An XML-Based Approach for Fast Prototyping of Web Applications

Antonio Navarro, Baltasar Fernandez-Manjon, Alfredo Fernandez-Valmayor, and Jose Luis Sierra

Departamento de Sistemas Informaticos y Programacion, Facultad de Informatica, C/ Juan del Rosal 8, 28040, Madrid, Spain
{anavarro, balta, alfredo, jlsierra}@sip.ucm.es

Abstract. This paper presents an XML-based approach for the fast prototyping of Web applications. Two XML Document Type Definitions (DTDs) are used to rationalize the development of these prototypes and a Java engine processes the instances of both DTDs to automatically obtain a running prototype of the application.

1 Introduction

The use of markup technologies to develop hypermedia applications, and more specifically to develop Web hypermedia applications has been from the beginning at the core of the Web. The basic language for the development of Web applications has been the *HyperText Markup Language* (HTML) [1], an SGML [2] based language. Other significant efforts have been the *Hypermedia/Time-based Structuring Language* (HyTime) [3], a very powerful SGML-based standard for the characterization of hypermedia applications, and the *Synchronized Multimedia Integration Language* (SMIL) [1], a markup language mainly centered on the multimedia domain. More recently the *Web Modeling Language* (WebML) [4] has become one of the most successful languages for Web development.

Our approach tries to gather the key benefits of the previous markup technologies while avoiding some of their shortcomings, and it is strictly centered at the prototyping stage.

In this short paper we introduce our general approach for the development of hypermedia applications. Then we present the process model that manages the use of our XML-based approach in the prototyping of Web applications. Finally, we present the conclusions and future work.

2 Plumbing, Pipe, and PlumbingXJ

The use of Software Engineering techniques is a must during the development of hypermedia applications [5]. In our approach we have defined the *Plumbing* process model [6] (which is an specialization of Fraternali's process model [7]). Plumbing has two loops. The first loop is centered on the conceptualization-prototyping phase, and the second on the design-development phase. We have focused our work at the conceptualization-prototyping stage, preferring an open solution (like the one presented in [8]) for the design-development stage. In our approach, the conceptualization-prototyping loop is guided by the *Pipe Hypermedia Model* [6], a model that characterizes a hypermedia application using three components: (i) the Pipe *contents graph* that characterizes the structure of the informational content of the application and its navigational relationships; (ii) the Pipe *navigational schema* that characterizes the elements of the graphical user interface and their relationships; (iii) and finally the Pipe *canalization functions* that relate both aspects of the hypermedia application. The Pipe model also has a default browsing semantics that describes the behavior of the application after an anchor selection.

The conceptualization stage is focused on obtaining a Pipe representation of the hypermedia application. Then, at the prototyping stage we need different formalisms to represent the Pipe characterization of the application in such a way that it can be processed by a tool that automatically builds the prototype. The selected mechanisms for representing the Pipe structures during the prototyping stage must be tuned with the technologies used at the design-development stage, or it must be flexible enough to permit a smooth transition from prototyping to development.

PlumbingXJ [6] is a specialization of Plumbing where XML is the selected mechanism for describing the structure of the contents graph and the navigational schema of the application, and where a Java tool generates the prototypes automatically processing the XML documents (hence the *XJ*). In PlumbingXJ the only changes appear at the prototyping stage, and as in Plumbing, PlumbingXJ is mainly centered at the conceptualization and prototyping stage. There are two reasons for choosing XML for describing the application at the conceptualization stage: today XML is accepted as a universal interchange format [9], and it provides a great structuring capacity [10]. Both characteristics make XML a perfect choice for the characterization of hypermedia applications [11].

3 Content DTD and Application DTD. Overmarkup

In PlumbingXJ the *content DTD* defines the markup language that structures the informational contents and links of the hypermedia application (and therefore it structures the Pipe contents graph). Due to the heterogeneous structure of the contents that can appear in a hypermedia application it is specific for every case. The *content document* is an instance of the content DTD and represents the contents and links of the hypermedia application (i.e. codifies the Pipe contents graph).

In PlumbingXJ the a*pplication DTD* defines the language that structures the graphical user interface of the hypermedia applications (and therefore it structures the Pipe navigational schema). It is unique for every application and uses a closer GUI terminology instead of the Pipe vocabulary. The *application document* is an instance of this DTD and not only provides the elements of the GUI and their navigational relationships, but also codifies the canalization functions that relate this GUI with the contents of the hypermedia application (i.e. it represents the Pipe navigational schema and the Pipe canalization functions). The overmarkup technique and some element attributes in the DTD are responsible for this canalization.

