WAPS: Web Application Prototyping System

Roberto Paiano and Andrea Pandurino

Set-lab, University of Lecce, Italy
{roberto.paiano,andrea.pandurino}@unile.it

Abstract. The growing demand for web applications and the new multi-user and multi-device requirements of these has led to the need for a structured and well-reasoned approach that helps both the application designer and the developer to produce a good quality product. On one hand we have the application designer, who has to describe all aspects of the application and manage the complexity of the tasks; on the other hand both the application customer and the developer need to validate and understand the designer's choices. To conciliate these needs, we propose a prototyping framework based on W2000 [1][2] methodology: in this way the designer has a powerful tool to keep control over the web application, the customer has a point of reference for evaluating the design, and the developer can better understand the design thanks to the prototyped application.

1 Introduction and Background

During the development and design of web applications (WAs) there are many critical factors to be considered; the designer has to manage the individual aspects and must be able to predict the inevitable interactions between them. In order to manage WAs fully, the production of a mock-up application that prototypes the most important aspects as quickly as possible, with minimum investment of time and resources, is desirable. Szekely stated that prototyping involves building a small scale version of a complicated system in order to acquire the critical knowledge required to build a complete system [3]. Considering the relation between requirement elicitation and the mock-up application [4], the first version of the prototype will rarely get the users excited but the developers should encourage the users to give feedback for the revision of the prototypes and let them know that their feedback will be considered in the redesign. It might seem like a large effort is being spent on something that will eventually be discarded but this upstream activity represents good investment as it will avoid costly problems downstream. In the past few years, web application engineering (and thus prototyping) has been synonymous with ad hoc development and a lack of any structured methodological approach. Thus, several methodologies, supported by a suite of tools, have emerged in order to simplify and automate the development of WAs. HDM [5] proposes a model for hypermedia application design, which divides the conceptual schema into two categories: structural and navigational. Starting from the HDM model, Autoweb [6] introduces a variant notation - HDM-Lite - that adds a presentation schema to design a WA and store the schema together with

N. Koch, P. Fraternali, and M. Wirsing (Eds.): ICWE 2004, LNCS 3140, pp. 256–260, 2004.

the data in a database. Jweb [7] provides a design/prototyping environment, integrating XML technology with HDM. In order to stay modular and flexible, one of the relevant features of the JWeb environment is the use of XML to the exchange of information among the different tools. RMM [8] methodology, HERA [9], ARANEUS project [10] and the WebRatio generation tool [11] represent other interesting approaches in a model-driven prototyping. Besides these approaches, UML [12] has been extended to model web specific elements [13]; starting from the use of UML as a language and the new WA features, HDM methodology has evolved into a W2000 approach. W2000 has been defined in response to the transformation of Web-based hypermedia from read-only navigational "repositories" to complex applications that combine navigational and operations in a sophisticated way. This paper describes a completely new prototyping architecture based on the W2000 methodology and the results of the UWA project.

The W2000 experience gave rise to the UWA Consortium [14], which in turn started the UWA project, which began in January 2000 and was completed in December 2002. This defines a set of methodologies, notations, and tools to tackle the main problems foreseen in the design of WAs. In addition, the project addresses the need for standard design and documentation to improve the possibility for exchanges and interoperability by extending the Unified Modeling Language (UML) as a design notation and using XML for internal representation. The project output, interesting for the purpose of prototyping, is the *unified software* environment, based on Rational Rose [15], which integrates the tools specific to each modeling activity.

2 WAPS: Web Application Prototyping System

In accordance with the W2000 methodology and the UWA design support environment, we propose a prototyping tool, called WAPS, which is able to understand the design model and create a mock-up application, with the benefits described above. Rose helps to design the WA in graphic format, using standard UML notation, in accordance with W2000 methodology; in order to obtain a "machine understandable" description we use another rational rose add-in: "Unisys Rose XML Tools" produced by Unisys [16] that exports the UML diagram into a standard XMI [17] output. Before choosing the XMI as the input format for WAPS, we considered two other reading model methods such as to read the rose petal file in raw mode or to produce an add-in to export the model into an xml-like format. The XMI solution seems the more standard, and appears to be suitable for our purpose; thus WAPS uses as input the XMI model description produced using the Unisys add-in which exports the Rational Rose W2000 model designed with the UWA add-in. As mentioned above, W2000 methodology and the UWA design tool were thought to be more appropriate for the design of WAs than for prototyping purposes. Therefore it is necessary to complete the WA model in order to support the prototyping process. It is clear that the complementary information tool must be a Rational Rose add-in (WAPS-Add-in), which provides a uniform environment for designing (with the UWA add-in) and prototyping WAs. The prototyping information is exported at the same time as the design information, in XMI format by the Unisys add-in.

