A Survey on Web Modeling Approaches for Ubiquitous Web Applications

Andrea Schauerhuber[1,*], Wieland Schwinger[2], Werner Retschitzegger[3], Manuel Wimmer[4], and Gerti Kappel[4]

[1]Women's Postgraduate College for Internet Technologies, Vienna University of Technology
schauerhuber@wit.tuwien.ac.at

[2]Department of Telecooperation, Johannes Kepler University Linz
wieland.schwinger@jku.ac.at

[3]Institute of Bioinformatics, Johannes Kepler University Linz
werner@bioinf.jku.at

[4]Business Informatics Group, Vienna University of Technology
{wimmer|gerti}@big.tuwien.ac.at

1. INTRODUCTION

Today's web applications are full-fledged, complex software systems for which a methodologically sound engineering approach is crucial. Web engineering has emerged as an independent branch of software engineering and "comprises the use of systematic and quantifiable approaches in order to accomplish the specification, implementation, operation, and maintenance of high quality web applications" [Kappel et al. 2006]. During the past 10 years, academia has provided various web modelling approaches, each aiming at counteracting a technology-driven and ad hoc development of web applications. These web modeling approaches originally have emerged as proprietary languages rather focused on notational aspects. As the types of web applications have evolved over time so have the web modeling approaches come up with appropriate concepts for them. Thus, increasingly more web modeling approaches are supporting the development of so called Ubiquitous Web Applications (UWAs), i.e., Web applications that adhere to the anytime/anywhere/anymedia paradigm, taking into account that services in the web are nowadays not exclusively accessed through traditional desktop PCs but through mobile devices with different capabilities and constraints, by users with various interests and goals at anytime from anyplace around the globe. Services provided by UWAs, consequently, need to be adapted to the actual context of use in order to preserve or even enhance their semantic value for users. Thus, on the one hand, capturing the context, e.g., user, location, time, and device to understand the situation of use is of importance. On the other hand, and probably even more important is to provide appropriate adaptation operations to adjust the Web applications with respect to their different kinds of contents, e.g., text, images, and links, as well as the offered navigation structure and presentation. Thus, context and adaptation in combination are the main prerequisites for customization of web applications towards ubiquity denoting the mapping of the required adaptation of an application's services with respect to its context.

Several web modeling approaches have recognized this new requirement by providing new modeling concepts that capture customization functionality. Additionally, the rise of Model-Driven Engineering (MDE) already has had impact on current web modeling languages. Consequently, some of the approaches are supported with a modeling tool and possibly code generation facilities and have also aimed at providing for a model-driven development in the sense of MDE, i.e., on the basis of MDE techniques and technologies including metamodels and model transformations.

Based on previous work [Schauerhuber et al. 2007], this paper is dedicated to present the state-of-the-art in model-driven development of UWAs and throw some light on the current limitations and strength of some of the most prominent modeling approaches. More specifically, an in-depth comparison of seven web modeling approaches currently supporting the development of UWAs is provided. This conducted by applying, on the one hand, a detailed set of evaluation criteria and, on the other hand, by demonstrating their applicability with respect to a web application example in terms of a tourism web application. This includes five commonly found customization scenarios, thus providing initial insight into the concepts for modeling customization of each approach as well as to facilitate their comparability.

For this, in Section 2, the evaluation set-up is presented, i.e., the selection of web modeling approaches as well as a detailed and well-defined catalogue of evaluation criteria used for the structured comparison of the approaches, including the description of the example and customization scenarios used in the comparison. Based

* This research has been partly funded by the Austrian Federal Ministry for Education, Science, and Culture, and the European Social Fund (ESF) under grant 31.963/46-VII/9/2002.

on this, in Section 3, a per-approach evaluation is presented including a detailed description of the realization of the customization examples in the respective approaches. An extensive report on lessons learned is given in Section 4, pointing out the approaches' strengths and shortcomings. Thereafter, in Section 5, existing related work and the contributions of this survey are discussed. Finally, in Section 6, the paper is closed with a brief summary.

2. EVALUATION SET-UP

This survey's goal is primarily to provide a study of existing web modeling languages having as specific focus the model-driven development of UWAs. In particular, the focus is on design level concepts for modeling customization as well as on tool support for the model-driven development of UWAs. As a consequence, other possible strengths of the approaches under investigation that lie beyond customization, e.g., support for workflow-based web applications, will not be considered herein.

2.1 Selection of Approaches

With respect to the selection of evaluation candidates, in literature, several well-established and well-published web modeling languages having different origins and pursuing different goals can be found. In Schwinger et al. 2006, a categorization of fourteen approaches into data-oriented, hypertext-oriented, object-oriented, and software-oriented web modeling approaches has been proposed (cf. Figure 1).

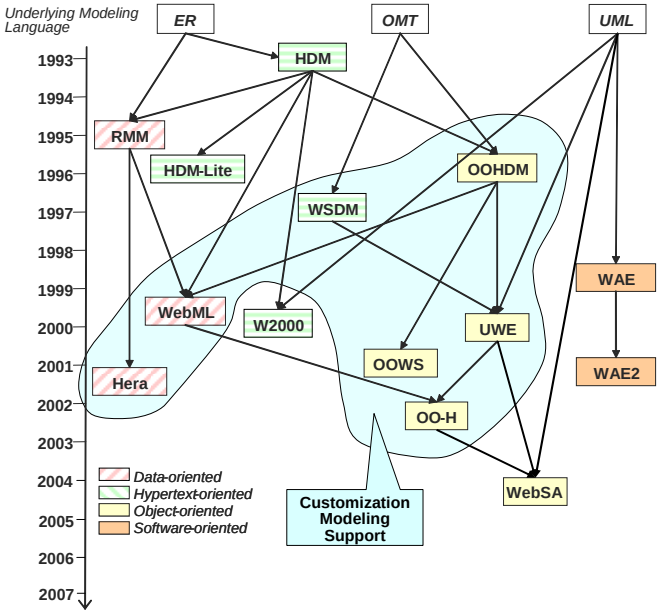


Figure 1. Overview on Web Modeling Approaches

Still, since focused on customization modeling, in this survey, only web modeling approaches that also provide concepts for customization modeling and thus enable to appropriately model UWAs will be considered. In the following, seven out of the fourteen approaches depicted in Figure 1 will be investigated. From the data-oriented category, the WebML [Ceri et al. 2003] and the Hera [Frasincar et al. 2006] approaches, from the hypertext-oriented category, the WSDM [Troyer and Leune 1998] approach, and from the object-oriented category the OOHDM [Rossi and Schwabe 2006], the UWE [Koch and Kraus 2002], the OO-H [Gomez et al. 2001], and the OOWS [Pastor et al. 2006] approaches are evaluated. All of them can be considered to be well-established since each approach has been published in more than 25 publications including reviewed papers, books, and manuals.

2.2 Catalogue of Evaluation Criteria

In the following, a catalogue of criteria for the structured evaluation of web modeling approaches is proposed, having a particular focus on criteria for evaluating the support for customization modeling and for model-driven development. The criteria are, on the one hand, the result of a top-down approach considering the four dimensions of web application development (cf. [Schwinger et al. 2006]) and on the other hand the result of a

bottom-up approach taking into account interesting issues from related surveys as well as from previous work [Kappel et al. 2001], [Kappel et al. 2003], [Schwinger and Koch 2006].

The overall emphasis of the catalog of criteria is on functional criteria. Since this survey is based on an in-depth study of literature of the approaches' documentation, the inclusion of non-functional criteria in terms of several ”-ilities” such as evolvability, scalability, traceability, reusability, understandability, maintainability, or flexibility is not considered within the scope of this evaluation. Still, this survey paves the ground for a later evaluation in real-world projects allowing to investigate the web modeling approaches with respect to the aforementioned ”-ilities”. Nevertheless, such an evaluation in real-world projects would also raise currently unaddressed questions associated with empirical evaluations in web engineering, e.g., how to get an unbiased set-up for the evaluation including control groups.

Aiming at a solid definition of criteria, for each criterion applied in this survey its name and a definition along with the appropriate measures are given. The abbreviation of the criterion allows for referencing it during evaluation of the approaches in Section 3.

Furthermore, these criteria are grouped into five categories with three out of them being inferred from corresponding functional requirements and two additional categories, one providing general criteria on the *Maturity* of an approach and the other one providing criteria related to the *Tool Support* of an approach. The catalogue of criteria is presented along with its categories in the following.

2.2.1 Maturity

The maturity of an approach is characterized by the following criteria.

Topicality (G.T). This criterion provides for each approach the year of introduction as well as when the most recent piece of work has been published in order to indicate whether the approach is still under development or not.

Modeling Examples (G.ME). Another indication for the maturity of an approach is the number of different modeling examples discussed. Admittedly, besides evaluating the number of existing examples, their depth would also be of interest. Such a depth measure could be composed of the number of modeling concepts used, i.e., the number of content classes, nodes, links, etc. Still, this is not feasible, since often only parts of the examples are shown in available literature, or the examples have been simplified for readability purposes.

Application in Real-World Projects (G.A). Another indication for a high level of maturity of an approach is its employment in designing real-world applications. This criterion evaluates whether real-world applications exist or do not exist.

2.2.2 Web Modeling

The *Web Modeling* category covers criteria for evaluating the dimensions of web application development, namely levels, features, and phases. Note that, this category does not focus on the specific support for ubiquity terms of customization, for which criteria can be found in a separate category.

Web Application Levels (W.L). This criterion indicates which web application levels (i.e., content, hypertext, and presentation) are considered by an approach and which formalisms/types of diagrams are employed.

Interfaces (W.I). How the interrelationships between the web application levels are modeled, is indicated by the *Interfaces* criterion. In case the interface specification is defined separately from the levels, this criterion names the mechanism used for content-hypertext and hypertext-presentation interfaces in terms of notations, natural language, or text-based query languages. If an interface specification is not defined separately but as part of one of the two levels in question, this criterion additionally evaluates to intermingled.

Feature Modeling (W.F). For each web application level, this criterion investigates if modeling of structural and/or behavioral features of web applications are supported by the web modeling approach or not.

Development Phases (W.Ph). This criterion checks which phases of web application development, i.e., requirements elicitation, analysis, design, and implementation, are supported by the evaluated approaches.

Development Process (W.Pr). The extent to which a developer is supported by a development process is covered by the *Development Process* criterion. Specifically, it distinguishes whether a well-defined development process is a proprietary one or is based on a standard development process, e.g., the Rational Unified Process (RUP) [Kruchten 2000]. Furthermore, it lists the detailed steps, output artifacts, and involved actors.

2.2.3 Customization Modeling

The Customization Modeling category explicitly deals with characteristics of the customization dimension in web application development. This includes criteria investigating support for modeling context information as well as the necessary adaptations. The following criteria are based on previous work [Kappel et al. 2003] but specifically focus on the modeling level.

Context Properties (C.P). Although the relevant kind of context is specific to each UWA, a web modeling approach should support a set of common context properties including user, location, device, time, and network. Consequently, this criterion evaluates if the approach supports explicit concepts for modeling context and context properties, and what context properties have been used in modeling examples illustrating the approach.

Context Extensibility (C.CE). It is required that built-in modeling concepts for context properties can be easily extended by additional ones in case a UWA needs further context information (e.g., temperature). This criterion, thus evaluates to supported or not supported.

Chronology (C.C). This criterion tells if the approach offers (or does not offer) concepts that allow modeling how contextual information changes over time. For example, considering video streaming, information on how the bandwidth changed in the past can be used to infer how the bandwidth will develop or how stable it can be considered in the future in order to be able to tune the resolution of the video accordingly.

Complex Context (C.CC). Complex context information is aggregated using different context properties, e.g., "Vienna at night". In this respect, this criterion evaluates if an approach supports or does not support appropriate modeling concepts to specify complex context.

Separation of Context (C.SC). Customization modeling should ensure the separation of context information and not just intermingle context with adaptation or the web application itself, i.e., usually the content model. This criterion tests whether an explicit representation, e.g., in terms of a separate context model, would allow for reusability of already defined context information across several UWAs.

Adaptation Operations(C.O). This criterion evaluates if the approach supports predefined modeling concepts for adaptation operations, e.g., filter some content, add links, change resolution of an image, change hypertext, etc.

Adaptation Extensibility (C.AE). Similar to the required extensibility of modeling concepts for context properties, this criterion evaluates whether adaptation operations can be extended by user-defined adaptation operations. The criterion evaluates to supported or not supported.

Complex Adaptations (C.CA). Complex adaptations define multiple adaptation operations performed on the same or on different subjects, e.g., adapting appearance for visually impaired persons may require resizing and color adaptation in combination. This criterion evaluates if an approach provides or does not provide appropriate means of modeling complex adaptations.

Adaptation of Levels (C.L). Customization influences potentially all levels of web applications. Content adaptation changes the information that is provided to the user by, e.g., adding or removing/filtering content, hypertext adaptation changes the navigation structure by, e.g., disabling a link or foreseeing different access structures, and presentation adaptation changes the way information is presented to the user by, e.g., changing colors or modality. Thus, this criterion investigates for which web application levels concepts to model adaptations are offered by an approach.

Adaptation of Interfaces (C.I). Customization may also influence the interfaces between levels, e.g., queries to underlying web application levels may need to change due to a certain context. Consequently, this criterion evaluates if modeling of adaptations with respect to interfaces is supported or not supported.

Granularity (C.G). The granularity of adaptation indicates the number of modeling concepts affected by a certain adaptation. While micro adaptation is concerned with fine-grained adaptations by affecting a single application element only (e.g., disabling a specific link on a certain page), macro adaptation means that rather large parts of a model are adapted, thus affecting multiple modeling concepts (e.g., changing the language or changing the prevailing access structures).

Separation of Adaptation (C.SA). Customization modeling should ensure the separation of adaptation modeling and thus, prevent an intermingled representation of adaptations within the content, hypertext, and presentation levels. This criterion tests whether separation of adaptation is supported and thus, allows for reusability of already defined adaptations within the same or across several UWAs, or if it is not supported.

Customization Phases (C.CP). This criterion evaluates during which development phases an approach considers customization modeling. Ideally, customization is considered during all development phases.

2.2.4 Model-Driven Engineering

The *Model-Driven Engineering* criteria focus on modeling language definitions, model transformations, and platform descriptions as a prerequisite for successfully employing MDE.

Language Definition (M.L). This criterion evaluates if a web modeling language has been defined explicitly in terms of a metamodel (including UML profiles), a grammar, a semantic description in terms of semantic web technologies, or if such a definition is absent.

Model Transformation Types (M.T). The investigated approaches might support or not support various types of model transformations such as transformations between platform-independent models (PIM2PIM), transformations between platform-independent and platform-specific models (PIM2PSM), transformations between platform-specific models and code (PSM2Code), and transformation from platform-independent models to code directly (PIM2Code).

Platform Description Model (M.P). This criterion evaluates if the information about a platform is represented separately within a platform description model, or if this information is implicitly captured within the transformation rules.

2.2.5 Tool Support

For those approaches offering tool support, this category of criteria provides detailed information about the tool.

Tool Base (T.B). This criterion checks if the tool is developed as a stand-alone application or as a plug-in/extension to an existing tool.

Tool Openness (T.O). Whether the offered tool support can be tailored to the developer's needs or not, is tested by this criterion. Either the tool explicitly foresees extension possibilities or the tool's source code is publicly available under an opensource license and thus can be changed.

Version (T.V). This criterion records the current version of the tool at the time of evaluation in order to give a rough estimation about the developed status of the tool in terms of revision cycles.

Costs (T.C). A tool might be free of charge or require a license fee to be paid. This criterion evaluates to either freeware or commercial.

Modeling Support (T.M). A tool supporting a specific approach can be evaluated if web modeling and/or customization modeling is supported or not supported.

Model Pre-Generation Support (T.MG). Furthermore, a tool might also assist modeling of a developer by pre-generating (parts of) models to be refined later on. This criterion evaluates to true or false.

Consistency Check (T.CC). The criterion tests if a tool has a built-in consistency check to verify the correctness of the models or not.

Code Generation (T.CG). Tools may generate code from the models the developer has defined. The criterion evaluates to true, if a code generation facility is available, otherwise it evaluates to false.

Process Support (T.P). The kind of guidance through the development process within the tool is evaluated by this criterion. Firstly, it indicates whether the process supported by the tool is realized for the approach as described in literature. Secondly, this criterion details if the developer is bound to a step-wise procedure, or has the possibility to go back and forth between the development steps without loss of information (i.e., wizard-like), or can freely choose where to start and what to model as long as all phases are processed.

Collaboration (T.Co). The evaluated tool might support or might not support version control and thus, allow collaboratively working on a project in a team.

