device like large screen, adequate processing power, easy input mechanism etc. As mentioned earlier this practice leads in the distribution of information in a form that is usually not appropriate for the new-age internet capable devices that spring out like mushrooms these days. With this notion the authors of [25] propose a dual-mode Web browsing model that supports navigation and action in separate interfaces, serving the needs of users of smaller footprint devices. To demonstrate their model they have developed **m-Links**, a middleware proxy system that retrieves hypermedia documents and provides users with a navigation interface that separates links from page content. This "middleman" approach for content adaptation is contradictory to our model where content is customized by the issuing server.
The issue of determining the clients' capabilities and restrictions has proven an overwhelming task for web application designers since the early days of the WWW. Modern trends address this issue with a variety of methods, most of them based on the interpretation of the HTTP [17] headers of the client's browser. Specifically the "User-Agent" header is compared to a list of predefined user agents, e.g. browser software and/or client device and their characteristics. Another method is to use client-side scripting like JavaScript or VBScript. In this manner, the script can find out more about the specific settings of the environment it is called to operate on, like the resolution or color depth of the users' display. Furthermore many server side technologies like ASP, PHP etc. typically use a file named **browscap.ini** that holds information regarding the browser capabilities. Finally commercial products have been developed to extend the functionality of this technology like the BrowserHawk (http://www.cyscape.com/products/bhawk) that aid in the development of cross-platform, browser-friendly web applications.

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 an issue that concerns the model per se is the use of the Object Constraint Language (OCL) [18] 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", Springer-Verlag LNCS Volume 2426, pp 217-228.
2. D. Synodinos and P. Avgeriou, "m-WOnDA: The 'Write Once 'n' Deliver Anywhere' model for mobile users", in proceedings of Workshop on Conceptual Modelling Approaches to Mobile Information Systems Development (MobIMod'2002), October 2002, to be published by Springer-Verlag LNCS.
3. World Wide Web Consortium (W3C), "Composite Capabilities/Preference Profiles: Requirements and Architecture", W3C Working Draft, 28 February, 2000.
4. World Wide Web Consortium (W3C), "XForms 1.0", W3C Candidate Recommendation 12 November 2002.
5. World Wide Web Consortium (W3C), "XHTML 1.0 The Extensible HyperText Markup Language (Second Edition): A Reformulation of HTML 4 in XML 1.0", W3C Recommendation 26 January 2000, revised 1 August 2002.
6. World Wide Web Consortium (W3C), "Scalable Vector Graphics (SVG) 1.0 Specification", W3C Recommendation 04 September 2001.
7. 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.
8. 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/
9. MarcAbrams and Constantinos Phanouriou, "UIML: An XML Language for Building Device-Independent User Interfaces", in proceedings of XML '99, Dec. 1999, Philadelphia.
10. 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.
11. G. Booch, J. Rumbaugh, and I. Jacobson, The UML User Guide, Addison-Wesley, 1999.
12. The Rational Unified Process, 2000, v. 2001.03.00.23, Rational Software Corporation, part of the Rational Solutions for Windows suite.
13. 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.
14. D. Scwhabe & G. Rossi, "The Object-Oriented Hypermedia Design Model (OOHDM)", Communications of the ACM, Vol. 35, no. 8, August 1995.

15.T. Isakowitz; E. Stohr; P. Balasubramaniam, "RMM, A methodology for structured hypermedia design", *Communications of the ACM*, August 1995, pp 34-48
16.S. Ceri. P. Fraternali, A. Bongio, “Web Modeling Language: a modeling language for designing Web sites”, proceedings of WWW9 Conference, Amsterdam, May 2000.
17.IETF RFC 1945, Hypertext Transfer Protocol -- HTTP/1.0
18.OMG, Object Constraint Language Specification, version 1.3. Framingham, MA, 1999.
19.World Wide Web Consortium (W3C), “Authoring Challenges for Device Independence”, Working Draft, 25 October 2002
20.“Smart Style on the Semantic Web’, CWI, Jacco van Ossenbruggen, Lynda Hardman
21.World Wide Web Consortium (W3C), “Resource Description Framework (RDF) Schema Specification 1.0.”, W3C Candidate Recommendations, 27 March 2000, http://www.w3.org/TR
22.“Reference description of the DAML+OIL (March 2001) ontology markup language”, http://www.daml.org/2001/03/reference.html
23.Smart Graphics, http://www.smartgraphics.org
24.Jacco van Ossenbruggen & Lynda Hardman, "Smart Style on the Semantic Web", CWI.
25.Bill N. Schilit, Intel Research; Jonathan Trevor and David M. Hilbert, FX Palo Alto Laboratory; and Tzu Khiau Koh, Xerox Singapore , "Web Interaction Using Very Small Internet Devices", IEEE Computer, Oct 2002.