

Meta-Design for “Sensible” Information

Louis Weitzman

IBM Internet Technology Group

1 Rogers Street

Cambridge, Ma 02142 USA

louisw@us.ibm.com

ABSTRACT

Many content management systems today do not take into account how the information is to be used. These tools, and the designers who use them, must adapt to the new environment in which we live. Designers can not be present at all times to craft each presentation. Likewise, systems must support the design process to ensure the consistent and timely presentation of information in a variety of contexts. These systems need to provide information fragments for reuse in a variety of contexts. In addition, these information fragments need to have a “sense of themselves” in order to be able to render properly in these different contexts. These different situations each have their own requirements on the information. The information, therefore, must be carefully designed and can not be automatically derived. Content management systems must support *sensible* fragments of information that can be presented in different contexts so that it is relevant to the task at hand and to the device on which it is rendered.

Keywords

Content management, XML, XSL, information reuse, object-oriented technology

INTRODUCTION

As we move into more dynamic environments, where information is customized to each user and task, the content management systems must support the notion that designers cannot be present at all times in dynamic environments. Designers must create meta-designs, or representations of designs, to be used in the interactive and personalized delivery of information. This ability to provide “agile publishing” [1] challenges the standard practices in the design community. In addition, the content must have a “sense of itself” [4] so it knows how to render itself given its current context.

Current publishing systems manage the content by providing version and access control, publishing of the pages, organizing the workflow, separation of content and design, etc. However, in real life, information is reused

across applications, portals, devices, and users. Even with the support of the separation of content and design, most systems today do not easily accommodate the reuse of information in an efficient way. Their frameworks enable it, but it is up to the organization to customize the environment with complicated scripts and custom code. These tools need to support not only the separation of content from design, but also the reuse of content for multiple purposes, and support the context in which each information fragment is being used.

Franklin Content Management System

At IBM’s Internet Technology Group, we developed a prototype content management system, called Franklin [3, 5], to address some of these issues. Franklin was based on the notion that content is organized into reusable “fragments” of information. These fragments support content reusability with a simplified model of the management of content and design that enforces integrity and consistency. In addition, we supported the customization of content to users and the delivery of content to a variety of display devices. In Franklin, the separation of content from design was achieved through the industry standards of XML and XSL stylesheets.

A strong guiding principle in Franklin’s development was to hide the details of the underlying implementation and data representations from the user. Automatically generated forms provided editors with a simplified interface to enter content. This form is generated from the document type definition (DTD) and presents input fields for only the data the user needs to edit. Other data is hidden or automatically generated by the system.

Franklin incorporated research from IBM that was developed for the events infrastructure, including the Olympic games web sites. This technology, called Trigger Monitor [2], maintained an object-dependency graph of information fragments within an HTML web site. When a fragment of information changed, all the pages that depended on that fragment were automatically and efficiently updated. This only began to address the issue of content reuse.

We extended Trigger Monitor to support XML and XSL stylesheets. We added dependencies to track the XML content, as well as the XSL stylesheets that embody the

site's design. Then, when the content or the design of the web site changed, all pages that depended on those fragments would automatically be updated. The ibm.com country portal sites now use Franklin, which contains approximately 29,000 fragments of information for 86 countries. The system is currently used by approximately 150 editors. For ibm.com the use of XML and XSL as fragments has enabled the easy publication to different audiences and different devices. The use of information fragments has ensured the consistency and relevance of the information.

A document-based information model

The separation of content from design provides the basis of the document-based model. It simplifies the maintenance of both the content and the design. Each is controlled separately and can be authored by people with the appropriate skill sets. As the content evolves, authors of that content can modify the information documents. As the design evolves, designers in charge of the look & feel can modify the new stylesheets. Thus, design rules are encoded and stored centrally which increases the integrity of the overall design strategy. It guarantees a strong and consistent brand across the published information. This separation of content and design also eases the delivery of content to new devices. Each new device represents a new set of stylesheets but the content remains unaffected. Today, this model is a widely accepted practice in the industry.