2.3 Modeling Example: A Tourism Information Web Application

In order to support the textual comparison on the basis of a structured set of criteria, an example is provided, which is modeled by means of the concepts of each web modeling approach. The example further enhances the evaluation in that it first, provides an initial insight into each approach and second, facilitates the comparison of the approaches' modeling means in terms of their notation, especially with respect to modeling customization. It has to be emphasized, however, that the role of the modeling example in this survey is of a supportive nature,

only. It is not intended to give a comprehensive introduction to all of a web modeling language's features. In particular, the example shall provide insight to the approaches' means for modeling customization functionality.

2.3.1 Running Example

The web modeling field lacks commonly excepted reference modeling examples, which can be used to "assess" individual approaches, and in particular lacks modeling examples including comprehensive customization. Nevertheless, a first attempt has been conducted at the International Workshop on Web-Oriented Software Technologies (IWWOST 2001)[1] by introducing a Conference Management System. The second attempt initiated at the Workshop on Model-driven Web Engineering (MDWE 2005)[2] proposes a Travel Agency. Most recently, [Rossi et al. 2007] applies the example of a movie database to allow a comparison of different web modeling approaches but not specifically focusing on customization issues. Furthermore, in the course of surveying the different web modeling approaches, it became obvious that some modeling examples have been used particularly often across several web modeling approaches, amongst them library systems, e-stores, and art galleries.

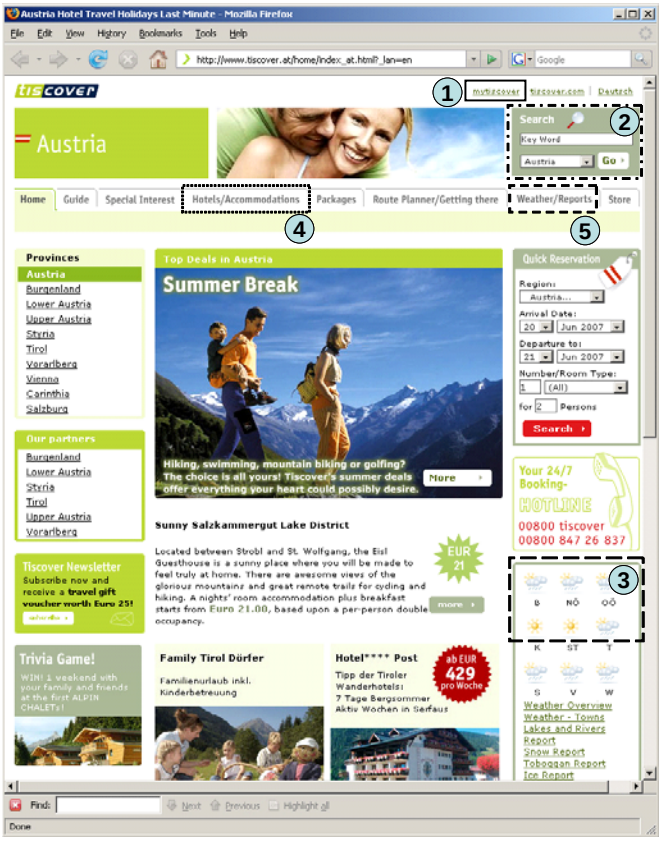


Figure 2. The Tiscover Start Page

The main reason for not adopting one of these examples for this survey is that the modeled web applications typically are limited with respect to their ubiquitous nature. If at all, customization functionality is very simple, e.g., encompassing content adaptations with respect to an isolated context factor such as the user, the device, or the location, only. Instead, the aim of this survey was to choose an example from a domain with sufficient customization potential while possibly not creating any biases, e.g., due to reusing existing examples in some of the investigated approaches.

The domain of tourism information systems is able to provide such examples requiring customization, especially, when considering accessibility through mobile devices as is the case for mobile tourism guides [Grün et al. 2006]. Consequently, a hypothetical Tourism Information Web Application (TIWA), which has been

[1] http://www.dsic.upv.ex/~west/iwworst01/
[2] http://www.lcc.uma.es/~av/mdwe2005

inspired by Tiscover[3], the official Austrian tourism platform, has been chosen as the running example in this survey.

2.3.2 Basic Functionalities

In the following, a short description of the requirements for our TIWA is given, briefly laying out the basic functionalities of a typical tourism information system which is necessary to be able to understand the modeling of the customization scenarios throughout this survey. The essential use cases are illustrated in Figure 2.

In the course of this example, guest users, i.e., unauthenticated users, shall be enabled to browse and search for hotels, regions, activities as well as information about the weather. Registered users shall be allowed to book rooms and to browse their prior bookings. Administrators shall be responsible for the web application's content and therefore have to execute typical CRUD operations for hotels, regions, users, activities, and bookings.

How such a web application could look like, is indicated with the screenshot mock-up in Figure 3, showing a possible home page of the TIWA – based on the Tiscover web-site - with dedicated areas for (1) login, (2) search, and (3) weather information. Furthermore, the menu allows navigating to, e.g., (4) hotels as well as more detailed (5) weather information.

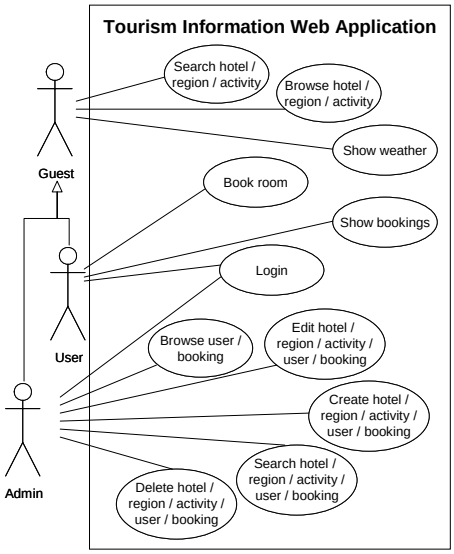


Figure 3. Use Cases of the Tourism Information Web Application

Thus, the TIWA shall allow users to browse the offer of hotels either for special regions or as the result of the user's search. These lists of hotels will provide essential information, e.g., the hotel's name, a picture, a short description, a quick link for booking, etc. Whichever hotel the user picks leads to an according detail page, allowing to obtain all the information available for the chosen hotel as well as the possibility to start a booking process. Likewise, the user will be able to browse regions as well as activities and request more information on either subject by following links to the according detail pages. Furthermore, if already logged in to the TIWA, the user is able to book a room and will be able to browse all prior bookings. Administrators have a special interface to create, edit or delete hotels, regions, activities, as well as bookings and users. Consequently, the content of the web application encompasses information on users, bookings, hotels and their special features, rooms, regions as well as activities.

2.3.3 Customization Scenarios

TIWAs naturally give space for a series of customization possibilities to better support user experience. In the realm of the TIWA, five customization scenarios which have been designed to support the catalogue of criteria, are laid out in the following as the supposed system's customization functionality.

(1) Customized Activities: Users will have the possibility to view activities that are particularly relevant to their needs. The activities (C.L) made available to the user shall be filtered according to the user's age, location, the current time, and the weather at the user's location (C.P), (C.CE). Similar scenarios, representing

[3] http://www.tiscover.at

personalization, could filter content according to the user's preferences, only, e.g., list all hotels according to the user's interest.

(2) Special Offers: Users should enjoy special offers based on their navigational behavior (C.P). They shall be able to navigate to their special offers, if they have visited several pages of a specific region (C.CC). Thus, this scenario describes adaptation of the hypertext level (C.L). Alternatively, a user that has already booked three times will get 3% off the price for the next reservation, which represents a content level adaptation (C.L).

(3) Administrator Links: Administrators (C.P) shall be provided with "edit" links for every concept of the content they are allowed to edit (C.L), allowing for an additional and possibly easier way of content management. For normal users or visitors, these links will not be available requiring fine-grained hypertext adaptation (C.G).

(4) Multi-Delivery: In order to consider users equipped with small devices (C.P), an appropriate adaptation of the system shall be chosen. If a small-screen device is used, pictures and detailed descriptions shall not be shown. This can be achieved by creating a dedicated hypertext model for each device type which represents a coarse-grained hypertext adaptation (C.G) or by hiding content for devices with small displays, which represents a fine-grained content adaptation (C.L).

(5) Current Season's Style: The look & feel of the web application shall be different according to the season. Different styles shall be chosen during summer and winter, respectively. The look & feel might also be subject to customization for supporting users having special needs, e.g., visually impaired users (C.L).

3. COMPARISON OF APPROACHES

In the following,the seven selected web modeling approaches will be compared applying the catalogue of criteria presented in Section 2.2. The description of each approach will be examplified using the modeling example described above (cf. Section 2.3). Each of the following sections is dedicated to one approach. The approaches are ordered along the categories they belong to, starting with the category of data-oriented web modeling approaches.

3.1 The Web Modeling Language (WebML), Ceri et al.

3.1.1 Maturity

WebML [Ceri et al. 2003] is one of the well elaborated web modeling languages stemming from academia and is supported already over several years by the commercial tool WebRatio[4]. Furthermore, WebML and WebRatio have already been successfully employed in several real-world projects (G.A). WebML is explained by a comprehensive number of different modeling examples including various forms of e-Stores selling different kinds of products, a conference management system, an online travel agency, a loan brokering web application, a museum guided tour, a campus tour, and an e-learning application (G.ME). Since its introduction in 1999, WebML has continuously evolved and recently has been extended by additional concepts addressing context-aware [Ceri et al. 2007], service-enabled [Manolescu et al. 2005], workflow-based [Brambilla et al. 2006], and semantic [Brambilla et al. 2006] web applications as well as rich internet applications [Bozzon et al. 2006] (G.T).

3.1.2 Web Modeling

The WebML language as presented in [Ceri et al. 2003] provides modeling concepts for content modeling and hypertext modeling, only. While the content level resembles the well-known ER-model [Chen 1976] (cf. Figure 4(a)), at the hypertext level a proprietary graphical notation is provided (cf. Figure 4(b)) (W.L).

Within the WebRatio tool there are available additional means for configuring the presentation level (e.g., for defining the web page's stylesheet and the positioning of information on a web page) which, however, are not part of the WebML language [Ceri et al. 2003]. The interface between content and hypertext level is specified graphically by denoting the entity of the content level (e.g., *Hotel* in Figure 4(a)) to be displayed in a so-called content unit (e.g., Hotels in Figure 4(b)) as well as in a textual representation to specify additional properties. In Listing 1, the textual representation of the *IndexUnit HotelList* of the *Hotels* page in Figure 4(b) is given.

[4] www.webratio.com

```
IndexUnit HotelList
(sourceHotel;
attributes Name , Picture , Description;
orderby Name)
```

Listing 1. WebML: Textual Representation of the IndexUnit *HotelList*

Since specified at the hypertext level, however, the interface cannot be modeled separately (W.I). Additionally, the approach allows behavioral modeling to a limited extent, only. While UML activity diagrams are used during the requirements specification phase, some behavioral features are represented in the hypertext model by the control-flow-like semantics of WebML's operation units (cf. *ModifyUnit* and *ConnectUnit* in Figure 5). Recently, the introduction of process modeling concepts into WebML [Brambilla et al. 2006], as well as the introduction of customization modeling concepts [Ceri et al. 2007], allows further modeling of behavioral aspects at the hypertext level (W.F). The WebML approach includes its own seven-phase, iterative, and incremental development process based on Boehm's Spiral model [Ceri et al. 2003] (W.Pr) comprising all development phases from requirements to implementation (W.Ph).

3.1.3 Customization Modeling

According to available literature, the WebML approach provides two proposals for modeling customization. The first approach introduces context-aware pages [Ceri et al. 2007], whereas the second approach is based on event-condition-action-rules [Ceri et al. 2006] for which it is, however, not clear how they are related and how they can be used together. Both approaches have in common that they do not provide an explicit context model, but represent context information within the content model.

In the first proposal, for dealing with UWAs [Ceri et al. 2007], however, the modeler is supported in designing the context information with guidelines proposing to imagine the content level as a set of overlapping sub-schemas (C.SC). For illustration purposes, these sub-schemas can be indicated in the content model (cf. Figure 4(a)). The entities of the, so called, *Basic User Sub*-Schema are always available in a WebML model and consists of the User, Group, and Module entities. The so called *Personalization* Sub-Schema associates the *User* entity with other entities to denote, e.g., user preferences whereas the so called *Context Model Sub*-Schema associates the *User* entity with other context information, e.g., the user's location, the device used, etc. (C.CE). As a form of static adaptation, WebML siteviews, i.e., several hypertext models defined upon the content level, on the one hand, may be used to personalize a web application according to users and user groups and, on the other hand, serve the purpose of expressing alternative forms of content presentation for different devices [Ceri et al. 2003] (C.P). Modeling complex context is not explicitly supported within WebML (C.CC). For modeling context-awareness, the idea of a context-aware web page (cf. Activities Page in 5) and concepts for retrieving context information have been introduced. For example, for retrieving context information from the content level, the *GetDataUnit* concept is employed (cf. *GetRegion* GetDataUnit Figure 5), while the *GetClientParUnit* concept is used to obtain context information from the client (cf. *GetLatitude* GetClientParUnit Figure 5), e.g., information on the user's location or on wireless connectivity [Ceri et al. 2007] (C.P). For modelling adaptation, two predefined adaptation operations are available, namely *ChangeSiteview* (cf. Figure 7(a)) and *ChangeStyle* (cf. Figure 7(b)) (C.O), which allow for a coarse-grained adaptation of the hypertext and the presentation level, respectively (C.L), (C.G). The possibility of changing the navigation flow, e.g., through WebML's *IfUnit* (cf. Figure 6(b)), enables more fine-grained adaptations. There is no way to explicitly adapt the interface between the content and the hypertext level (C.I), since adaptations are defined for a page rather than for a content unit. It is not possible to extend the predefined set of adaptation operations (C.AE), neither is there a concept for defining complex adaptations (C.CA). In WebML, UWAs are modeled as refinements of the models of a non-ubiquitous web application, i.e., the development process has not yet been adapted to guide developers in considering customization functionality throughout the development lifecylce (C.CP). Since adaptations are modeled as an extension of the hypertext model, WebML does not provide a separation of adaptations from the rest of the web application model (C.SA).

In the second proposals for modeling UWAs with WebML, the focus is on personalization purposes and allows defining adaptive behaviour of a web application based on event-condition-action (ECA) rules [Ceri et al. 2006]. The approach relies on adaptive pages that specify the action part of the rule and are similar to context-aware pages. Conditions are described on the basis of a so-called web behavior model (WBM) script, which is a timed state-transition automaton for representing classes of user behaviors on the web, i.e., navigation patterns. Thus, some form of context chronology is supported (C.C) for this approach.

3.1.4 Model-Driven Engineering

WebML's language concepts are partly defined in terms of XML document type definitions (DTDs) (M.L) and partly hard-coded within the approach's accompanying tool, WebRatio. Consequently, the WebML approach currently cannot profit from the benefits of MDE. Nevertheless, two proposals have recently invested some efforts to port WebML to MDE. One of them provides a metamodel for WebML based on the Meta Object Facility (MOF) [Schauerhuber et al. 2007] and another effort provides a UML profile for WebML [Moreno et al. 2007]. While the WebRatio tool provides code generation support (using J2EE and Jakarta Struts) from the platform-independent WebML models, the WebML approach does not provide for model transformations to different platforms prior to generating code (M.T) As a consequence, the approach does neither provide platform description models for the above mentioned platforms (M.P).

3.1.5 Tool Support

WebRatio is a commercial, proprietary tool developed as a standalone application (T.C), (T.O), (T.B). WebRatio 4.3 has been evaluated with an academic license (T.V) which is free of charge (T.C). Currently, WebRatio does not include the concepts for modeling customization, except for providing the User, Group, and Module entities in every web application's content model. Still, it is reported that, the concepts proposed in [Ceri et al. 2007], [Ceri et al. 2007] have been realized in prototype implementations (T.M). The tool allows extending the WebML language via specialized content units and operation units (T.O). This way the WebML language and tool support have already been extended, e.g., to support service-enabled web applications. WebRatio allows generating a running application out of a web application model, with no additional coding. The generated web application can be automatically deployed to an integrated Tomcat Servlet Container (T.CG). Moreover, the tool offers a so called pattern wizard that allows building common parts of the hypertext model automatically, like a login mechanism or content management functionality for selected entities from the content level (T.MG). The tool also allows consistency checks that can be executed on-demand (T.CC). The process of the method is only partly supported by the tool, e.g., requirements engineering is not considered, and the user is allowed to start wherever s/he chooses (T.P). Cooperative work is supported in that the WebRatio tool is also capable of shared editing via a concurrent version system (CVS) (T.Co).

