From Adaptation Engineering Towards Aspect-Oriented Context-Dependency

Sven Casteleyn[1], Zoltán Fiala[2], Geert-Jan Houben[1,3], Kees van der Sluijs[3]

[1]Vrije Universiteit Brussel
Pleinlaan 2
1050 Brussels, Belgium
+32 2 629 33 08
{Sven.Casteleyn, Geert-Jan.Houben}@vub.ac.be

[2]Dresden University of Technology
Chair for Multimedia Technology
01062 Dresden, Germany
+49 351 463 38516
Zoltan.Fiala@inf.tu-dresden.de

[3]Technische Universiteit Eindhoven
PO Box 513, 5600 MB Eindhoven, The Netherlands
+31 40 2472733
{g.j.houben,k.a.m.sluijs}@tue.nl

ABSTRACT

The evolution of the Web requires to consider an increasing number of context-dependency issues. Therefore, in our research we focus on how to extend a Web application with additional adaptation concerns without having to redesign the entire application. Based on a generic transcoding tool we illustrate here how we can add adaptation functionality to an existing Web application. Furthermore, we consider how an aspect-oriented approach can support the high-level specification of such additional concerns in the design of the Web application.

Categories and Subject Descriptors

D.2.11 [**Software Engineering**]: Software Architectures – *Languages*; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces – *User-centered design*; H.5.4 [**Information Interfaces and** Presentation]: Hypertext/Hypermedia – *Architectures, Navigation*.

General Terms

Design, Languages

Keywords

Web Engineering, Aspect-Oriented Programming, Adaptation, Component-Based Web Engineering.

1. INTRODUCTION

The fast evolution and growing popularity of the WWW necessitate to take various issues into account which were previously less relevant: device-dependence, privacy, security, accessibility, personalization etc. While all these issues require the application to exhibit a certain user- or context-dependency, existing methods typically provide only support for a few of them. Furthermore, this support is often intertwined with both the original application design (most often in the form of adaptation conditions spread over the different models) and its corresponding implementation. This is in sharp contrast with the desired separation of concerns and also significantly complicates *adding* additional adaptation concerns to a Web application.

In this paper we focus on extending an existing Web application with new (adaptation) functionality without completely redesigning/reimplementing it. Based on the key observation that the generation of adaptive hypermedia applications is typically implemented as a series of data transformations, we illustrate how we can add adaptation to a Web application based on the Generic Adaptation Component (GAC [1]). We exemplify this in the context of an example from the model-driven Hera design methodology [3]. Next to that, we indicate how an aspect-oriented approach can support the high-level specification of additional (design) concerns in Web application design.

2. APPLICATION SCENARIO

Based on the model-driven Hera specification framework, our running example consists of (a part of) a research project's Web application. Figure 1 depicts its structure according to the visual representation of a Hera application model (see [3]). The starting page is the project homepage showing the project's name, its introductory project description and the project members' photos as thumbnails. Clicking on a photo one can navigate to the corresponding member's homepage containing the name, contacts, CV, image, as well as a list of his publications. In this basic application model there are no adaptations embedded yet.

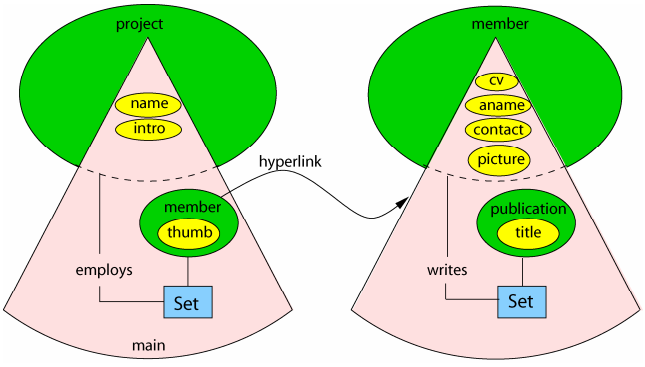

Figure 1: Example Application Model

For our example, we consider one additional adaptation issue, namely device-dependency. It implies to omit high-quality graphical material, as well as detailed information (e.g. the list of publications) on small-screen devices such as PDA's.

WWW 2006, May 23–26, 2006, Edinburgh, Scotland.
ACM 1-59593-323-9/06/0005.

3. IMPLEMENTATION OF ADDITIONAL ADAPTATION CONCERNS

For the implementation of additional adaptation issues we exploit recent work from the AMACONT project that followed the key observation that generating adaptive hypermedia presentations is typically implemented as a series of data transformations. It introduced the Generic Adaptation Component (GAC), a transcoding tool aimed at adding adaptation to Web applications. According to its rule-based configuration, it allows to perform instance-level adaptation operations (omission, inclusion, sorting, replacement etc.) on arbitrary XML input based on context information maintained in its adaptation context data repository. Thus, it is ideal for adding adaptation to the hypermedia presentation generation for XML-based content. For more information on the GAC the reader is referred to [1].

