Considering Additional Adaptation Concerns in the Design of Web Applications

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

[1] Vrije Universiteit Brussel, Pleinlaan 2, 1050 Brussels, Belgium
{Sven.Casteleyn, Geert-Jan.Houben}@vub.ac.be
[2] Dresden University of Technology, Chair for Multimedia Technology, 01062 Dresden, Germany
Zoltan.Fiala@inf.tu-dresden.de
[3] Technische Universiteit Eindhoven, PO Box 513, 5600 MB Eindhoven, The Netherlands
{g.j.houben, k.a.m.sluijs}@tue.nl

Abstract. The design of Web applications traditionally relies heavily on the navigation design. The Web as it evolves now brings additional design concerns, such as omni-presence, device-dependence, privacy, accessibility, localization etc. Many of these additional concerns are occurrences of user- or context-dependency, and are typically realized by transformations of the application (design) that embed adaptation in the navigation. In this paper we focus on how to extend an application with new functionality without having to redesign the entire application. If we can easily add functionality, we can separate additional design concerns and describe them independently. Using a component-based implementation we show how to extend a Web application to support additional design concerns at presentation generation level. Furthermore, we demonstrate how an Aspect-Oriented approach can support the high-level specification of these (additional) design concerns at a conceptual level.

1 Introduction

Part of the popularity of the Web is due to its omni-presence and accessibility: mobile devices (mobile phones, PDA's etc.) make the Web accessible 'anywhere and anytime'. A broad range of organizations, e.g. in e-government, e-health, e-commerce, e-learning, currently offer (part of) their services through this omni-present WWW. In this evolution, designing and creating a Web application has become an increasingly complex task. Not only are the applications larger, they also have to take into account various (design) issues which were previously irrelevant: device-dependence, privacy, security, accessibility, localization, personalization etc. Many of these design issues require exhibiting a certain user- or context-dependency: the application becomes adaptive (to the user). When designing applications and taking into account all these separate issues, ad-hoc design obviously becomes unfeasible.

Web application design methods (e.g. Hera [1], OOHDM [2], WebML[3], WSDM[4], OO-H[5]), proposed in literature as a systematic and conceptual approach to Web application design, offer design and implementation (generation) support for engineering complex Web applications. Adaptation of data, navigation, and presentation that suits the current user/context is specified by incorporating conditions in the relevant design models. This adaptation specification constitutes a significant part of the design of the Web application. Furthermore, it is closely intertwined with the (regular) design process, which complicates the Web engineering process and is in sharp contrast with the desired separation of concerns.

In this paper we first illustrate how to add adaptation to an existing (Hera-based) Web application, using a component-based implementation. We do so utilizing the Generic Adaptation Components (GAC [6]) provided by the AMACONT project, and exemplify this approach with a running example using Hera [1]. Furthermore, we illustrate how this adaptation can be specified at a higher-level of abstraction (i.e. at the design level), separately from the regular application, by applying aspect-oriented techniques. Using aspect-oriented techniques allows us to tackle cross-cutting design concerns, which adaptations typically are.

2 Implementing Additional Adaptation Concerns

Before explaining how to add adaptation to an existing Hera-based Web application, it is wise to recall the general architectural model behind such a Web application: under the direction of an application model (AM) it transforms domain content into a hypermedia presentation. When adding adaptation to such an application, each additional adaptation concern can be seen as a modification of the application, and thus, at implementation level all these additional adaptation concerns have to be incorporated into the overall hypermedia generation process (see Figure 1).

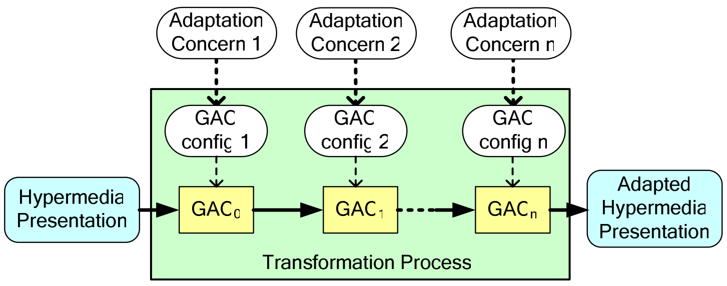


Figure 1: Implementing Additional Concerns at Presentation Generation

For this implementation of additional adaptation concerns we exploit the Generic Adaptation Component (GAC), a transcoding tool for making Web applications adaptive. According to its rule-based configuration, it allows to perform instance-level adaptation operations on XML-based content (in our case the original hypermedia presentation underlying the initial application model) based on available user/context information is stored in its *adaptation context data* repository. The GAC is thus a particularly suitable solution for implementing adaptation for hypermedia presentation generation, independent of the regular application design. To illustrate this approach, consider the running example of (a part of) a research project's Web application.