3.1.6 Modeling Example

In Figure 4(a) the content level, i.e., an ER-diagram, of the TIWA example is shown for WebML. In WebRatio, the predefined entities User, Group, and Module enable basic personalization of the hypertext level towards user and user groups (cf. the Basic Sub-Schema indicated in Figure 4(a)). The other entities are representing the web application's data: Hotels have Rooms, which in turn can have multiple Bookings, whereby each Booking belongs to a User (cf. the Personalization Sub-Schema indicated in Figure 4(a)). Hotels have additional Features such as a swimming pool, an animation team or the like. Every Hotel is situated in a Region which in turn has neighboring regions. In every Region there will be an offer of Activities, e.g., in terms of events.

Further context information has been incorporated to the content level as follows: The User is directly related to the device s/he is using. Additionally, the User is also directly associated with the Region s/he is currently located as well as indirectly to the Weather entity that has been introduced to allow for customization according to the current weather situation (cf. the Context Model Sub-Schema indicated in Figure 4(a)).

Part of the hypertext level of the TIWA is shown in Figure 4(b). This particular view shows that users can browse hotels, regions, and activities via three dedicated landmark pages (cf. 'L' label). The specific subjects can then be selected via WebML's *IndexUnits* to be displayed in detail on separate pages, i.e., *HotelDetails*, *RegionDetails*, and *ActivityDetails*. Each of these pages uses a *DataUnit* for presenting the selected item. In addition, the *RegionDetails* page presents an index of the region's activities and hotels, respectively.

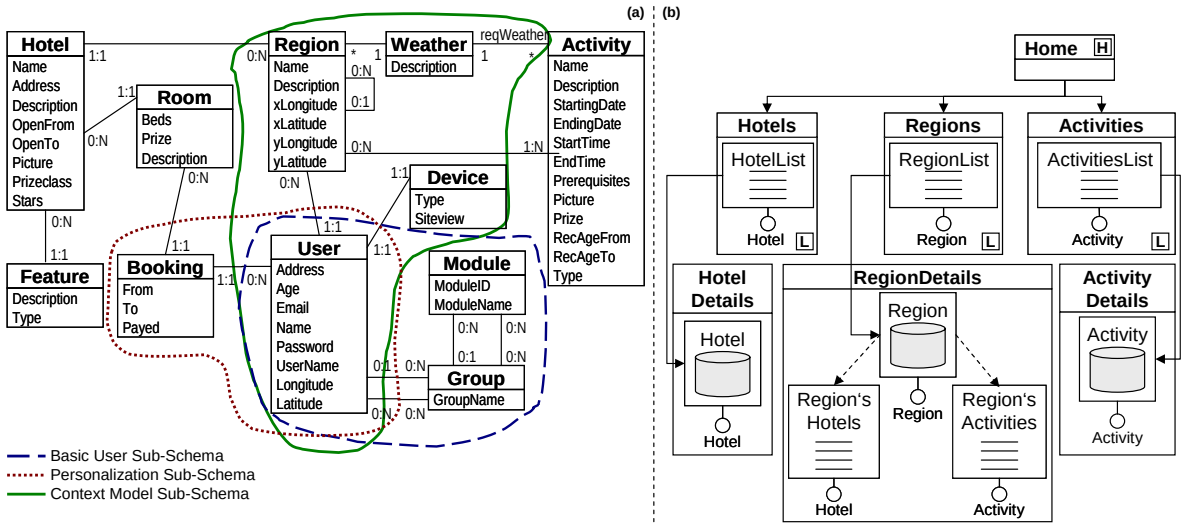


Figure 4. WebML: (a) Content Level, (b) Hypertext Level

Customization Scenario Customized Activities. The *Activities* page can be made context-aware in order to show only activities of the region the user is currently located in and are employed to filter them according to the user's age and the current date (cf. first approach of WebML to address UWAs). A *GetUnit GetUser* acquires the current user and two *GetClientParUnits* provide the user's longitude and latitude (cf. Figure 5). The location information then can be stored for the current user in the content model with a *ModifyUnit*. The location information is also used to select the region the user is currently located with the *GetRegion GetDataUnit*. The *ConnectUnit* is used to relate the user with the current region, i.e., the user's location. A *TimeUnit* which is available also in WebRatio, can be used to obtain the current date so that only those activities are shown that have not yet started. The current weather situation for the user's region is assumed to be updated in the data source by some external service so that it can be queried by the *GetDataUnit GetWeather*. In order to correctly filter the information on activities for the user, all this context information is then used in a set of so-called *SelectorConditions* (cf. square brackets) for the *ActivitiesList IndexUnit* in the page *Activities*.

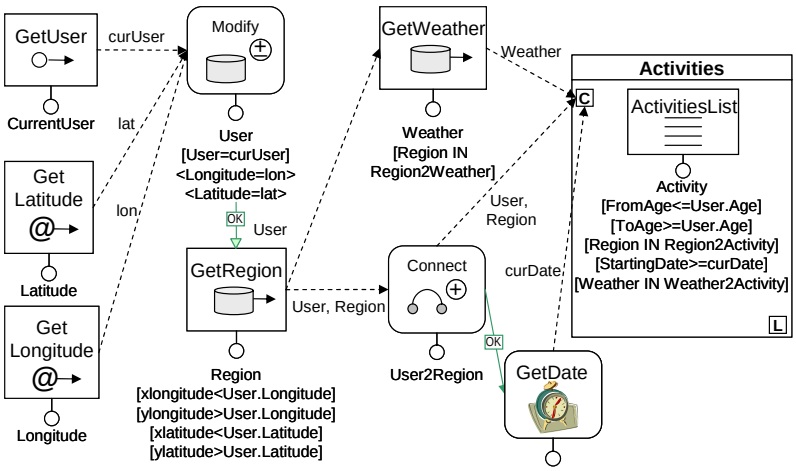


Figure 5. WebML: Customized Activities Scenario

Customization Scenario Special Offers. For modeling the Special Offers customization scenario, WebML's rule-based approach (cf. second approach for modeling UWAs with WebML) is employed, which allows reacting to user navigation. In the following example, the TIWA will monitor the user navigation to find out what region a user is particularly interested in. As soon as this information is available, the user shall be redirected from the *HotelDetails* as well as from the *ActivityDetails* page to a page presenting special offers for hotels and activities. As depicted in Figure 6(a), the condition of this rule is specified by a WBM script, i.e., a timed finite state automaton, stating that a user needs to visit three arbitrary pages (denoted with '*'), having in common that each displays information about the same region. In a WBM script, a state (denoted with a circle) represents the user's inspection of a page. The three pages do not necessarily have to be visited in sequence, i.e., the user can navigate to other pages in between. Still, the transition constraint states that after 180 seconds the

next page displaying the same region information has to be navigated otherwise the script will be discarded. In case the third page is visited, the automaton reaches its accepting state, triggering the action of the rule. In the action part of the rule, the *HotelDetails* and the *ActivityDetails* pages become adaptive pages (A), having a link to the Special Offers page, which will be navigated in case the rule's condition is true.

Customization Scenario Administrator Links. In general, WebML's siteviews can be used for modeling different hypertexts for different user groups, e.g., external users and administrators. To enable editing from the public siteview, however, fine-grained adaptation shall allow/disallow dedicated links in the *ActivityDetails* page of the public siteview, in the following. In Figure 6(b), depending on the current user's group, the *ActivityDetails* page will present different navigation possibilities. The *Alternative* concept is employed to indicate that the normal *Activity DataUnit* is displayed in the default case (cf. 'D' label). Alternatively, an *Activity DataUnit* that has a link to an *EditActivity* page can be displayed. An *IfUnit* is employed to redirect navigation to the respective alternative on the basis of the context retrieved by the *GetUser GetUnit* and the *GetGroup GetDataUnit*.

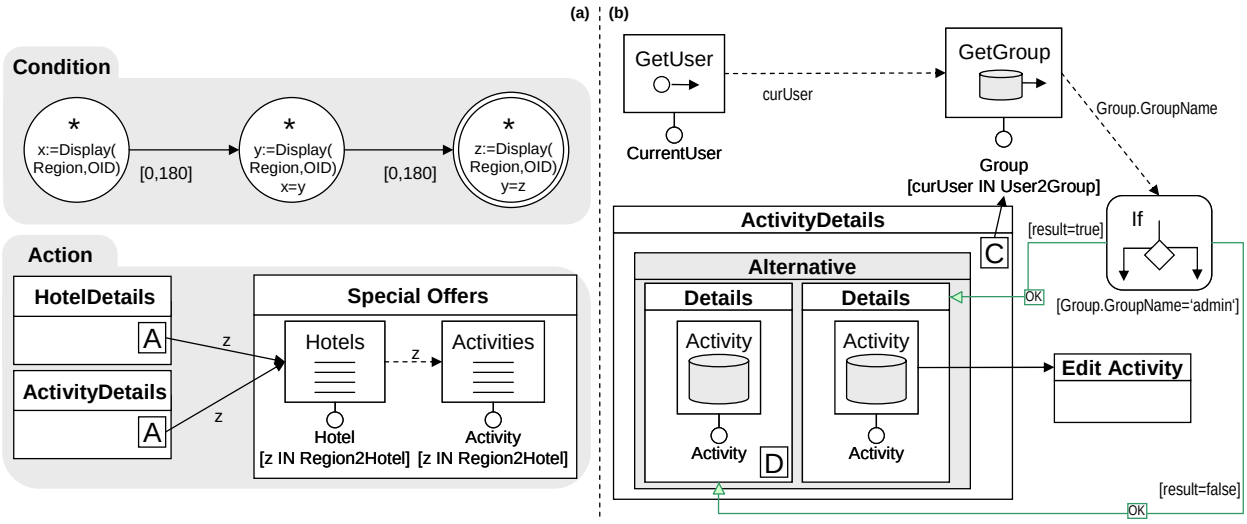


Figure 6. WebML: (a) Special Offers Scenario, (b) Administrator Links Scenario

Customization Scenario Multi-Delivery. WebML also suggests modeling dedicated siteviews for special navigation purposes as is the case for adapting the hypertext level according to the user's device. The *ChangeSiteViewUnit*, used in this scenario, has been recently introduced to the WebML language to allow for changing the hypertext model to better suit the current context of the user. In Figure 7(a), the *PublicSiteview* has been made context-aware. Thus, if a user requests any page located in this siteview, the *GetClientParUnit GetDevice* will provide the necessary information to find a device specification in the database via the *GetDataUnit GetDeviceSpec*. The user will then be redirected to an appropriate siteview that has been associated to the device in the database by the ChangeSiteviewUnit. In case no such specification can be found, the user will be redirected to a page where the device can be chosen from a predefined list.

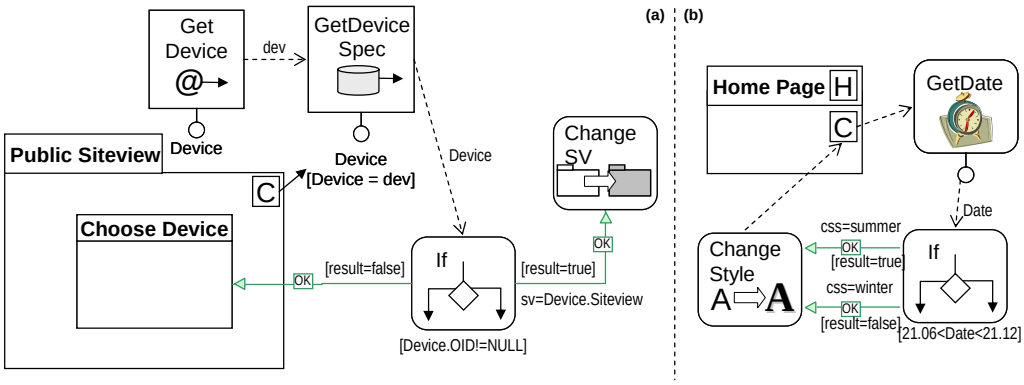


Figure 7. WebML: (a) Multi-Delivery Scenario, (b) Season's Style Scenario

Customization Scenario Season's Style. Finally, WebML does also allow customizing the presentation level with the recently introduced *ChangeStyleUnit*. This coarse-grained adaptation operation changes the look&feel of the web application by changing the Cascading Style Sheet[5] (CSS) used according to the current context. Figure 7(b), shows how the time context is obtained with a *TimeUnit GetDate*. An *IfUnit* is used to decide if the presentation of the home page shall be based on the summer style or the winter style by using the respective stylesheet.

3.2 The Hera Design Methodology, Houben et al.

3.2.1 Maturity

The Hera[6] approach [Frasincar et al. 2006], first introduced in 2000, has been heavily influenced by the hypermedia design method Relationship Management Methodology (RMM) [Isakowitz et al. 1995] and originally concentrated on web applications that, depending on a user query, automatically generate complete, static hypermedia presentations [Barna et al. 2002]. Recently, a revision of the Hera approach has been introduced with Hera-S [van der Sluijs et al. 2006], which considers dynamic web applications (G.T). The prevailing modeling example used in almost all publications is a virtual art gallery. Besides that, a conference management system, a digital photo library, a movie library, and e-store applications are used as modeling examples (G.ME). While it seems that real-world applications of Hera do not exist yet, four demo web applications have been developed including the museum application where real-world content has been reused (G.A).

3.2.2 Web Modeling

The Hera approach, in principle proposes models for the content, hypertext, and presentation levels (W.L). Since RMM has been a major influence to Hera [Frasincar et al. 2001], the content level (i.e. Hera's conceptual model or alternatively called domain model) initially has been based on the ER-model [Chen 1976]. Today, Hera is based on the Resource Description Framework (Schema) - RDF(S) [W3C 2004b], thus supporting the engineering of semantic web information systems. For modeling the content level, it provides a proprietary graphical notation [Frasincar et al. 2006]. At content level, as depicted in Figure 8(a), the domain model of Hera is based on concepts (e.g., Hotel and Region), attributes (e.g., Name and Picture), concept relationships, (e.g., provided by and provides) and media types (e.g., String and Image). The hypertext level (i.e., the application model) and the presentation level (i.e., the presentation model) still resemble RMM's graphical notation as is shown in Figure 8(b) and Figure 9, respectively. More specifically, the Hera application model addressing the hypertext level is mainly based on slices and slice relationships, which can be either compositional or navigational. In the upper part of Figure 8(b), the *HotelDetail* slice has *Hotel* as the owning concept from the content level (cf. the rounded rectangular) and presents the details of a hotel, including its name, picture, prizeclass, stars, description and address. Furthermore, the set of rooms it contains and the features it provides are part of the slice. From the *HotelDetail* slice, a navigational relationship leads to the *HotelEdit* slice. The presentation level is based on a hierarchy of regions allowing the positioning of slices from the hypertext level [Fiala et al. 2004]. A region represents a rectangular part of the display area and has associated with it a layout manager for positioning elements and a specific style. Figure 9 describes how the *HotelDetail* slice and its parts are placed within different regions. For example, the *HotelDetail* slice is placed within a *Box Layout*. The figures show that the interfaces between levels are specified graphically. For example, at hypertext level, a slice always denotes its owning concept from the content level in terms of a rounded rectangular (Figure 8(b)). Moreover, the interfaces between the levels are as well specified textually on the basis of queries [van der Sluijs et al. 2006]. Still, the interface is always intermingled with the upper level (W.I). The recent introduction of concepts for modeling workflow-based web applications (e.g., a task model and a workflow model) [Barna et al. 2006] allow for modeling behavioral aspects at the hypertext level (W.F). Hera proposes a development process that includes development phases from requirements engineering to implementation [Vdovjak and Houben 2002] (W.Pr), however, this process has not been detailed beyond a set of guidelines. Recent elaborations on the approach focus particularly on design and implementation, but do not consider the requirements engineering phase (W.Ph).

[5] http://www.w3.org/Style/CSS/
[6] http://wwwis.win.tue.nl/~hera/

3.2.3 Customization Modeling

In the Hera approach, customization is focused on the user and device context, only (C.P). The approach strives for encapsulating context information separately (C.SC): On the one hand, static adaptations are based on context information obtained from a user profile [Frasincar et al. 2004], [Fiala et al. 2004]. In fact, Hera uses a CC/PP vocabulary [W3C 2004a] to model user preferences and device capabilities. For example, in the profile instance presented in Listing 2, the hardware platform is characterized by the property client, which identifies the device used, while the group property characterizes the user preferences in terms of providing information on the user's group. On the other hand, dynamic adaptations, that are based on a user model storing the users' navigational behavior, have been proposed but not further detailed [Frasincar and Houben 2002]. Moreover, further dedicated concepts are absent for supporting extensibility of context explicitly (C.CE), for capturing context history (C.C), and for modeling complex context information (C.CC). Adaptation is realized by means of so-called appearance conditions attached to different design artifacts, i.e., to the content, hypertext, and presentation models (C.L), (C.I). If such conditions evaluate to true or false, the presence of their associated artifacts, e.g., content information or links (C.G), is enabled or inhibited, respectively [Frasincar et al. 2004]. In this respect, appearance conditions can be seen as a form of adaptation operations (C.O). As a consequence, the approach does neither allow for extensibility of adaptation operations nor for complex adaptations (C.AE), (C.CA). Since adaptations are only annotations to different design models (C.CP), they cannot be modeled separately either (C.SA). Still, in the Hera-S approach the recent introduction of concepts from the aspect-orientation paradigm is intended to provide for such a separation [Casteleyn et al. 2007]. This way, appearance conditions can be modeled separately and "woven" into the models before generating the web application. Another proposal for customization presents how a high-level personalization rule language [Garrigos et al. 2005], can be mapped onto modeling concepts from Hera and OO-H (cf. Section 3.6).