A fragment-based information model

A slightly enhanced content model supports a fragment-based approach. In such a system, there is no limit to the ability to reuse the information. When properly tagged, content can be reused for the customization to different audience segments as well as different devices. But along with this reuse comes the need to effectively update dependent content and design fragments. A system that takes this approach, must maintain dependencies for automatic and efficient updates. With the object dependencies being tracked, when content is modified in one place, it can automatically trigger updates to documents wherever that content occurs. This intelligent updating of content also applies to the updating of the designs. The dependency of a particular stylesheet is maintained and when that asset changes, the dependent pages are automatically updated.

Challenge: A context-based information model

One of the problems of a fragment-based model is that it is difficult to extend given a new context. It is not possible to know, a priori, all the possible contexts of use. When new uses arose, we could add elements to the document definition and modify the stylesheets to enable the presentation of the fragment in the new contexts.

However, if the information fragments would have a "sense" of themselves, adding a new context could

control the way the information would be used in that situation. These contexts would be based on these constraints and should be made explicit for each fragment. By applying the techniques from object-oriented and constraint-based programming, we could simplify the interface for the content management user by abstracting and encapsulating the knowledge in the information fragments. The end user of the information could then see a more task- and context-based presentation while the content manager could see information that is more understandable and maintainable.

CONCLUSION

The first generation of content publishing systems on the web was focused on HTML. Now, XML/XSL is an important standard. But, because of the HTML legacy, many systems don't properly support this new separation of form and content. More over, fewer systems actually support the model of reusable information fragments. Unfortunately, no systems today support the notion of context-based information management systems with information that has a sense of itself. Future work needs to focus on the information objects and their potential use in different contexts. Explicitly representing these contexts would allow the system to be smarter in how it would apply that information to new situations.

ACKNOWLEDGEMENTS

I would like to thank the Internet Technology Group for providing the support for this project and especially Dikran Meliksetian for his review and input on the this work.

REFERENCES

1. Haimes, R. Managing Workflow and Content for Agile Publishing, Color Publishing, pp 24-33, January/ February, 1994.
2. Iyengar, A., Challenger, J., Dias, D., and Dantzig, P. "Techniques for Designing High-Performance Web Sites," Submitted to *IEEE Journal of Internet Computing*, October, 1998.
3. Meliksetian, D., Weitzman, L., Elo-Dean, S., Milton, J., Zhou, N., Davis, P., and Wu., J. XML Content management: Challenges and Solutions. XMLEurope 2001, Berlin, Germany, 21-25 May 2001.
4. Negroponte, N. presented the idea of "signals with a sense of themselves" MIT Media Lab, circa 1995.
5. Weitzman, L., Meliksetian, D., Elo-Dean, S., Wu, J., Zhou, N., and Gupta, K. Transforming the Content Management Process at ibm.com, CHI2002-AIGA Experience Design Forum Minneapolis, Minnesota USA, April 21-22, 2002.

Terminology sidebar

Industry Standards within Franklin:

XML: Extensible Markup Language (XML) is the universal format for structured documents and data on the Web. [World Wide Web Consortium, Extensible Markup Language, <http://www.w3.org/XML/>]

XSL: Extensible Stylesheet Language (XSL) is a language for expressing stylesheets. It consists of three parts: XSL Transformations (XSLT), a language for transforming XML documents; the XML Path Language (XPath), an expression language used by XSLT to access or refer to parts of an XML document; and the XSL Formatting Objects (XSLFO), an XML vocabulary for specifying formatting semantics. [World Wide Web Consortium, Extensible Style Sheet Language, <http://www.w3.org/Style/XSL/>]

DTD: Document Type Definition (DTD) provides a means for defining the structure, content and semantics of XML documents. This standard has evolved into XML Schemas. [World Wide Web Consortium, XML Schemas and Document Type Definitions, <http://www.w3.org/XML/Schema>]

HTTP: Hypertext Transfer Protocol (HTTP) is the protocol for communication on the web. [World Wide Web Consortium, Hypertext Transfer Protocol, <http://www.w3.org/Protocols/>

WebDav: Web-based Distributed Authoring and Versioning (WebDav), is a set of extensions to the HTTP protocol to support users to collaboratively edit and manage files on remote web servers. [Web-based Distributed Authoring and Versioning, <http://www.webdav.org/>]