Figure 2 depicts the structure of the application according to the visual representation of a Hera AM. The starting page is the project homepage showing the project's name, its introductory project description and the project members' name and photos as thumbnails. Clicking on a member, one can navigate to the corresponding member's homepage containing the name, contacts, CV, image, as well as a link list showing the publications (title, conference, year of publication). Note that in this basic application model there is no adaptation embedded yet.

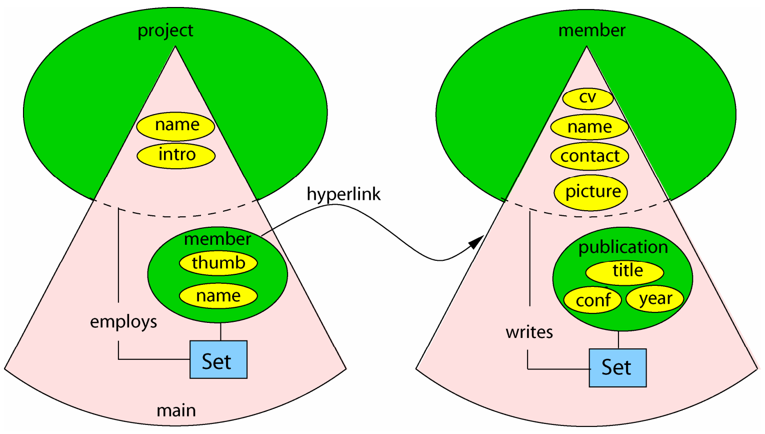


Figure 2: Example Application Model

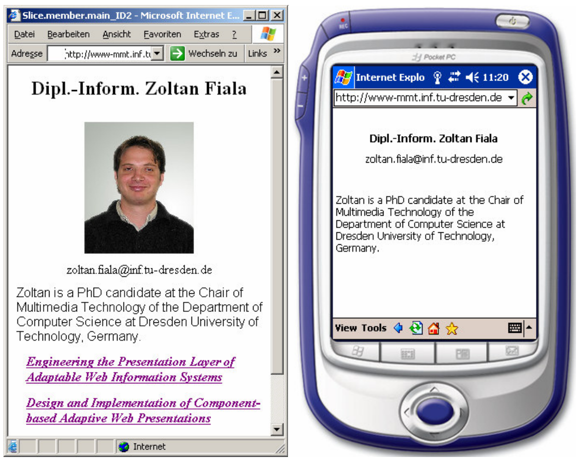
Now imagine we would like to express, in the context of an additional adaptation concern of device-dependency, that for pda-users no images will be shown, and on the member page, no publication-list will be given. The corresponding GAC rule for filtering images (which in this case will omit members' thumbs on the project page, and members' pictures on each member's page) looks as follows:

```
<gac:AppearanceRule gac:selector="//Slice/* [@mediatype = 'picture']">
   <gac:Condition gac:when="$device!=‘pda’"/>
</gac:AppearanceRule>
```

Figure 3 shows two versions of our running example's project member homepage. Based on the application model shown in Figure 2, the original (desktop) variant provides information of that member's title, name, CV, contacts but also a link list pointing to his publications. According to the above GAC rule, no pictures are shown on the PDA. Furthermore, the list of the member's publications is only shown on the desktop browser and filtered out on the handheld.