The WAPS architecture is composed of several modules: for the most part they make up the run-time environment (WAPS-RT), the other part provides a set of tools

(the *"InstanceDB Manager"* and all the "*Rational Rose Add-in*" in Fig. 1) to get WAPS-RT ready, such as the tool to insert the application data. The run-time environment, the WAPS core, has the main task of creating a mock-up application starting from the W2000 model in XMI format.

In accordance with the modular structure of the W2000 methodology and the various aspects of WA, it is possible to identify a clear n-tier architecture for the WAPS run-time environment. The architecture was born as a generalization of the three-layer software architecture used to develop a pilot application based on W2000 methodology [18]. This choice was highly suitable for the W2000 methodology: just as the model examines one by one the different aspects (information, navigation, publishing) of WAs in order to manage their complexity, each architecture layer manages a single aspect (or a pool of aspects), and provides services to the other levels. Since WAPS is based on an XMI description, it's clear that all the data managed in the modules are in XML format; furthermore, all interaction between modules must be in the same format; the XML verbosity and adaptability ensure good portability and scalability of the whole system; furthermore, after checking the environment, it is possible to distribute the modules on several machines using web service techniques based on XML messages.

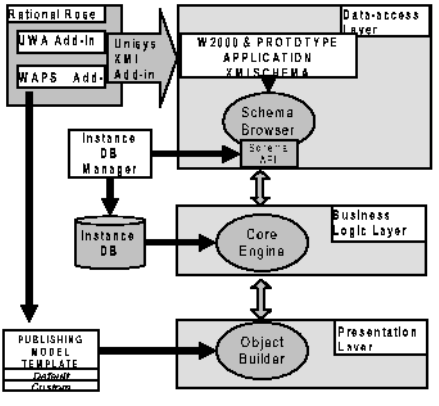

Fig. 1. WAPS architecture

In the WAPS-RT design, each level and thus each module manages a particular WA aspect; in detail:

- **Schema Browser**: WAPS uses the WA schema based on the W2000 model to prototype the WA. The schema is the XMI description exported from Rational Rose. This module allows a unique entry point to the WA schema, hiding the complexity in order to manage the XMI in raw mode. The module provides a set of schema APIs (S-API) to navigate the WA model via W2000 primitives.
- **Core Engine:** This module corresponds to the business level for a three-tier application. This module has the task of understanding the requests from the Object Builder, using the S-API of the schema browser to compose the reply schema that will contain the application data taken from the Instance DB. Since this module creates the reply schema (and thus the page that the user will see), it is clear that all

design customizations take effect at this stage. Users not only access the information but interact through the operations. In the future, this module will provide standard methods to implement the operations. Obviously the operation will be expressed as manipulation of the W2000 model elements. Thus, the task of adding a product to the shopping bag is translated in WAPS as adding an instance of PRODUCT to the dynamic collection called "Shopping bag". In this way it is possible to describe complex operations as a sequence of basic ones performed on model elements. The importance of this piece of the framework is plain, if we consider its function, and it is so big that we may call it the "Core Engine".

- **Object Builder**: this module is the door to WAPS systems: the user request comes in, the prototyped page goes out. The module moves the request to the Core Engine and receives the response in XML-like format. Its main task is to apply a template to make the page visible. In order to manage several devices, WAPS uses the XSLT transformation to obtain HTML or WML pages in XHTML [19] format. XHTML is the keystone in W3C's effort to create standards that provide richer web sites on an ever increasing range of browser platforms. Browsers are not limited to desktops and include mobile phones and PDAs. Additionally, Cascading Style Sheets (CSS) can be used to control the content presentation. The use of CSS to manage the multi-device task is quite suitable for the prototyping purpose where the layout aspects are to provide an idea of the final layout.

WAPS uses several information repositories to store the data: the XMI schema described earlier, the Instance DB and the publishing model template.