3.2.4 Model-Driven Engineering

As already mentioned before, the Hera modeling language relies on RDF(S) and CC/PP, respectively (M.L). Currently, the approach focuses on the generation of static hypermedia presentations, i.e., static web sites, in a specific output format, e.g., in terms of HTML and WML, which are deployable on any web server. For this, the approach suggests the subsequent transformation or rather the integration of the different design model artifacts, i.e., Hera's conceptual model, application model, presentation model, and user profile model, whereby in a last step, the output can be generated (M.T). The approach does not support a separate platform model (M.P).

3.2.5 Tool Support

Hera is supported by the Hera Presentation Generator (HPG) tool, which has been evaluated in version 1.3 (T.V). HPG is a proprietary tool, not extensible, and available as freeware (T.B), (T.O), (T.C). Support for a concrete notation for modeling the content and the hypertext as well as the presentation level has been realized by three Microsoft Visio[7] templates called *Conceptual Model Builder*, *Application Model Builder*, and *Presentation Model Builder*, respectively [Frasincar et al. 2006]. A load/export feature provides the RDF(S) serialization of the models for the HPG tool. Apart from that, modeling of the user profile, i.e., the CC/PP code, is supported through appropriate "textual" wizards of HPG (T.M). Since HPG 1.3 generates static hypermedia presentations only, it does not allow for modeling dynamic adaptations that consider e.g. the user's inputs or the user's browsing history [Frasincar and Houben 2002]. Today, transforming the models into a suitable static presentation (HTML, WML, etc.) is possible on the basis of XSL transformations using either HPG itself or the external AMACONT engine developed at the Dresden University of Technology [Fiala et al. 2004] (T.MG), (T.CG). The HPG tool follows the guidelines for developing a web application with Hera by starting with the content level, i.e., Hera's domain model. A wizard guides the user through the design and generation process (T.P). In every step the models can be viewed using a text editor or Microsoft Visio, where the model builders enforce some constraints while the user is designing a model in order to produce correct models. Moreover, the models' consistency can also be checked by HPG (T.CC). Up to now, there is no shared editing or versioning support (T.Co). Recently, modeling support for user input and tool support for generating dynamic web applications, which are able to react to user input, has been proposed in [Houben et al. 2004] and [Frasincar et al. 2006], respectively. This new tool support is currently implemented in Java and provides a runtime environment for the generated applications on the basis of the Java-Servlet platform.

[7] http://office.microsoft.com/en-us/visio/

3.2.6 Modeling Example

In the Hera approach, not all customization scenarios can be realized. Since the approach is limited to user and device context information as well as to appearance conditions which do not allow filtering of content, it is possible to present the Multi-Delivery and the Administrator Links scenarios, only.

Customization Scenario Multi-Delivery. The content level as well as the other levels can be customized with annotations, i.e., appearance conditions such as *prf:imageCapable = true* in Figure 8(a).

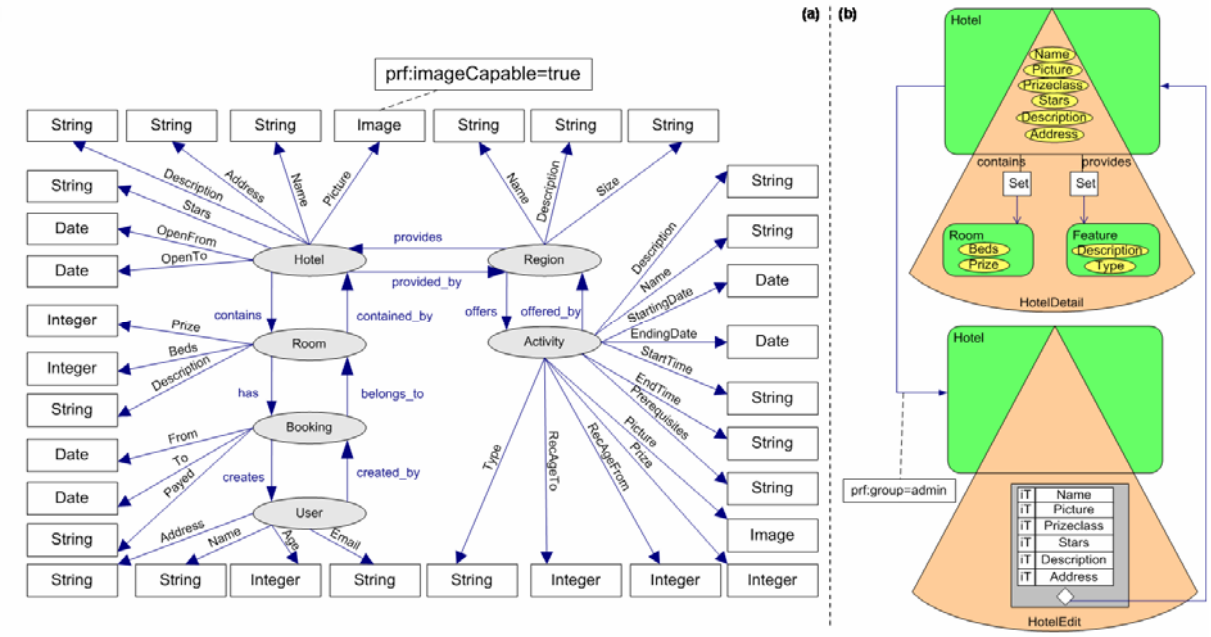


Figure 8. Hera: (a) Conceptual Model, (b) Application Model

This appearance condition is specified on the basis of the user profile instance of Listing 2 and means that the conceptual model is to provide a picture of the hotel only if the device used is capable of presenting images. Since according to the client property in the profile instance, the *imageCapable* property is set to true, the hotel's picture will remain in the domain model and thus in the generated hypermedia presentation.

With respect to the presentation level, appearance conditions can also be used to choose one of a set of alternative regions according to the current device. In Figure 9, the set of rooms of the *HotelDetail* slice are placed at the bottom part of the display (cf. the regions *RegionBottomRooms*). More specifically, two alternatives for presenting the hotel's set of rooms are provided depending on the device used.

```
<Description rdf:about= "Profile" >
    <ccpp:component>
        <HardwarePlatform>
                <imageCapable>true</imageCapable>
                <client>Desktop</client>
        </HardwarePlatform>
    </ccpp:component>
    <ccpp:component>
        <UserPreferences>
                <group>admin</group>
                . . .
        </UserPreferences>
    </ccpp:component>
```

Listing 2. Hera: User Profile Instance

The appearance conditions associated with the two alternative regions *RegionBottomRooms* state, that in case of a PDA, the rooms also shall be displayed according to a *Box Layout*, while when using a PC users shall be presented with a *Table Layout*.

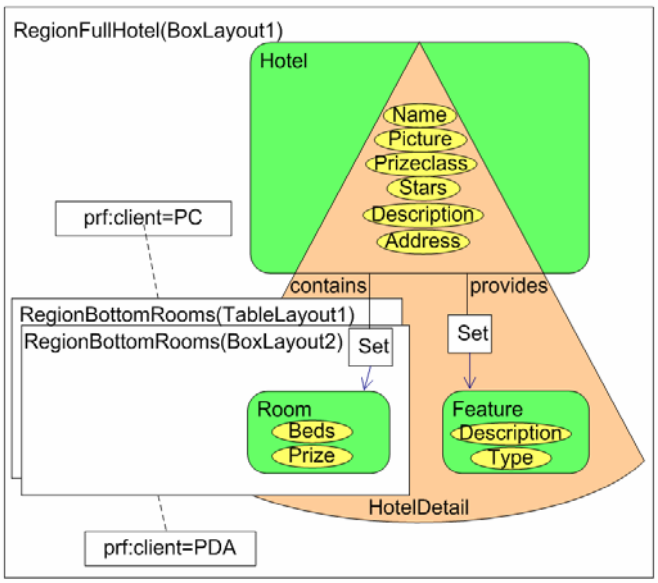


Figure 9. Hera: Presentation Model

Customization Scenario Administrator Links. Similarly, appearance conditions can be used to customize the hypertext level, e.g., a link in the Hera application model can be adapted according to the user profile. Figure 8(b) presents a small excerpt from the application model showing how the Administrator Links scenario can be realized. The link from the *HotelDetail* slice to the *HotelEdit* slice is annotated with the appearance condition *prf:group = admin* stating that only users of the group 'admin' will be able to see this link. According to the group property in the profile instance presented in Listing 2, the user is part of the admin group and thus, the edit link will be available. Another way of hypertext adaptation is the possibility to define multiple application models for the content level. In this respect, the application models or rather the RDF(S) specifications just need to be exchanged and the web application can then be re-generated. Please note that appearance conditions also can be associated with a whole slice or one of its shown attributes, e.g., the hotel's description.

Following, Listing 3 indicates how such an appearance condition can be modeled separately as is proposed in [Casteleyn et al. 2007] using the textual Semantics-based Aspect-oriented Adaptation Language (SEAL) of Hera-S. In the example below, the *ADVICE* specifies that an appearance condition needs to be added to the hypertext level, while the *POINTCUT* specifies the exact place(s). Please note, that in Hera-S the modeling concepts of the language partly have been renamed, e.g., unit corresponds to a slice and the reference to the user profile 'prf' has been changed to 'cm'. Consequently, the *POINTCUT* specifies the link from the *HotelDetail* slice to the *HotelEdit* slice. The advantage of this approach is that within the *POINTCUT*, several links of the hypertext level can be specified in one place and extended with an appearance condition.

```
POINTCUT type relationship and
    from (type unit and hasName "HotelDetail") and
    to (type unit and hasName "HotelEdit")
ADVICE ADD CONDITION cm: group = "admin"
```

Listing 3. Hera: Separating Appearance Conditions with Aspects

3.3 The Web Site Design Method (WSDM), De Troyer et al.

3.3.1 Maturity

The Web Site Design Method[8] (WSDM) has been first proposed in 1998 [Troyer and Leune 1998] and thus, is one of the earliest web modeling approaches. Recently, WSDM is evolving towards the semantic web [Casteleyn et al. 2006] (G.T). There seem to be no applications of the approach in real-world projects (G.A). Modeling examples used are a conference management system, an e-store, and a department website (G.ME).

[8] http://wsdm.vub.ac.be/Research/publications.php

3.3.2 Web Modeling.

WSDM provides its own five-phase, audience-driven development process (W.Pr) and specifies all its modeling concepts within the WSDM Ontology[9] which is based on the Web Ontology Language (OWL) [W3C 2004c]. The process starts from a mission statement, specifying purpose, subject and targeted users, followed by a user-driven requirements engineering and analysis phase, which results in the audience model, i.e., a set of audience classes, having specific requirements and characteristics. During the design phase, structural modeling of all web application levels is supported (W.L). The content level is represented as the integration of object chunks to the so-called business information model. Each object chunk represents a tiny conceptual schema, which is derived from the tasks each audience class needs to perform. In turn, these tasks are identified in a task modeling activity based on the Concurrent Task Tree (CTT) technique [Paterno 2000], whereby a task is decomposed into elementary tasks arranged in a temporal order (W.F) (cf. Figure 10). Then, for each elementary task an object chunk is created, which (formally) models the necessary information and functionality needed to complete the task (cf. Figure 11). Moreover, an object chunk can have associated object chunk functions, which allow to model system functionality (e.g., instance creation and selection functions) [Casteleyn et al. 2006], [Plessers et al. 2005] (W.F). Formerly, object chunks have been based on the Object Role Modeling (ORM) method[10] [Halpin 2001]. Today, they are specified within the WSDM Ontology, while the graphical notation still follows ORM (cf. Figure 11). At the hypertext level, WSDM's navigational model (W.L) consists of navigation tracks, one for each audience class, and models the conceptual structure of the web application in terms of nodes and links (cf. Figure 12). A navigation track can be considered as a "sub-site" dedicated to the information and navigation needs of the audience class. Nodes can comprise several object chunks from the content level, which is specified with the OWL object property *hasChunk*. This relationship is also graphically represented in the navigation track (cf. rounded rectangles in Figure 12) (W.I). At presentation level, the site structure model (W.L) maps the concepts modeled at the hypertext level onto pages, i.e., the OWL object property *hasNode* decides which nodes and links will be grouped onto web pages. Nodes to be presented on one page are surrounded with a rectangle shape (indicated for the *Show Bookings* node in Figure 12) (W.I). In addition, WSDM proposes page models defined for each separate page in the site structure model, which allows for positioning of page elements. The mapping of object chunks onto the actual data source is performed in a data source mapping step. Finally, the development process ends with the implementation of the web application (W.Ph).

3.3.3 Customization Modeling

In WSDM's history, two independent proposals for dealing with customization modeling have been published. The first one [Casteleyn et al. 2005], [Casteleyn et al. 2003] focuses on modeling adaptive navigation based on rules specified at design time and triggered due to the browsing behavior of users (C.P). The second one [Troyer and Casteleyn 2004], basically extends each of WSDM's development phases (except implementation) in order to model localization. To do so, the concept of "locality" has been introduced describing a particular place, situation, or location (C.P). In particular, the audience modeling phase has been extended to model localities, i.e., their specifications, characteristics, and their mappings to audience classes. In further design models, localization is considered by annotation-like extensions to the models, which implies no separation of adaptations (C.SA). Context information, i.e., locality, is actually modeled at the content level within the locality labels. Consequently, there is no separation of context from the rest of the application (C.SC). Context cannot be extended to support further context properties (C.CE), and neither chronology (C.C) nor complex context can be modeled (C.CC). In [Casteleyn et al. 2003], the Adaptation Specification Language (ASL) has been defined, allowing designers (C.CP) to express rules specifying when and how the hypertext level (C.L), needs to be adapted according to the monitored user behavior. In this respect, the approach offers an implicit context model, providing several functions to query data on user behavior, e.g., *numberOfVisits* of a node and *numberOfTraversings* of a link. This way, some form of context chronology can be realized (C.C). The approach offers a fixed set of primitive adaptation operations (C.O), (C.AE) on nodes (i.e., *deleteNode* and *addNode*), connections between nodes and object chunks (i.e., *connectChunk* and *disconnectChunk*) (C.I), and links (i.e., *deleteLink* and *addLink*). More complex adaptations based on the primitive ones are amongst others *promoteNode* / *demoteNode*, which moves a node closer to / further away from the root of a web site, thus making it easier / harder to find (C.CA). Rules in ASL can be applied to single elements but also to groups of elements (C.G) and represent a mechanism to specify adaptations separately from the rest of the web application (C.SA). The approach suggests that these design time rules can be used to generate rules for an adaptation

[9] http://wise.vub.ac.be/ontologies/WSDMOntology.owl
[10] http://www.orm.net

engine within the web application but does not explain possible effects of navigation model rules on the presentation level.

3.3.4 Model-driven Engineering

As already mentioned before, all modeling concepts of WSDM have been defined within the WSDM Ontology. WSDM models thus, represent instances of this ontology (M.L). The WSDM approach foresees an implementation phase with a four-step transformation process, respectively a five-step process when considering semantic annotations [Casteleyn et al. 2006]. In this process, the previously defined models are supposed to be subsequently transformed to the selected output platform, e.g., (X)HTML and WML (PIM2Code). During this process, the different models are integrated into one, thus realizing a PIM2PIM transformation (M.T). The approach, however, does not provide platform description models for the above mentioned platforms (M.P). A prototype for generating code and particularly semantic annotations from WSDM's design model has recently been described in [Casteleyn et al. 2006]. This prototype currently is able to generated HTML code based on XSLT.

3.3.5 Modeling Example

On the basis of WSDM's current customization mechanisms, it is possible to model the Customized Activities scenario, the Administrator Links scenario, and a variant of the Special Offers scenario. The customization scenarios Multi-Delivery and Season's Style could possibly be realized by defining different variants of the hypertext and presentation levels for supporting different devices as well as for different seasons. Still, how the different variants are selected in the web application at runtime is assumedly left to implementation. Consequently, these customization scenarios have not been realized.