3 Specifying Additional Adaptation Concerns at Design Level

Using the GAC to specify additional adaptation concerns allows to specify adaptation separately from the regular Web application. However, this solution operates at the level of presentation generation, and requires detailed knowledge of the specific XML formats of the Web design method's implementation models (in our case, the Hera implementation models). Furthermore, the richness of these models is a determining factor for the specificity or generality of the GAC rules. A higher-level solution, which allows the designer to specify additional adaptation concerns at the design level exploiting their implementation independent semantics, yet still separate from the regular design, is thus desirable. Therefore, we will apply aspect-oriented techniques at design (model) level (see also [7] for a similar, yet lower level approach). From such a higher-level specification, one or more GAC rules for the specific XML (implementation) format can then be generated.



Slice.member.main_ID2 - Microsoft Internet E...
Datei Bearbeiten Ansicht Favoriten Extras ?
Adresse http://www-mmt.inf.tu
Wechseln zu
Links
Internet
Pocket PC
Internet Explo 11:20
http://www-mmt.inf.tu-dresden.de
View Tools

Dipl.-Inform. Zoltan Fiala

zoltan.fiala@inf.tu-dresden.de

Zoltan is a PhD candidate at the Chair of Multimedia Technology of the Department of Computer Science at Dresden University of Technology, Germany.

Engineering the Presentation Layer of Adaptable Web Information Systems

Design and Implementation of Component-based Adaptive Web Presentations

Dipl.-Inform. Zoltan Fiala

zoltan.fiala@inf.tu-dresden.de

Zoltan is a PhD candidate at the Chair of Multimedia Technology of the Department of Computer Science at Dresden University of Technology, Germany.

Figure 1: Generated Member Page

As our "omit all pictures" example suggests, adaptation is, in most cases, not fixed to one particular element, yet distributed over the entire design. 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 [8] was introduced. An *aspect* captures the code originating from the cross-cutting concern, and is split up in two parts: the *advice* (i.e. the programming code that needs to be injected in the regular code) and the *pointcut* (i.e. the particular place(s) in the regular code where the advice needs to be injected).

Returning to our running example, and applying aspect-oriented principles at the design level of a Web Application, we can express the required adaptation "for pda-users, remove all images from the application" as follows (using pseudo-code):

POINTCUT ALL ELEMENTS WITH mediatype = 'picture' **PROPERTY**
ADVICE ADD CONDITION device != 'pda'

This pointcut/advice pair specifies that, for all design elements which are pictures (the pointcut), a condition is added which restricts visibility of these element to nonpda users only. Such an aspect-oriented specification can subsequently be (automatically) translated in one or more corresponding GAC rules, which perform the desired adaptation at implementation level, as described in the previous section. Note that both at design and implementation level, we strive for a separation of adaptation (concerns), from the regular design/application.

4 Conclusion

In this paper, we demonstrated how to add (additional) adaptation concerns to an existing (Hera-based) Web application, both at design and implementation (generation) level. Our approach is based on the observation that adaptation mostly boils down to transformations that realize a certain user- or context dependency. We showed how these transformations can be specified independently from the original application at presentation generation level, by using a generic transcoding tool GAC. We furthermore illustrated how higher-level support for this adaptation specification can be realized applying aspect-oriented techniques. We are currently experimenting with such aspect-oriented adaptation specifications, and with the required mapping to GAC rules.

References

[1] 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.

[2] Schwabe, D., Rossi, G.: The Object-Oriented Hypermedia Design Model. In Communications of the ACM 38(8), ACM Press, ISSN 0001-0782 (1995) 45-46

[3] Ceri, S., Fraternali, P., Bongio, A., Brambilla, M., Comai, S., Matera, M.: Designing Data-Intensive Web Applications, Morgan Kaufmann (2003)

[4] De Troyer, O., Casteleyn, S.: Designing Localized Web Sites. In Proceedings of the 5th International Conference on Web Information Systems Engineering (WISE 2004), Brisbane, Australia (2004) 547-558

[5] Gómez, J., Cachero, C., Pastor, O.: Conceptual Modeling of Device-Independent Web Applications. In IEEE Multimedia Special Issue on Web Engineering, IEEE Computer Society Press (2001) 26-39

[6] 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

[7] 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

[8] Bausmeister, H., Knapp, A., Koch, N., Zhang, G. Modelling Adaptivity with Aspects. In Proceedings of the International Conference on Web Engineering (ICWE2005), Sydney, Australia, 2005, 406-416