The key idea behind *overmarkup* is quite simple and extremely useful. The actual contents of the XML elements of the application document are XPath [1] references to the content document. These references select the contents that must appear in every pane. The Java application *Automatic Prototypes Generator*, APG, processes the application and content documents and produces the desired prototype. The navigational schema of the application is generated by the APG from the application document. This schema is a Java Swing skeleton that supports the GUI of the application. The contents that appear in this Java Swing skeleton are HTML pages generated by the APG from the content document and the XPath expressions that appear in the application document. An XSLT transformation [1] produces the transition from XML to HTML. This transformation is always the same and uses the hypermedia structure codified in the content document via some selected attributes. In this way the content document can be structured in any desired way because the XSLT transformation only needs a few attributes to produce the HTML pages.

The separation of layers (as in the Dexter model [12]) allows for a fast generation of prototypes where changes in the navigational schema have no effect at the content level, or changes in the content level (maintaining the structure referenced by XPath expressions) have no effect on the navigational schema.

The construction of the navigational schema and its relationships with the contents can be done manually, or via a CASE tool. At the last stage, the customers evaluate the prototype, and when the hypermedia structure is stable enough, the design stage begins. The use of XML in the structuring of the document offers an easy transition to HTML (or another format) using XSLT transformations. This approach is preferable to the direct use of HTML in prototyping due to the flexibility and structuring power provided by XML [1].

4 Conclusions and Future Work

The presence of a process model assures a systematic way for application development, while the design of the Pipe hypermedia model allows for the changes in one of the two main aspects of hypermedia applications (contents or GUI) without affecting the other.

Moreover, the utilization of XML at the prototyping stage adds legibility and produces a high-level language appropriated for the prototyping of Web applications. Note that all the markup technologies referenced in this paper increase the abstraction level in the development of Web applications, constraining the flexibility of direct coding using a general purpose programming language (e.g. Java), and our approach is not an exception. Finally, XML and Java provide valuable platform independence.

At present, we are working on the APG finalization in order to deal with the full expressive power of Pipe. The APG is the kernel of the *PlumbingMatic* application, a CASE tool that integrates the Pipe structures and the automatic generation of prototypes. This tool will help us in the application of PlumbingXJ to the development of several applications that we are creating.

Finally, we plan to extensively apply PlumbingXJ to confirm the viability of using our techniques of conceptualization and prototyping in the design and development phases.

Acknowledgements. The Spanish Committee of Science and Technology (TIC2000-0737-C03-01, TIC2001-1462 and TIC2002-04067-C03-02) has supported this work.

References

1. World Wide Web Consortium. http://www.w3.org/
2. International Standards Organization: Standard Generalized Markup Language (SGML), ISO/IEC IS 8879 (1986)
3. International Standards Organization: Hypermedia/Time-based Structuring Language (HyTime), ISO/IEC IS 11744.1992 (1992)
4. Ceri, S., Fraternali, P., Bongio. A.: Web Modeling Language (WebML): a Modeling Language for Designing Web Sites. In Proc. www9 Conference, Amsterdam (2000).
5. Garzotto, F., Paolini, P., Schwabe, D.: HDM: A model-based approach to hypertext application design. ACM Transactions on Information Systems **1** (1993) 1–26.
6. Navarro, A., Fernandez-Manjon, B., Fernandez-Valmayor, A., Sierra, J.L.: Formal-Driven Conceptualization and Prototyping of Hypermedia Applications. In: Kutsche, R.D., Weber, H. (eds.): Fundamentals Approaches to Software Engineering 2002, Proc. European joint conferences on Theory and Practice of Software 2002, Lecture Notes in Computer Science, Vol. 2306. Springer-Verlag, Berlin Heidelberg New York (2002) 308–322.
7. Fraternali, P.: Tools and Approaches for Developing Data-Intensive Web Applications: A Survey. ACM Computing Surveys **3** (1999) 227–263.
8. Rossi, G., Schwabe, D., Lyardet, F.: Designing Hypermedia Applications with Objects and Patterns. International Journal of Software Engineering and Knowledge Engineering **7** (1999) 745–766
9. Bryan, M.: Guidelines for using XML for Electronic Data Interchange. http://www.xmledi-group.org/xmledigroup/guide.htm (1998).
10. Sperberg-McQueen, C. M., Goldstein, C.M.: HTML to the Max A Manifesto for Adding SGML Intelligence to the World-Wide Web. In Proc. 2nd World Wide Web Conference '94: Mosaic and the Web, Chicago (1994).
11. Navarro, A., Fernandez-Valmayor, A., Fernandez-Manjon, B., Sierra, J.L.: Using Analysis, Design and Development of Hypermedia Applications in the Educational Domain. In Ortega, M., Bravo, J. (eds.): Computers and Education: Towards an Interconnected Society. Kluwer Academic Publisher, The Netherlands (2001) 251–260.
12. Halasz, F., Schwartz, M.: The Dexter Hypertext Reference Model. Communications of the ACM **2** (1994) 30–39