Customization Scenario Customized Activities. In WSDM, the activities can be filtered according to the current context at the content level. The content level, i.e., the object chunks, is derived from a CTT, specifying the details of an audience class's task. The task of browsing activities is rather simple. Consequently, for illustration purposes, an example for the more complex booking task of the user audience class is specified by the CTT in Figure 10: To book a room, a room has to be found and after that, a reservation has to be made. To find a room, a hotel must be found first. Either before or after that, the user has to login and can then select a room that s/he likes, etc. Some of the object chunks capturing the information and functional requirements of an elementary task are depicted in Figure 11. The *Select Room* object chunk shows the information requirements for the *Select Room* task of the CTT shown in Figure 10. The input to the object chunk is an instance '*h' of the hotel object type for which name and picture are shown as the only value types. From the set of rooms of the hotel which is denoted with brackets '{*r}', one can be selected as is indicated with '!'. The price value type of the room is tagged with locality labels to indicate information that is dependent on the locality, i.e. the price is different for localities 'G' for Germany and 'US' for United States. The object chunk *Book Room* creates a new instance (annotated by the keyword 'NEW') of booking for a specified room and date, which are inputs from prior tasks, namely Select Room and Provide Date. The relationships are established with the '+' notation. The object chunk *Browse (Customized) Activities* of Figure 11, shall present customization scenario Customized Activities. In the example, the location of the user is assumed to be resolved to a region and to be stored in the data source. Thus, having the user '*u' as input to the object chunk, the region for the user as well as the region's activities can be selected. Furthermore, the activities are selected according to the current date with the keyword 'TODAY'. Please note that, the WSDM specification actually does only allow for testing the equality of values with '==', thus the example is not fully compatible with WSDM. Furthermore, the notation does not allow selecting the activities according to the user's age.

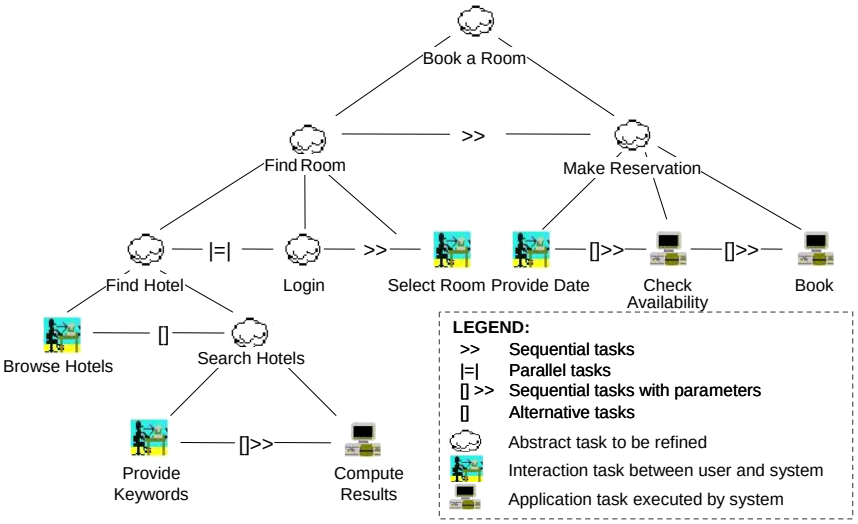


Figure 10. WSDM: The Booking Task CTT

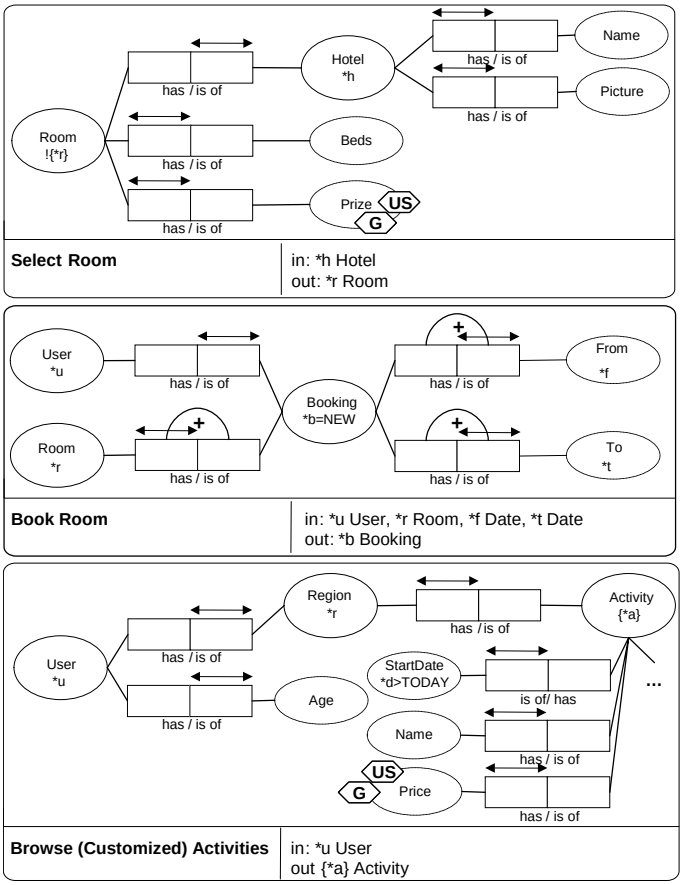


Figure 11. WSDM: Object Chunks at Content Level

Customization Scenario Administrator Links. In Figure 12, the structure of the hypertext level is depicted in terms of the specific tracks for the audience classes. The *User Track* is presented in detail: nodes are connected with the necessary object chunks and links have attached parameters to be transported. Note that, only the object chunks previously presented in Figure 11 are depicted. With respect to the customization scenario *Administrator Links*, the WSDM notation allows specifying conditional links (cf. link to *Edit Activity* node), which is only available if the associated condition formulated in natural language turns out to be true (cf. the bottom-right corner in Figure 12).

Customization Scenario Special Offers. The *Special Offers* scenario can be realized on the basis of ASL. The users shall be provided with a link to the *Region Details* node, if their browsing behavior indicates a certain

interest in the region. The ASL rule of Listing 4 specifies to provide a link from the *Activity Details* (n12) to the *Region Details* node (n11) if in more than 90% of the cases, the *Region Details* node is accessed by starting navigation through the hypertext from the *Activity Details* node and if the *Region Details* node is visited in 5% of all node accesses of the web site.

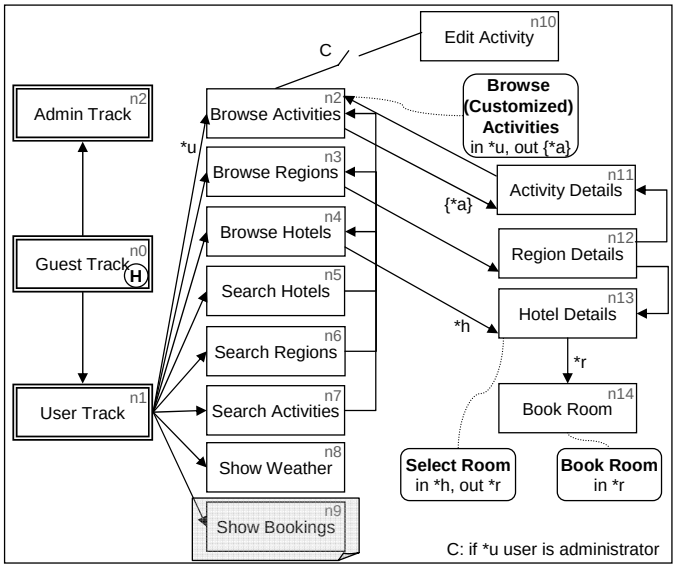


Figure 12. WSDM: The Navigation Model and the User Navigation Track

```
if      numberOfVisits (n12, ((n11, n2), ( n2 , n3), (n3, n12))) /
                                    numberOfVisits (n12)> 0,9 and
        numberOfVisits (n12, ((n11, n2), (n2, n3), (n3, n12))) /
                                    totalNumberOfVisits > 0.05
then
        addLink ((n11, n12))
```

Listing 4. WSDM: Special Offers Scenario

3.4 The Object-Oriented Hypermedia Design Model (OOHDM), Rossi et al.

3.4.1 Maturity

The Object-Oriented Hypermedia Design Method (OOHDM) [Rossi and Schwabe 2006] is amongst the earliest approaches to web application modeling and – dating back to 1994 [Schwabe and Rossi 1994] – the oldest approach included in this survey (G.T). The approach has been demonstrated by different modeling examples, such as an online catalogue for architecture, an online magazine, a CD store, a university department site, and a conference management system (G.ME). Apparently, two real-world applications of OOHDM exist: first, the Portinari project[11], a web site presenting information about the painter Candido Portinari[12] [Schwabe et al. 1996], and second, an enterprise knowledge portal in the Brazilian subsidiary of the Xerox Corporation [Schwabe and Salim 2002] (G.A).

3.4.2 Web Modeling

The approach specifies a development process comprising five steps [Rossi and Schwabe 2006], namely requirements gathering on the basis of user interaction diagrams (UID), a refinement of the well known use cases technique [Vilain et al. 2000], conceptual design, navigational design, abstract interface design, and implementation where the OOHDM-Java2 software architecture is proposed [Jacyntho et al. 2002] (W.Pr), (W.Ph). OOHDM supports all web application levels. The content level, in terms of the conceptual model, is represented as a UML-like class diagram (cf. Figure 13). It has to be noted, however, that due to the use of multi-typed attributes, the OOHDM approach is not fully conform to the UML [Schwabe and Rossi 1998] (W.L). At hypertext level, the navigational class schema (cf. Figure 16) is derived from the conceptual model

[11] www.portinari.org.br

[12] Candido Portinari (December 29, 1903 - February 6, 1962) was amongst one of the most important painters of Brazil and also a prominent and influential practitioner of the neo-realism style in painting.

and at the same time integrated from the navigational contexts[13]. The navigational contexts indicate possible navigation sequences to help users complete a task (cf. Figure 14). At the end, the navigational class schema represents a view on the content level in the form of a proprietary text-based query language - similar to view-definition approaches in object-oriented databases. It is used to specify the mapping between content level and hypertext level concepts [Schwabe and Rossi 1998] (W.I). In the navigational class schema presented in Figure 16, the *Hotel* node encompasses the original *Hotel* class as well as the *Feature* class from the conceptual model. In the node *Person*, the attributes of the *Customer* and *Admin* classes have been integrated. An example of the textual specification for the *Hotel* node is given on at bottom of Figure 16. The navigational class schema again is represented as a UML class diagram, while the navigational context schema uses a proprietary notation (W.L). For the presentation level, OOHDM originally proposed the concept of Abstract Data Views (ADV) [Cowan and de Lucena 1995] which represent abstract interfaces for navigational objects such as nodes, links, and access structures [Rossi et al. 1995] (W.L). More specifically, the approach made use of so-called Configuration Diagrams for specifying interface objects, their structural relationships, and the relationships among interface objects (ADVs) and navigational objects, e.g. nodes [Rossi et al. 1995] (W.I). Furthermore, the approach made also use of Abstract Data View-Charts (a statechart derivative) for expressing the behavior of an ADV, thus being the only way of behavioral modeling in OOHDM (W.F). Recently, however, the ADV approach seems to have been abandoned in favor of the Abstract Widget Ontology (AWO) [Rossi and Schwabe 2006] (W.L). The entire interface is specified by several ontologies using RDF and OWL as a formalism. The AWO proposes a set of concepts whose instances will comprise a given interface: *SimpleActivators* react to external events such as mouse clicks, *ElementExhibitors* exhibit a type of content such as text or images, etc. For lack of an explanation, the interface between hypertext and presentation level is assumed to be specified in natural language, only (W.I).

3.4.3 Customization Modeling

Customization in OOHDM is considered during the design phase, only [Rossi et al. 2001], [Schwabe et al. 2002]. OOHDM supposes that different users might have different tasks, access rights, and information needs. Thus, on the one hand, it is suggested to reuse the conceptual model by building different navigational views for different user profiles (C.G). Likewise, it is suggested to build different interfaces for different devices, though this possibility has never been exemplified (C.P). On the other hand, the approach proposes more fine-grained adaptation of content and hypertext levels according to the current user (C.G). In this respect, a ”user” variable is introduced to be employed in the text-based query language for defining nodes and links from the navigational class schema, i.e., constraining them, as well as for defining context classes in the navigational context schema. This user variable is then ”mapped” to an appropriate class in the conceptual model, e.g., separate context model (C.SC). However, OOHDM neither suggests ways for supporting further possibly complex context factors nor context chronology (C.CE), (C.CC), (C.C).

With respect to adaptation, the approach does not define adaptation operations (C.O), (C.AE), (C.CA); as already stated before, adaptations are expressed using OOHDM’s query language (C.SA). This language can be used to specify nodes and links (cf. Figure 16) as well as context classes (cf. Figure 14(c)) and thus, supports content adaptation as well as hypertext adaptation (C.L). Since the query language is used for defining the interface between content and hypertext, OOHDM also supports adaptation of the interface (C.I). Though support of presentation level adaptation is claimed [Rossi et al. 2001], [Schwabe et al. 2002], a corresponding demonstration in terms of a modeling example seems to be missing (C.L).

3.4.4 Model-Driven Engineering

OOHDM is described as a set of models, which are built using object-oriented primitives with a syntax close to UML [Rossi et al. 2001]. A language specification of OOHDM in terms of a metamodel, a grammar, or a semantic specification, however, is missing. Nevertheless, as already explained above, the interface model has recently been replaced by the Abstract Widget Ontology making use of RDF and OWL (M.L). There is no way of specifying model transformations in the OOHDM approach (M.T), neither are platform description models available (M.P). Still, OOHDM-Web [Schwabe et al. 1999] is a web-based environment for OOHDM, which allows the specification of an OOHDM design in XML. Given this specification, and a template page, it generates web pages for the CGILua environment [Hester et al. 1998], thus implementing the specified application.

[13] Please note that OOHDM’s notion of context is different to the notion of context in the realm of customization used in this thesis.

3.4.5 Modeling Example

Following, the OOHDM solutions to the customization scenarios Customized Activities, Administrator Links, and Special Offers are presented. Although, the approach suggests modeling different hypertext models for enabling device-independence, it is not clear, how the corresponding hypertext can be associated with the current user. Furthermore, the approach offers no mechanism for customizing the presentation level of web applications. Consequently, the customization scenarios Multi-Delivery and Season's Style can not be realized in OOHDM.

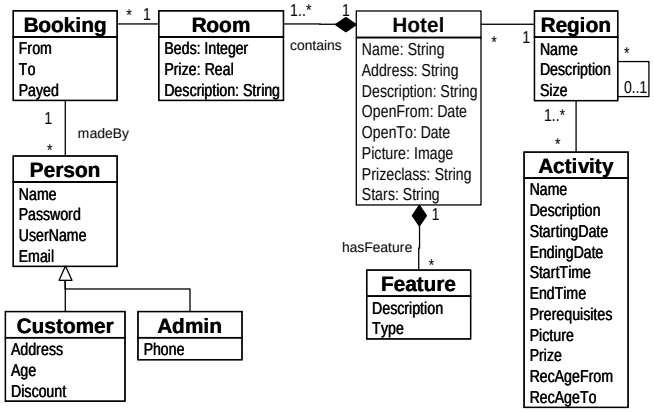
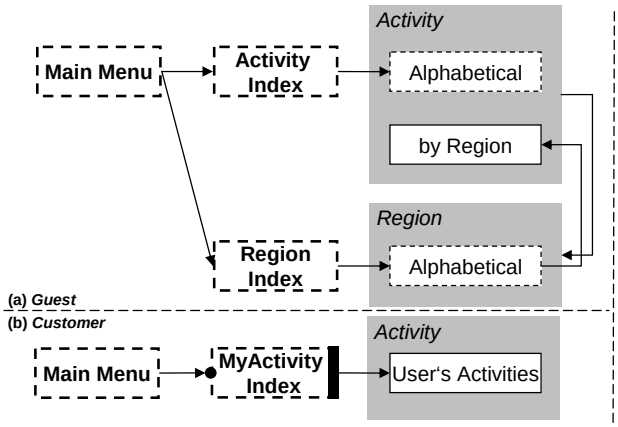


Figure 13. OOHDM: The Conceptual Model

The conceptual model of OOHDM is described with a UML class diagram (cf. Figure 13). Although possible in OOHDM, in this example we refrain from using multi-typed attributes.