The Instance DB is an E-R database that contains the data that will be showed to the user. The E-R schema stores the data in meta-structures derived from the W2000 model; thus the schema is fixed and doesn't change with the domain of the WA being prototyped. The Publishing Model Template is a E-R database which contains the references to the visualization template to create the page. There are two kinds of template: the default templates, which are generic and will be used to view the model structures such as the publishing unit, and the custom template, which the designer can create and associate with a specific page or publishing unit.

The Instance DB manager is a specific tool to populate the Instance-DB: it accesses the WA schema to understand the data structure in order to guide the operator through the schema to insert meaningful data.

3 Conclusion and Future Work

This paper has concentrated on the model-driven prototyping of web applications. We propose a prototyping system called WAPS that allows the designer to produce a WA mock-up starting from its W2000 model drawn up using Rational Rose. In accordance with the W2000 model structures, the multi-layer WAPS architecture encapsulates in each level a specific aspect; this, combined with the extensive use of XML-like format (to describe the model and the application data, and to exchange information between modules) ensures good scalability and adaptability of the environment.

Future work is evolving in two directions: the first aims to offer a better support for operation and customization and to improve the quality of the prototype; the second aims to produce support tools: considering the importance of application data in obtaining a good prototype, we are thinking of creating a tool that populates the

Instance-DB with data from statistics algorithms. As for the quality of application data, we are studying the features offered by the semantic web in order to add a new semantic layer to WAPS.

References

1. L. Baresi, F. Garzotto, Paolo Paolini, From Web Sites to Web Applications: New Issues for Conceptual Modeling, *Proceedings WWW Conceptual Modeling Conference*, Salt Lake City, October, 2000.
2. L. Baresi, F. Garzotto, and P. Paolini, Extending UML for Modeling Web Applications, *Proceedings of 34th Annual Hawaii International Conference on System Sciences (HICSS-34)*. IEEE Computer Society, 2001.
3. P. Szekely, User Interface Prototyping: Tool and Tecniques, 1994, USC/Information Sciences Institute
4. Steve McConnell, Software Project Survival Guide, *Microsoft Press*, 1998
5. F. Garzotto, P. Paolini, and D. Schwabe, HDM–A model Based Approch to Hypermedia Application Design, *ACM Transactions on Information Systems*, 11,1, pp 1-26
6. P. Fraternali and P. Paolini, Model-driven Development of Web Applications: The AutoWeb System, *ACM Transactions on Information System (TOIS)*, Vol. 18, Issue 4, October, 2000, pp. 323-382
7. Nicola Fiore, Leonardo Mangia, Roberto Paiano, Delivering Web Application: JWeb II Navigation Engine, *TELEC '02 International conference*, Santiago de Cuba (Cuba)
8. Isakowitz T., Stohr E.A. and Balasubramanian P., RMM: a methodology for structured hypermedia design, *Communications of the ACM 38 (8)*, Aug. 1995, pp. 33-44
9. Flavius Frasincar, Geert-Jan Houben, Richard Vdovjak , "Specification Framework for Engineering Adaptive Web Applications", *WORLD WIDE WEB CONFERENCE 2002*, Honolulu, Hawaii, USA
10. P. Atzeni, A. Masci, G.Mecca, P.Merialdo, G. Sindoni, " The Araneus Web-based Management Sistems", 1998
11. WebRatio site development studio, www.webratio.com, 2002
12. G. Booch, I. Jacobson, and J. Rumbaugh, *The unified modeling language user guide*, Addison-Wesley 1998, Readings, MA
13. J. Conallen, Modelling Web application achitectures with UML, *communication of the ACM*, Vol. 42, Iss. 10 Oct. 1999, pp. 63-70
14. UWA Consortium. General Definition of the UWA Framework. Technical report EC IST UWA Project, 2001.
15. IBM Rational Software, *www.rational.com*
16. Unisys Corporate, *www.unisys.com*
17. Object Management Group, *www.omg.org/technology/documents/formal/xmi.htm*
18. Paiano, A. Pandurino, From the Design to the Development: a W2000 Based Framework, Issues and Guidelines, *IRMA International Conference*, Philadelphia, 2003
19. XHTML 1.0 The Extensible HyperText Markup Language (Second Edition), www.w3.org/TR/xhtml1/