

WCML: An enabling Technology for the Reuse in object-oriented Web Engineering

Martin Gaedke, Daniel Schempf, Hans-Werner Gellersen

Telecooperation Office (TecO), University of Karlsruhe, Germany
email: {gaedke, schempf, hwg}@teco.edu

Abstract: Much is known about quality improvements and cost reduction by reusing software. Nevertheless, making good use of reuse in the Web is a rare practice. One reason for that is the coarse-grained implementation model of the Web. We introduce the WebComposition Markup language, which is an application of the XML and allows describing design and code in an object-oriented way. WCML addresses the software engineering problems by automatically mapping an object-oriented model to the Web implementation model. The object-oriented Markup makes it possible to use software engineering knowledge on top of the Web implementation model and to reuse the code and design.

Keywords: Web Engineering, XML, Web Application Maintenance, Object-orientation, Software Reuse.

1 Introduction

The development of applications for the World Wide Web (WWW) has progressed in the last few years and the Web has become a standard platform for distributed applications beyond publishing. Nevertheless, the development discipline is usually ad-hoc, resulting in applications of poor quality and causing tremendous cost for maintenance and evolution. It is widely recognized [1, 4, 7] that the lifecycle of Web applications is no longer manageable without Web Engineering, the application of software engineering practice to the Web. Unfortunately, design and code reuse for quality improvement and cost reduction in the WWW is a tiring venture. The main cause for this problem is within the Web implementation model itself, because of its deliberately simple and coarse-grained nature [2, 3].

2 The WebComposition Model

In WebComposition, Web entities are modeled as component objects with a set of operations specifying the component behavior. Components can model Web entities with respect to a variety of target languages and of arbitrary granularity. Components can reference other components to model aggregation or specialization. WebComposition is based on a prototype-instance OO-model, cf. [9]. Prototyping is a mechanism to implement code sharing among objects. Another possibility to share the code of a component is to allow multiple references on the same component. Sharing is fundamental for reuse and for maintainability as it helps keeping modifications local.

3 The WebComposition Markup Language (WCML)

The WebComposition Markup Language is an application of the eXtensible Markup Language (XML). WCML enables engineers to reuse design and code fragments based on the WebComposition approach by providing a simple notation that is capable of defining objects and their relationships.

Components described in WCML reside in a WCML document, which we refer to as a virtual component store, in conformity to the WebComposition system described in [6]. The following element of the WCML-DTD describes the structure of a component:

```
<!ELEMENT      component
              (prototype*, property+)>
```

Each *component* is identified by a universally unique identifier (UUID) and consists of prototypes and properties that constitute its behavior. With the *prototype* declaration every component can be used as prototype, in this case the state and behavior of an instantiated component is inherited. WCML supports ordered multiple inheritance. The order is implied in the enumeration of the prototype components. *Properties* are name/value pairs binding a value to a name. The following example demonstrates the usage of these declarations:

```
<COMPONENT uuid="uuid-1">
  <PROPERTY name="a" value="1"/>
</COMPONENT>
<COMPONENT uuid="uuid-2">
  <PROTOTYPE is="uuid-1"/>
  <PROPERTY name="b">2</PROPERTY>
</COMPONENT>
```

The above example demonstrates the use of WCML by defining two components with a prototype relationship.

Using prototypes easily allows reuse of components and separation of design and content.

To keep the language easy and orthogonal further properties are introduced. The *content*-property implements the presentation of a component by defining code of the target language or references to other properties within the property tags. The modeled component-content will be mapped to a file using the UUID of the component as filename. With the *filename*- and *directory*-properties the WCML compiler can be forced to create human readable directory and file names. A property may also define relationships between components on a conceptual level. The Compiler maps these relationships to the corresponding hypertext links. The benefit of modeling the hypertext links in this way is the possibility to define the links outside the layout components respectively the composed document and thus to redefine a navigation structure without modifying any components.