Customization Scenario Customized Activities. Figure 14 presents some of the navigation context schemata for the different tasks of different user types. For example, a guest user will be able to navigate from a *Main Menu* to both an *Activity Index* and a *Region Index* (cf. Figure 14(a)). From these indices, s/he will be able to access the *Activity* and *Region* node (depicted as grey rectangles) in different contexts, e.g., alphabetically or sorted according to some criteria. Navigating from the *Region Index*, all regions will be displayed in alphabetical order. Having selected a region, the user can navigate to the *by Region* context of the *Activity* node and visit the activities of the region. With respect to the customization scenario Customized Activities, customers of the web application have an additional index, which requires the user to login (see the black bullet at the incoming arrow of *MyActivityIndex* in Figure 14(b)). The *Activity* node then can be visited in the *User's Activities* context, which will display only activities that are appropriate for the current user's age. This filter condition as well as the access restriction is defined within the Elements section of the context specification card User's *Activities* in Figure 14(c), where it is also tested if the user is of type *Customer*. Unfortunately, an explanation of OOHDM's user variable and a possible user model is missing. This example thus has been directly derived from examples given in [Rossi et al. 2001], [Schwabe et al. 2002].



(c)

Context	User's Activities		
Parameters	user		
Elements	Activity: a WHERE a.RecAgeFrom >= user.Age AND a.RecAgeTo <= user.Age AND user.login = customer.login		
InContext Class	ActivityUser'sActivities		
Navigation	sequential, order by a.Name		
Access Restriction	Customer	**Permission**	read

Figure 14. OOHDM: Administrator Links Scenario

Customization Scenario Administrator Links. OOHDM's solution to the Administrator Links scenario is shown in Figure 15(a). The administrator will be able to navigate from the context showing the activities in an

alphabetic order to the context for editing. Obviously this time, similar access restrictions to this context are needed, which are specified in the context specification card *Edit Activity* in Figure 15(b). How a user is identified as administrator, however, cannot be expressed in OOHDM.

Customization Scenario Special Offers. For realizing the Special Offers scenario, the node *SpecialOffer* is introduced to the navigational class schema in Figure 16 and realizes a content adaptation, since the prize of a room is adapted according to the user's discount. Nevertheless, it is not possible to model that this discount shall be given according to the user's booking history and thus, it is not possible to fully model the customization scenario Special Offers. On the bottom of Figure 16, the full specification of the *SpecialOffer* node is presented in terms of OOHDM's textual query language. This mechanism can also be used to define several navigational class schemata for different types of users with different access rights. For example, for guest users there will be no need for information on booking or special offers. Furthermore, the query language allows for link personalization. For example, on the bottom of Figure 16, the specification of the link *yourBookings* can be found. This link will be displayed on the homepage and will include the bookings of the current user, only. To do so, the user variable is used, which is mapped to the *Customer* class.

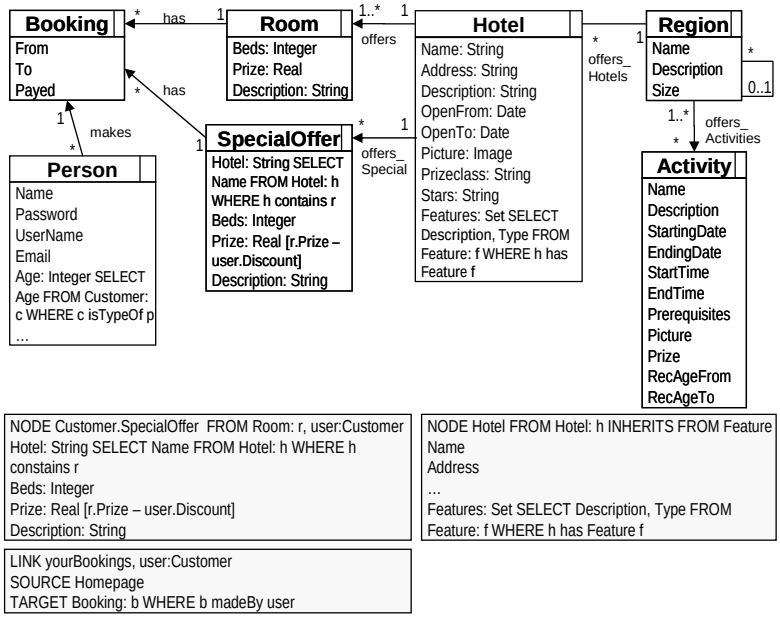


Figure 15. OOHDM: Navigational Class Schema

3.5 The UML-based Web Engineering Approach (UWE), Koch et al.

3.5.1 Maturity

The UML-based Web Engineering[14] (UWE) approach [Koch 2001], [Koch and Kraus 2002] has been continuously developed since 1998. Recent work heads towards model-driven development of web applications [Koch et al. 2006], [Koch 2007] based on model transformation techniques and MDA standards [OMG 2003] (G.T). The approach has been illustrated using several modeling examples, amongst them a music portal, an online library, a conference management system, and an e-store (G.ME). Nevertheless, its application in real-world projects has not yet been reported (G.A).

3.5.2 Web Modeling

The approach supports UML models for content (conceptual model), hypertext (navigation space model and navigation structure model), and presentation levels (presentation model) (W.L). Structural modeling is based on class diagrams at all levels, i.e., special stereotypes for the specific web concepts are introduced. While at hypertext level, state chart diagrams are used to model navigation scenarios, at presentation level, sequence diagrams can be employed to depict presentation flows (W.F). The interface between content and hypertext levels is described in terms of OCL. This means each attribute of a «navigation class», i.e., a hypertext node, in the navigation space model is derived from information stored in the conceptual model (W.I). The navigation

[14] http://www.pst.ifi.lmu.de/projekte/uwe/

structure model is an ”extension” of the navigation space model and describes how navigation is supported by access elements such as menus, indices, guided tour, etc. For each navigational class of the navigation space model, at the presentation level a separate model is created, describing where and how the navigation primitives will be presented to the user. The UML composition notation for classes is used together with a set of stereotypes for the nested elements [Koch and Kraus 2002], e.g., «text», «form», «button», and «image» [Baumeister et al. 1999]. Furthermore, UWE provides natural language guidelines to infer the navigation space model from the conceptual model and to infer the presentation model from the navigation space model. These guidelines also form the basis for model transformation within the tool ArgoUWE. UWE has a well-defined development process based on RUP [Kruchten 2000] (W.Pr) and provides explicit support for RUP’s five phases including for each phase the workflows requirements engineering (use-case diagram), analysis (conceptual model, navigation space model), and design (presentation model and refinement of all models) (W.Ph) [Koch 2001].

3.5.3 Customization Modeling

Customization modeling support within UWE is mainly considered during analysis and design phases (C.CP) and has its origins in the Munich Reference Model [Koch and Wirsing 2002]. At content level, Koch et al. distinguish between a domain model and a user model capturing contextual information about the user. Nevertheless, relationships between classes from both models allow for partial separation of context, only (C.SC). Besides the user model, an adaptation model is suggested, which is realized by means of a UML communication diagram and describes a set of condition-action rules (i.e., construction rules, adaptation rules, and acquisition rules), the rules’ temporal relationships as well as when they are triggered. Still, the rules themselves are described in natural language, only. More recently, the UWE approach suggests concepts for modeling context which are separated into a *User* package and an *Environment* package in the UWE metamodel presented in [Koch and Kraus 2003] (C.P). The details of those packages, however, have not been illuminated yet and corresponding modeling examples do not seem to be available either. No indication of either UWE’s extensibility with respect to further context properties (C.CE), nor the support of complex context (C.C) and context chronology (C.CC) is given. Furthermore, another proposal for customization modeling at navigation level has been made (C.L), (C.I), which supports link annotation, link hiding, and link generation (C.O) [Baumeister et al. 2005]. The proposal extends the UWE metamodel with concepts from aspect-oriented modeling and adaptations are provided in terms of aspects. The *LinkAspect*, *LinkAnnotationAspect*, *LinkTraversalAspect* and *LinkTransformationAspect* are modeled as sub-classes of UML packages. Thus, further adaptations could only be supported by extending the UWE language itself, i.e. through sub-classing within the UWE metamodel. (C.AE). The adaptations can be separated with aspects from the rest of the application (C.SA). The granularity of adaptations –depending on the aspects’ pointcuts – ranges from micro to macro (C.G). A mechanism for specifying complex adaptations is not available, however (C.CA).

3.5.4 Model-driven Engineering

The UWE language is defined as an extension of the UML metamodel [Koch and Kraus 2003] and additionally provides a UML profile for interoperability purposes (M.L). The guidelines for generating preliminary models, e.g., the navigation space model, from models created during earlier development phases have been automated within the tool support. Furthermore, transformation of requirements specified in terms of use case diagrams and activity diagrams into UWE models at content and hypertext level based on MOF [OMG 2002] and QVT [OMG 2005] has been recently proposed in [Koch et al. 2006]. This approach, however, has not yet been automated (M.T). UWEXML represents the proprietary code generation framework prototype and describes PIM2Code transformations based on XSLT to J2EE and Cocoon platforms [Kraus and Koch 2002]. Separate platform description models are not existent (M.P).

3.5.5 Tool Support

ArgoUWE is the free tool support for UWE (T.C). It is a proprietary extension of the UML modeling tool ArgoUML[15] (T.B), (T.O). ArgoUWE 0.16 provides general modeling support (T.V) but customization modeling has not yet been realized within the tool (T.M). During modeling, the tool checks the artifacts for errors as well as offers help to resolve the identified problems with a wizard (T.CC). Moreover, the tool allows for semi-automatically generating the navigation space model from the conceptual model, as well as the presentation model from the navigation space model (T.T). With respect to the presentation model, ArgoUWE does not yet support the composition notation of classes but uses composition associations to connect the attributes of the navigation class to the owning presentation class. UWE’s development process, however, is not realized within

[15] http://argouml.tigris.org/

the tool, since typical iterations within the development of a web application are not supported without loosing information, e.g., changing the conceptual model without loosing information from the hypertext level is not possible (T.P). Code generation (T.CG) is currently not supported, neither is there versioning support allowing for shared editing (T.Co).

3.5.6 Modeling Example

In the following, UWE's customization approach as described in [Baumeister et al. 2005] is presented. Currently, the approach is restricted to hypertext level adaptations. Thus, it is only possible to model the customization scenario Administrator Links.

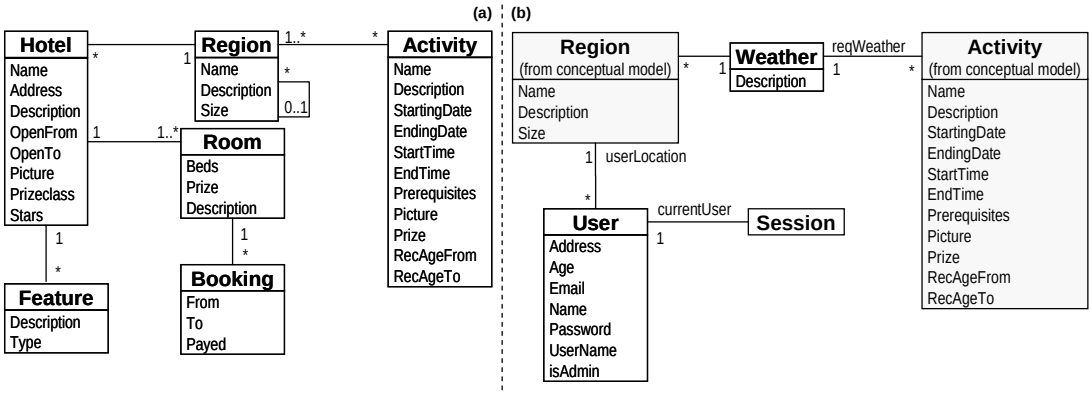


Figure 16. UWE: (a) Conceptual Model, (b) User Model

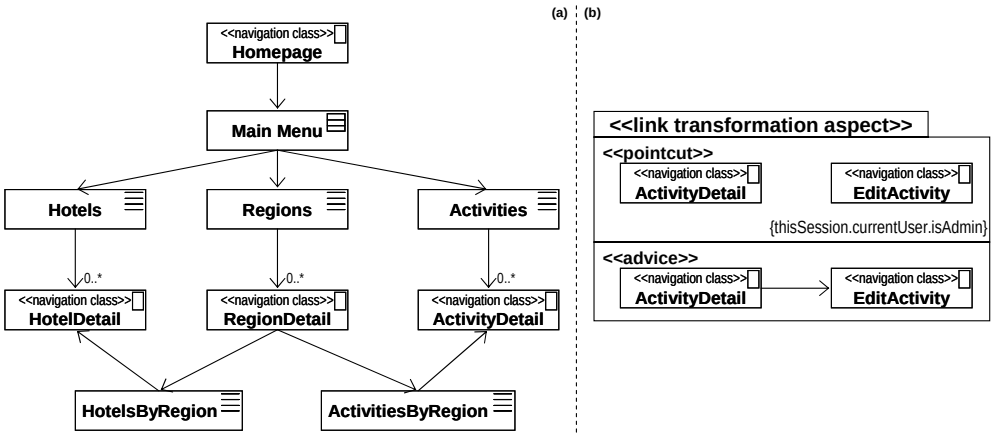


Figure 17. UWE: (a) Navigation Structure Model, (b) Administrator Links Scenario

Customization Scenario Administrator Links. The content level is depicted in Figure 17 in terms of the conceptual model and the user model. In the user model, context information about the user is captured. Users belonging to the administrator group are indicated with the *isAdmin* attribute set to true. As can be seen in Figure 17(b), the user model is connected to the conceptual model via associations to its concepts, i.e., *Region* and *Activity*. It is assumed that the application will be able to detect the user's location and resolve it to a certain region, i.e., the user model will have the necessary data. Furthermore, the user model captures the current weather of a region and activities can be selected according to their required weather situation. Part of the navigation structure diagram is depicted in Figure 18(a), which shows that indices of hotels, regions, and activities can be reached from the main menu of the TIWA. From the indices, the hotels, regions and activities can be viewed in detail. In particular, the *RegionDetails* navigation class offers to browse the region's hotels and activities via two further dedicated indices *HotelsByRegion* and *ActivitiesByRegion*, respectively. Customization functionality can then be introduced to the hypertext level using aspects. Each aspect has exactly one pointcut and one advice. According to customization scenario Administrator Links, edit links shall be provided from the *ActivityDetails* navigation class to the *EditActivity* navigation class if the user is an administrator.

The «link transformation aspect» of Figure 18(b) specifies that some link transformation needs to be performed. More specifically, the «pointcut» section defines where the adaptation takes place, i.e., the participating navigation classes *ActivityDetails* and *EditActivity* are identified. Within the aspect, any

information of the current session can be queried using OCL via 'thisSession', which is supposed to be able to identify the current session the TIWA is responding to. Thus, an additional OCL constraint specifies that the adaptation can only take place for a certain context. It constrains the adaptation to users that are administrators. In the «advice» section, the necessary adaptation is depicted, i.e., a link is introduced from the *ActivityDetails* navigation class to the *EditActivity* navigation class. Additionally, UWE supports link traversal aspects, which allow executing an action when a link is traversed (e.g., updating the user model), and link annotation aspects, which allow adding information to a link (e.g., how many times it is traversed).

3.6 The Object-Oriented Hypermedia Method (OO-H), Gomez et al.

3.6.1 Maturity

The Object Oriented Hypermedia (OO-H) Method [Gomez et al. 2001], [Gomez et al. 2000] first emerged in 2000, formerly as an extension of the OO-Method [Pastor et al. 1997], a design method for object-oriented systems. The most recent publications deal with personalization of web sites [Garrigos et al. 2005], [Garrigos et al. 2007] (G.T). The approach is demonstrated using multiple examples including amongst others a conference management system, a library system, a forum application, an e-store application, a hotel reservation system, and an e-mail system. (G.ME). Moreover the OO-H method and its tool support in terms of the VisualWade CASE tool[16] have been applied in the development of several real-world applications as is reported in [Gomez et al. 2005] (G.A).