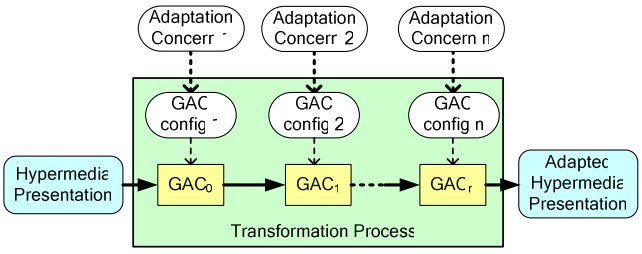

Figure 2: GAC-based Implementation

Based on a number of GAC components, Figure 2 illustrates our data transformation pipeline aimed at the implementation of additional adaptation issues. The original hypermedia presentation generation process is extended by a series of GACs that each perform a different adaptation on their XML-based input content. Each GAC is configured by a number of adaptation rules according to one of the given (independent) additional adaptation concerns.

Based on this implementation, Figure 3 shows two versions of a project member's homepage from our running example. Note that the generated presentations incorporate the device-dependency concerns mentioned above.

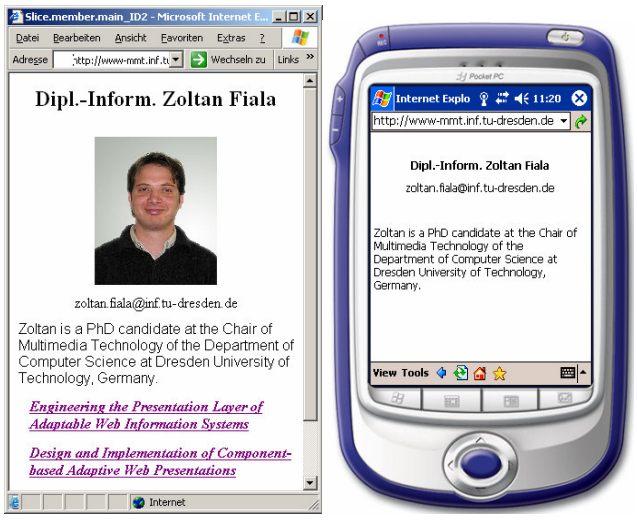

Figure 3: Generated Member Page

4. TOWARDS ASPECT-ORIENTED ADAPTATION DESIGN

The GAC-based implementation architecture illustrated above allows to easily incorporate additional (independent) adaptation concerns into the hypermedia presentation generation process. Still, whereas the GAC supports powerful adaptation operations on XML input at instance level, the complexity of Web applications, and the typical distribution of adaptation throughout the application, necessitates the high-level specification of such adaptations at design level. Thus, extra support is needed to easily extend an application design with additional context-dependent design concerns.

Currently, adaptation is in most design methods specified in the form of *conditions* that are embedded (intertwined) in the relevant design models. Extending a design with one particular context-dependency design concern therefore requires that the designer transforms the current design (model) and embeds for relevant design elements the new adaptation condition(s) that result in the desired context-dependency. Though these conditions can occur at one specific place in the design (e.g. to remove a link between two concrete pages), it (more) frequently happens that they cannot be pinpointed to one particular element (e.g. to hide for privacy reasons all sensitive data) and need to be applied at distributed places in the design (model).

A similar observation was made in the programming community, when considering different design concerns of a software application: some concerns cannot be localized to a particular class or module; instead they are inherently distributed over the whole application. Such a concern is called a *cross-cutting* concern. To cleanly separate the programming code addressing this concern from the regular application code, Aspect-Oriented Programming [2] was introduced. An aspect captures the functionality of a crosscutting concern and can be applied at different parts of the application.

Therefore, our ongoing work concentrates on applying principles of Aspect-Oriented Programming to Web design, thus separating a given Web application design from the specification of additional context-dependency design concerns. If we can easily add such functionality, we can cleanly separate additional design aspects and describe them independently from the base application. We have observed that by translating high-level design aspect descriptions to appropriate GAC rules (according to the above described implementation), we are able to automatically generate a component-based implementation.

5. REFERENCES

[1] Fiala, Z., Houben G.J. A Generic Transcoding Tool for Making Web Applications Adaptive. In Proceedings of the CAiSE'05 FORUM, Porto, Portugal, 2005, 15-20.

[2] Kiczales, G., Lamping, J., Mendhekar, A., Maeda, C., Lopes, C.V., Loingtier, J., Irwin, J. Aspect-Oriented Programming. In Proceedings of the 11th European Conference on Object Oriented Programming (ECOOP'97), Jyväskylä, Finland, 1997, 220-242

[3] Vdovjak, R., Frasincar, F., Houben, G.J., Barna, P. Engineering Semantic Web Information Systems in Hera. Journal of Web Engineering, Vol. 2, No. 1&2, 2003, 3-26.