4 Processing WCML

A WCML document is processed by the WCML compiler mapping the described components of the Virtual Component Store to the Web implementation model or a target language. The WCML compiler is implemented in Java using an arbitrary XML parser. The components are provided by WCML documents (XML files) accessible through the file system, a database system, or a Web server.

The WCML document is parsed by an XML parser. The resulting parser-tree is then evaluated using a resolution algorithm to resolve the relationships of the components. It is obvious here that the WCML-system remains transparent for the existing Web infrastructure. After parsing, the content-property with its references and strings is processed to generate the presentation output. In the final step, the compiler creates the target resources with the filenames given by the UUIDs of the components, if not otherwise specified by filename and directory properties.

5 Conclusion and Future Work

Web development suffers from the coarse-grained Web implementation model that makes it hard to map fine-grained design entities to the implementation model. Due to the tremendous progress in Web technology and the Web model itself, it is difficult to reuse code and design for cost reduction and quality improvements. We introduced the WebComposition Markup Language that enables the object-oriented specification of Web content. In the WebComposition model, a site with its different entities is composed in terms of components of arbitrary granularity. Components can capture the entities that are basic units for more complex patterns, cf. [8], but that are hidden in resources in a standard Web implementation. One instance of such patterns is different views on the content that can be modeled following the idea of the decorator design pattern [5].

Further work on the support for reuse in Web Engineering aims at the development of an open WCML component repository. We also investigate on design pattern for hypermedia applications and their integration in the repository to support the development process of Web applications.

The WCML-compiler and further information about WebComposition are available from:

<http://www.teco.edu/~gaedke/Webe/>

Bibliography

- [1] R.A. Barta and M.W. Schranz. *JESSICA: an object-oriented hypermedia publishing processor*. In Computer Networks and ISDN Systems 30(1998), Special Issue on the 7th Intl. World-Wide Web Conference, Brisbane, Australia, April 1998, 239-249.
- [2] F. Coda, C. Ghezzi, G. Vigna and F. Garzotto. *Towards a Software Engineering Approach to Web Site Development*. In Proceedings of 9th International Workshop on Software Specification and Design (IWSSD). April 16-18 1998, Ise-shima, Japan.
- [3] M. Gaedke, M. Beigl, H.W. Gellersen: *Mobile Information Access: Catering for Heterogeneous Browser Platforms*. Workshop on Mobile Data Access at the 17th International Conference on Conceptual Modeling (ER98), Singapore, November 16-19, 1998.
- [4] M. Gaedke, H.W. Gellersen, A. Schmidt, U. Stegemüller, W. Kurr. *Object-oriented Web Engineering for Large-scale Web Service Management*. Thirty-Second Annual Hawaii International Conference On System Sciences (HICSS-32) on the Island of Maui, USA, January 5 - 8, 1999.
- [5] E. Gamma, R. Helm, R. Johnson and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1994.
- [6] H.-W. Gellersen, R. Wicke and M. Gaedke. *WebComposition: an object-oriented support system for the Web engineering lifecycle*, Computer Networks and ISDN Systems 29 (1997), Special Issue on the 6th Intl. World-Wide Web Conference, Santa Clara, CA, USA, April 1997, p. 1429-1437.
- [7] A. Kristensen. *Template resolution in XML / HTML*. In Computer Networks and ISDN Systems 30(1998), Special Issue on the 7th Intl. World-Wide Web Conference, Brisbane, Australia, April 1998, p. 239-249.
- [8] F. Lyardet, G. Rossi and D. Schwabe. *Using Design Patterns in Educational Multimedia applications*. In Proceedings of ED-Media'98, World Conference on Educational Multimedia and Hypermedia, Freiburg, Germany, June 1998.
- [9] D. Ungar and R.B. Smith. *Self: The Power of Simplicity*, OOPSLA '87 Proceedings, p. 227-42, 87.

About the Authors

All authors share the following address:

Telecooperation Office (TecO), University of Karlsruhe

Vincenz-Prießnitz-Str. 1, 76131 Karlsruhe, GERMANY

<http://www.teco.edu/>

Ph.: +49 (721) 6902-79, Fax: +49 (721) 6902-16