3.6.2 Web Modeling

The OO-H method is based on UML. It defines its own process for the design phase, while other phases in the development lifecycle are described with guidelines, only (W.Pr) [Gomez et al. 2001]. The approach supports all phases from requirements engineering with use-case diagrams to implementation with the methods tool support [Cachero et al. 2001] (W.Ph). The approach comprises different models for the content, hypertext and presentation level (W.L). Standard UML class diagrams are used for content modeling. With the introduction of UML activity diagrams for modeling processes [Koch et al. 2004], the OO-H method also supports behavioral modeling at the content level (W.F). At hypertext level, so called navigation access diagrams (NAD) are associated with each user type, capturing the navigation paths and the services (from the content level) the users can activate (cf. Figure 20). They consist of navigation classes which represent views on the conceptual classes from the content level. The interface is specified graphically, i.e., the "label" of a navigation class consists of its name separated with a colon from the name of the underlying conceptual class (cf. Figure 20) (W.I). Navigation classes are associated with each other through different types of navigation links, which may have both, a navigation pattern from the OO-H pattern catalogue [Gomez et al. 2001] (e.g., *Index*, *GuidedTour*) and a set of OCL-like navigation filters, associated. Different types of collections represent (possibly hierarchical) access structures defined on navigation classes or navigation targets and may also have both, a navigation pattern and a set of navigation filters associated. Additionally, these concepts can be grouped within navigation targets in order to cover a certain user navigation requirement. Mapping rules allow inferring default navigation access diagrams from activity diagrams, which are used for modeling processes [Koch et al. 2004]. At presentation level, starting from a NAD, a default abstract presentation diagram (APD) [Cachero et al. 2000] can be generated, by using so called NAD2APD mapping rules [Gomez et al. 2001] (W.I). An APD can be interpreted as the sitemap of the web application consisting of a set of "abstract pages" associated with links and its modeling concepts are defined in several DTDs. The default APD can be refined through patterns, defined as transformation rules [Gomez et al. 2001], which are implemented in Python. Furthermore, the OO-H CASE tool also includes the composite layout diagram (CDL), which allows further refinement of the user interface (i.e., the XML specification of the APD) in the manner of a WYSIWYG editor [Cachero et al. 2001].

3.6.3 Customization Modeling

The OO-H method suggests a personalization framework in the form of a UML class diagram [Garrigos et al. 2003b], [Garrigos et al. 2005] comprising a user model and a personalization model. The user model actually extends the content level (C.SC) and allows capturing context information with respect to user, location, device, time, and network (C.P). Further context properties can be introduced through inheritance from a generic *Context* class [Garrigos et al. 2005] (C.CE). The framework currently does not foresee complex contexts or a concrete mechanism for modeling a change in context over time (C.CC), (C.C). The personalization model consists of a set of ECA rules, which realize different personalization strategies. Firstly, acquisition rules define

[16] http://www.visualwade.com

how context information is acquired. Secondly, personalization rules, which belong to a certain profile, define the adaptation that has to be made. A profile encompasses a set of personalization rules supporting a group of users with similar needs, e.g., users accessing the web application through small screen devices. And third, profile rules associate a user to a profile. Personalization rules are further distinguished into rules manipulating the content, the hypertext, and the presentation level (C.L), (C.I). The existing set of concrete rules (C.O) includes fine-grained adaptations such as *Show* [Garrigos et al. 2005] (used to show attributes in the Customized Activities scenario in Listing 6 and to show a link in the Special Offers scenario in Listing 7) but also coarse-grained adaptations (cf. *SetCSSTemplate* [Garrigos et al. 2003b] used in the Season's Style scenario in Listing 13) (C.G). The set of rules can be extended again through inheritance from the generic rule classes (i.e., *ContentRule*, *NavigationRule*, and *PresentationRule*) [Garrigos et al. 2003b] (C.AE). Currently, the combination of adaptations to form complex adaptations is not considered (C.CA), however. The ECA rules are specified separate from the other models (C.SA) following the syntax of the EBNF-based Personalization Rule Modeling Language (PRML), which can be interpreted by a rule engine at runtime [Garrigos et al. 2003a]. The use of this rule language, however, means that any extension of existing adaptation operations has to be done to this language (C.AE). Customization is considered in the design and the implementation phase of the development process (C.CP).

3.6.4 Model-driven Engineering

The OO-H language definition for the content and hypertext level is realized as an extension of UML and OCL [Cachero et al. 2001]. Still, the concepts of the presentation level, i.e., the APD, are defined as a set of DTDs and a separate language for the specification of customization is provided with the EBNF-based PRML (M.L). The OO-H approach provides mapping rules from activity diagrams to NADs and from NADs to APDs. As mentioned before there is tool support for the OO-H method, which implements the NAD to APD transformations (PIM2PIM) as well as modeling PIM2Code transformations (M.T). The approach, however, does not describe platform models (M.P).

3.6.5 Tool Support

The current version of the VisualWade tool is 1.2 (T.V). VisualWade is a commercial tool, but offers a trial version (T.B), (T.O), (T.C). It allows modeling in general but does not support customization modeling. Apart from VisualWade, the generation of UWAs on the basis of the PRML language is recently provided by a prototype described in [Garrigos et al. 2007]. In this work, the OO-H approach has evolved to the distinct Adaptive OO-H approach, (T.M). The OO-H method's process is supported only partly within VisualWade, e.g., no use-cases for the requirements definition phase are available. Apart from that, the conceptual model, the NAD as well as the CLD are supported, as mentioned before. The APD actually is not used explicitly but indirectly via the CDL. The user will start from a conceptual model, via the NAD to the CLD, but can go back to previous models for changes. Changing the NAD, however, means loosing the presentation model which has to be generated anew (T.P). VisualWade supports PIM2PIM and PIM2Code transformations (T.MG) as well as provides an OCL-based action language for transformations and code generation capabilities [Gomez et al. 2005]. With respect to code generation, VisualWade currently supports one model compiler that generates code for the PHP target platform. The application can be generated in several independent steps such that the application logic and the database, for example, can be created in random order (T.CG). The modeled artifacts can be statically checked on demand and the user is provided with hints indicating how to solve the detected problems (T.CC). With respect to collaborative development of web applications, VisualWade currently provides no specific support (T.Co).

3.6.6 Modeling Example

OO-H's PRML allows for modeling all customization scenarios. Before presenting the individual customization scenarios in terms of PRML rules, however, the content and hypertext level as well as the context model of the web application are presented. The content level of the TIWA is extended with the context model (cf. Figure 19). More specifically, the given context framework has been extended with a *WeatherContext* class in order to be able to consider the weather context factor. Furthermore, the conceptual classes of the content level have been extended with operations for creating, updating, and deleting objects of the specific types. Figure 20 presents part of the hypertext level. In this case, one NAD has been designed for guests and registered users of the TIWA. From the homepage, users will have to login to the TIWA and will be redirected to the *Homepage* menu if they belong to the user group "user" (cf. *To Homepage* link). The links supporting the login process have attached OCL-like preconditions and navigation filters, which will be ignored for the remaining links for readability purposes. A separate view for administrators is indicated with the navigational target *AdminView* but not further detailed. If the user belongs to the usergroup "admin", s/he will be redirected to the *AdminView* (cf.

To AdminView link). In case the input retrieves no dataset, i.e., no user account can be found, the user is redirected to an error page (cf. *Error* link). From the *Homepage* menu the user can navigate, for example, to the Hotels navigation class which will present an index of hotels, as is indicated by the second part of the navigation class label which denotes the corresponding conceptual class from the content level. Likewise, the user can browse regions and activities and visit the navigational classes presenting an object of interest, e.g., the *Hotel* navigation class provides detailed information on one hotel. For those links that are not accessible for guests, e.g., *Create Booking*, *Your Bookings*, and *Edit Activity*, special rules need to be designed in PRML such as has been done for the *Edit Activity* link in the customization scenario Administrator Links. The *Edit Activity* link and the *Create Booking* link are service links that call the *Edit()* and *New()* operations of the *Activity* and *Booking* conceptual classes when the user books a room or the administrator edits an activity, respectively.

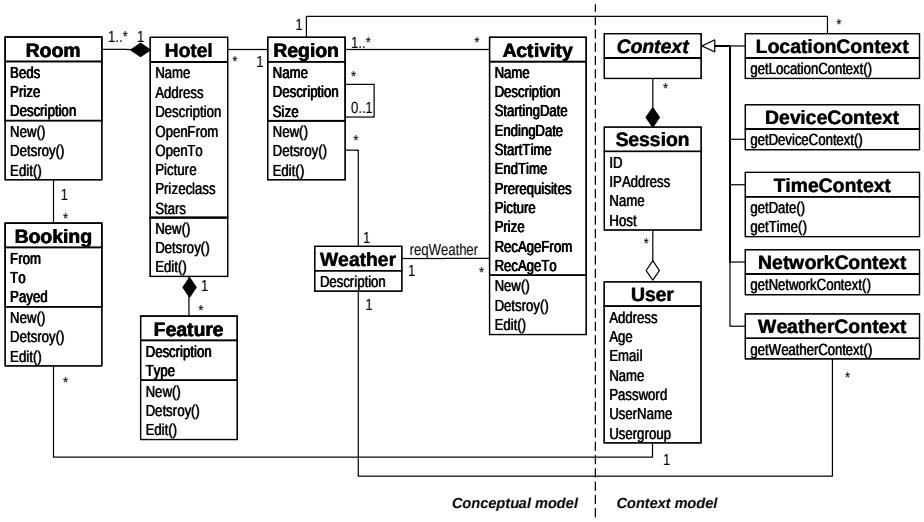


Figure 18. OO-H: Conceptual Model and Context model

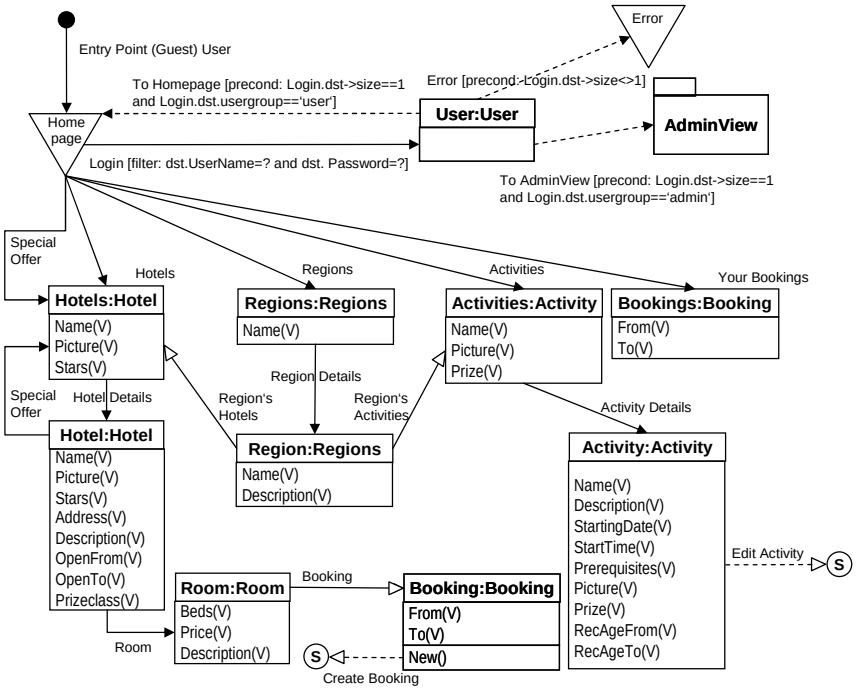


Figure 19. OO-H: Navigation Access Diagram

```
#INIT SECTION
When init do
  AttachRuleToGroup ("acquireContext", "OOH:all")
  AttachRuleToGroup ("defSmallScreen", "OOH:all")
  AttachRuleToGroup ("defSeason", "OOH:all")
  AttachRuleToGroup ("defUserGroup", "OOH:all")
  AttachRuleToGroup ("customizedActivities", "user")
  AttachRuleToGroup ("specialOffers", " OOH:all ")
  AttachRuleToGroup („editLink", „admin")
  AttachRuleToGroup ("ignoreImages", "smallScreen")
  AttachRuleToGroup ("summerStyle", "summer")
  AttachRuleToGroup ("winterStyle", "winter")
endWhen

#AQUISITION SECTION
#RULE: "aquireContext" priority="high"

#PROFILE SECTION
#RULE: "defSmallScreen" priority="medium" //CustomizationScenario (4)
#RULE: "defSeason" priority="medium" //CustomizationScenario (5)
#RULE: "defUserGroup" priority="medium" //CustomizationScenario (3)

#PERSONALIZATION SECTION
#RULE: "customizedActivities" priority="medium" //CustomizationScenario (1)
#RULE: "specialOffers" priority="medium" //CustomizationScenario (2)
#RULE: „editLink" priority="medium" //CustomizationScenario (3)
#RULE: "ignoreImages" priority="medium" //CustomizationScenario (4)
#RULE: "summerStyle" priority="medium" //CustomizationScenario (5)
#RULE: "winterStyle" priority="medium“ //CustomizationScenario (5)
```

Figure 20. OO-H: Personalization Rules

As already stated above, all customization scenarios can be realized with PRML. Figure 21 outlines the realization of the customization scenarios on basis of ten PRML rules: one acquisition rule, three profile rules, and six personalization rules. All of them need to be initialized (cf. init section) meaning that all rules will be associated to a profile group and thus, to the users of that group. The default group ’OOH:all’ applies to all users. In the acquisition section of Figure 21, when a new session starts, the acquire-Context rule will collect the necessary context information to be available in other rules, i.e., the device used to access the TIWA as well as the current date:

```
#RULE: ”aquireContext ” priority=”high”
When SessionStart do
    deviceContext=UM.DeviceContext.getDeviceContext()
    date=UM.TimeContext.getDate()
endWhen
```

Listing 5. OO-H: Context Aquisition Rule

Also at session start, the user will be attached to one or more profile groups (cf. profile section). In the personalization section, the actual rules for adapting content, hypertext, and presentation of the TIWA are specified. Following, the profile rules as well as the personalization rules will be explained for the individual customization scenarios.

Customization Scenario Customized Activities. The Rule *customizedActivities* (cf. Listing 6) is applied if the user navigates from the *Homepage* to the *Activities* node. Depending on the user’s age, location, the current date and time, as well as the current weather, the user will be presented a customized selection of activities. In this respect, the adaptation functionality is encapsulated in Show.

```
#RULE: ”customizedActivities” priority=”medium”
When Navigation (Activities (Activity)) do
    If (UM.User.Age >= NM.Activity.RecAgeFrom and
        UM.User.Age <= NM.Activity.RecAgeTo and
        UM.LocationContext.getLocationContext = NM.Activity.region and
        UM.WeatherContext.getWeatherContext = NM.Activity.reqWeather
and
        date < NM.Activity.StartingDate and
        UM.TimeContext.getTime() < NM.Activity.StartTime)
    then
        Show(Activity.Name, Activity.Description, Activity.Prize)
    endIf
endWhen
```

Listing 6. OO-H: Customized Activities Scenario

Customization Scenario Special Offers. If the user views three hotels of the same region for at least 60 second (cf. *LoadSet* event [Garrigos and Gomez 2006]), the user will be presented the link *Special Offers* to the *Hotels* node. In this node, special offers for the specific region will be presented (cf. Listing 7). This is specified with the rule *specialOffers*:

```
#RULE: ”specialOffers” priority=”medium”
When LoadSet [ Hotel (NM. Region reg , 3, 60)] do
    Show(SpecialOffers)
endWhen
```

Listing 7. OO-H: Special Offers Scenario

Customization Scenario Administrator Links. The profile rule *defUser*-Group (cf. Listing 8) identifies the user type of the current user, i.e., if the user has logged in s/he will be associated to either the user or the admin profile group defined in the init section in Figure 21 with *AttachUserToPGroup* [Garrigos et al. 2005]. The OO-H approach assumes that the current user group can be queried from the user model [Garrigos and Gomez 2006].

```
#RULE: ”defUserGroup” priority=”medium”
When Session Start do
If (UM.User.Usergroup=”admin” then AttachUserToPGroup (”admin”)
endIf
If (UM.User.Usergroup=”user” then AttachUserToPGroup (”user”)
endIf
endWhen
```

Listing 8. OO-H: Administrator Links Scenario (1)

The *editLink* personalization rule specified in Listing 9 is associated to the admin profile group (cf. init section Figure 21) and ensures that the service link *Edit Activity* will be available to administrators but not to registered users or guests of the TIWA:

```
#RULE: ”editLink” priority=”medium”
When LoadElement(Activity) do
Show(EditActivity)
endWhen
```

Listing 9. OO-H: Administrator Links Scenario (2)

Customization Scenario Multi-Delivery. The *defSmallScreen* profile rule in Listing 10 associates the user with the *smallScreen* group defined in the init section in Figure 21, if the device used to access the TIWA is either a PDA or a mobile phone (cf. *deviceContext* defined in the *contextAquisition* rule of Listing 5).

```
#RULE: “defSmallScreen” priority=”medium”
When SessionStart do
If (deviceContext=”PDA” or deviceContext=”Mobile”) then
AttachUserToPGroup(”smallScreen”)
endIf
endWhen
```

Listing 10. OO-H: Multi-Delivery Scenario (1)

Multi-delivery can be realized also at a fine-grained level. If a PDA or a mobile phone is used to access the TIWA, pictures shall be omitted. The rule *ignoreImages* (cf. Listing 11) applies to the *Hotel* and *Activity* nodes. As is defined by the above profile rule, it is attached to *smallScreen* group.

```
#RULE: ”ignoreImages” priority=”medium”
When LoadSet [Hotel(NM.Hotel h) | | Activity(NM.Activity a)]
do
    h.picture.Visible=false
    a.picture.Visible=false
endWhen
```

Listing 11. OO-H: Multi-Delivery Scenario (2)

Customization Scenario Season's Style. Finally, the *summerStyle* and *winterStyle* personalization rules change the CSS used to the ones specified. The rules are attached to the summer and winter profile groups, respectively (cf. *init section* Figure 21) and will cause the presentation to be adapted depending on the current season. According to the *defSeason* profile rule, the user will either be attached to the summer or the winter group depending on the current date:

```
#RULE: ”defSeason” priority=”medium”
When Session Start do
If (date >= 21.03. and date < 21.09) then
    AttachUserToPGroup(”summer”)
endIf
If (date >= 21.09 . and date < 21.03) then
    AttachUserToPGroup(”winter”)
endIf
endWhen
```

Listing 12. OO-H: Season's Style Scenario (1)

Depending on the group the user is associated with, one of the *summerStyle* and *winterStyle* personalization rules will be triggered and define an appropriate CSS style sheet to be used.

```
#RULE: "summerStyle" priority="medium"
When SessionStart do
    SetCSSTemplate ("summer.css")
endWhen

#RULE: "winterStyle" priority="medium"
When SessionStart do
    SetCSSTemplate ("winter.css")
endWhen
```

Listing 13. OO-H: Season's Style Scenario (2)

3.7 The Object-Oriented Web Solution Approach (OOWS), Pastor et al.

3.7.1 Maturity

Object Oriented Web Solutions (OOWS) [Pastor et al. 2006] is an approach that was first proposed in 2000 [Pastor et al. 2000] by extending OO-Method [Pastor et al. 1997] to support web modeling. Unlike the OO-H approach, OOWS is still based on the OO-Method. Recently, first results of the implementation of tool support for the OOWS approach have been published (G.T). The approach is presented with modeling examples like a university department site, a ticket ordering system, a book store, a travel agency system, and a university research group management system (G.ME). The university department site has been realized and is available online[17] (G.A).

3.7.2 Web Modeling

The OOWS method supports all software development phases including a so called solution development phase in which models are translated into code (W.Ph). Developers are provided with appropriate modeling means in each of OOWS's proprietary four-step development process (W.Pr). For the requirements engineering phase, use-case diagrams, scenarios, and task descriptions are used [Valderas et al. 2005], [Torres et al. 2005]. Task descriptions comprise task taxonomies, whereby each elementary task is described with an UML activity diagram, as well as information templates which are based on Class-Responsibility-Collaboration (CRC) cards. Thereafter a conceptual schema is built in the analysis phase. Since the approach is based on the OO-Method, only the models of the hypertext and presentation level are special to OOWS. In the design phase, different kinds of models are used (cf. [Fons et al. 2003]) (W.Ph): On the content level, the information is captured using the structural model, i.e., a UML-like class diagram, just like in OO-Method. In addition, UML state and sequence diagrams can be used to describe behavioral aspects and represent the dynamic model in OOWS (W.L). A functional model describes service effects in a textual formal specification language which is based on the OASIS formal language[18] as is described in [Pastor et al. 2006]. A navigational model comprises navigational maps which are used to define the global and structural aspects of the navigation, i.e., how so-called navigation contexts are related via navigational links with each other. Apart from that, the navigation contexts can be modeled using the navigation context diagram which allows designing the white-box view of a navigation context (cf. Figure 24(b)). The navigational contexts comprise navigational classes which are views on the content level. The interface between content and hypertext level is graphically specified. The hypertext level and the presentation level are intermingled though: A presentation model only enriches the "in-the-small" model of the navigational context with presentational patterns concerning aspects such as scrolling or ordering of information (W.I). As already mentioned, only the content and the hypertext level can be enriched with behavioral diagrams while on all three levels structural modeling is supported (W.F).

3.7.3 Customization Modeling

The only context properties addressed in OOWS literature seems to be the user property. In [Abrahao et al. 2002], more possible context properties are mentioned but they would have to be introduced by means of the standard OOWS models (C.P), (C.CE). Apart from that, no mechanisms for chronology (C.C) or complex context (C.CC) are available and since the context properties have to be included in the content level, there is neither separation of context (C.SC). Nevertheless, OOWS allows the definition of adaptations very early in the design process, namely during requirements specification [Rojas et al. 2006], i.e., within activity diagrams, until

[17] http://www.dsic.upv.es
[18] http://www.oasis-open.org/specs/index.php

the design phase, i.e., within navigational contexts (C.CP). In particular, the different user types can be incorporated into the content model by merging the base content model with a user stereotype model. Furthermore, when specifying tasks by means of activity diagrams, the modeler can define adaptation rules such as link or content hiding as well as sorting of information according to previous user behavior (C.O). These adaptation operations are based on OCL conditions and are limited to fine-grained adaptations such as accessibility, filter, or sorting conditions (C.AE), (C.G). Complex adaptations cannot be realized (C.CA). The adaptation operations can be used to influence the hypertext level, only (C.L), (C.I). Moreover, customization in the OOWS approach requires the adaptations to be integrated into the models, allowing no separation of adaptation (C.SA).

3.7.4 Model-driven Engineering

Recently, the OOWS language has been defined as a MOF-based metamodel and tool support on the basis of the Eclipse Graphical Modeling Framework (GMF)[19] is under development [Valverde et al. 2007]. Also a definition of the language in OWL has been proposed [Torres et al. 2004] (M.L). OOWS supports an MDA-based approach including the transformation of platform-independent models. The possibility to generate code for the front-end of a web application is also currently under development within the above mentioned tool support (M.T). In order to generate the back-end of a web application, the commercial tool OlivaNova[20] is used. In the future, the integration of the code generation for the back-end and the front-end is planned. A platform description model does not seem to be used, however (M.P).

3.7.5 Modeling Example

In the following, three customization scenarios are presented, namely Customized Activities, Administrator Links, and Special Offers. Concerning the customization scenarios Multi-Delivery and Season's Style, it is not possible to cope with them in OOWS due to the fact that only the user context can be exploited for adaptation purposes.

Figure 22 shows the example's underlying class diagram, in which the different user stereotypes (i.e., Customer and Admin classes) are already integrated with the structural model of the TIWA. The User class and its sub-classes represent a hierarchy of possible user stereotypes of the TIWA. Following, for each scenario, the diagrams which are most relevant for customization in the OOWS approach are shown. These are, on the one hand, the activity diagrams from the requirements phase and, on the other hand, the resulting navigational models, in particular the navigational contexts, to present how the adaptation rules are incorporated into the hypertext.

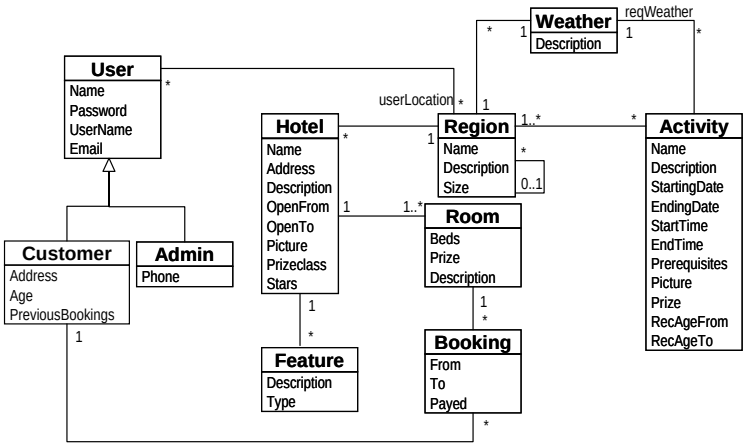


Figure 21. OOWS: The Structural Model

Customization Scenario Customized Activities. The scenario is explained starting from the requirements specification, in particular from the activity diagram shown in Figure 23(a). In the first action, the user has to select one region from an index of regions. After selection, it must be verified by the system if the user is logged in (cf. *action2 : Activity* in Figure 23a) or if s/he is an anonymous user. If the user is logged in, the following

[19] http://www.eclipse.org/gmf
[20] http://www.programmiermaschine.de/

precondition can ensure that only appropriate activities are shown by the expression: *self.recAgeFrom > #user#.age and self.recAgeTo < #user#.age*.

Using the keyword *self*, it is possible to access objects which should be displayed in this UML action (in this case, objects of the *Activity* class). With an OCL-like point-notation one can navigate through the structural model and retrieve attribute values to be compared with values from the current context. One can access attribute values of the user with the *#user#* variable representing the current context. If the user is not logged in (cf. *action 3 : Activity* in Figure 23a), all activities of the selected region are shown, i.e., no personalized content is provided to the user. Finally, the user can select one of the activities to navigate to a detailed view of the selected activity.

In Figure 23(b), the resulting navigational context can be seen. In particular, the third compartment of the *Activity* view is important for the required adaptation. The filter for presenting the user with activities according to his/her age is only applied if the user is logged in (cf. *if #user# isTypeOf(Customer)* in Figure 23b), otherwise no filtering is done.

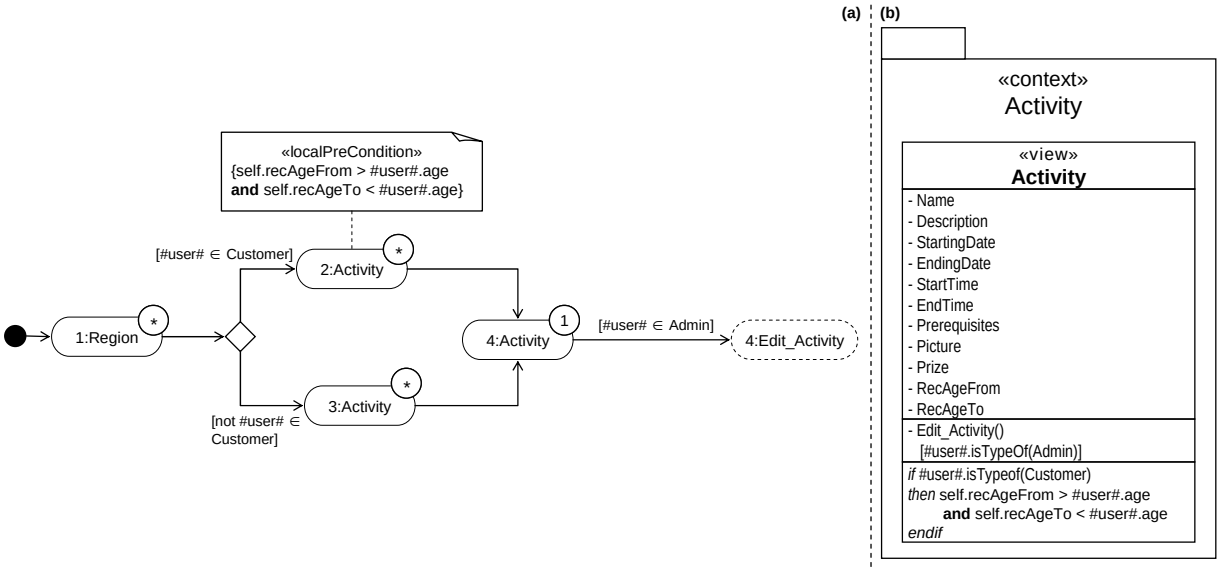


Figure 22. OOWS: (a) Activity Diagram Show Activities, (b) Navigation Context Show Activities

Customization Scenario Administrator Links. For editing activities, administrators need to log in to the web application and navigate to the list of activities for a certain region as is specified in Figure 23(a). Having selected one activity, administrators are presented with a special link allowing them to edit the activity, as opposed to other normal users. This is ensured by the following condition associated with the control flow of the activity diagram: *#user# 2 Admin*.

Furthermore, in the third compartment of the *Activity* view in Figure 23(b), the *Edit Activity()* operation is shown which corresponds to the previous condition and is only executable if the user is logged in as administrator.

Customization Scenario Special Offers. For the OOWS approach, this scenario is explained with special offers in the context of booking a hotel room. The activity diagram for this scenario is shown in Figure 24(a).

It is depicted that, in the first step, a hotel has to be selected from an index of hotels (cf. *action 1 : Hotel* in Figure 24a). The index is denoted with the '*' label associated to the action, while the '1' label indicates one instance to be displayed. Subsequently, the user can book a hotel, whereby the adaptation comes into play (cf. *action 3 : Book* in Figure 24a). If the user has made more then three bookings in the past s/he gets a discount provided by the *action 4:Get_Discount*. However, if the condition is not fulfilled, no discount is allowed. The required adaptation rule to realize this functionality is incorporated into the activity diagram by annotating the transitions with conditions. For example, with the *#user#* variable it is possible to retrieve the actual user object and its attribute values – in this case *#user#.previousBookings* retrieves the value of the *previousBookings* attribute of the class *User* or rather of *Customer*. In Figure 24(b), the corresponding navigation context is illustrated for *action 2:Hotel* of the activity diagram shown in Figure 24(a). Each navigation context must have exactly one so-called manager class which defines a view on the content level, i.e. the *Hotel* class in this case. Concerning the adaptation rule in the activity diagram, the resulting specification is an operation *GetDiscount()*

which is only executable under the condition which was specified in the activity diagram, namely the user must have made more than three bookings in the past.

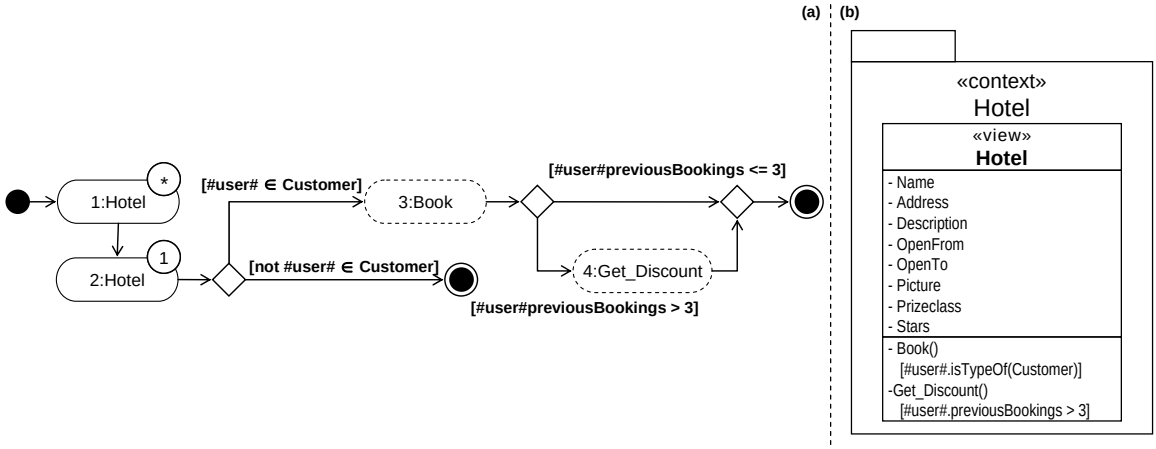


Figure 23. OOWS: (a) Activity Diagram Book Hotel, (b) Navigation Context Book Hotel

4. LESSONS LEARNED

In this section, the experiences acquired during the evaluation of the selected approaches as well as during modeling of the running example shall be summarized. In particular, it illustrates the results at a glance with comparison tables according to the five categories of criteria as well as points out future directions for web modeling approaches to comprehensively address the development of ubiquitous web applications.

4.1 Maturity

Small Set of Similar Modeling Examples. A first overview on the web modeling field showed that the individual approaches often made use of similar modeling examples. Since stemming from academia, it is not surprising that a conference management system as well as some kind of department web site have been used particularly often to demonstrate a web modeling approach. Beyond this, different kinds of e-stores, e.g., selling books and CD's, as well as art gallery web applications have been used several times. The current set of modeling examples used in the web modeling field, however, raises some important questions: Are those examples complex enough to show the approaches' applicability? Do they cover all different kinds of web applications, e.g., ubiquitous web applications, workflow-based web applications? Consequently, what the web modeling field needs is a set of generally acknowledged modeling examples in a public catalogue. Moreover, reference implementations for those examples can serve as "testbeds", i.e., to show if it is possible to develop such web applications with existing web modeling approaches. Additionally, it might be worthwhile thinking whether all the different approaches actually shall intend to cover the same types of applications or rather diversify with respect to specialize on specific web application types, or not.

Rare Application in Real-World Projects. Web modeling approaches have already a ten year old history. Nevertheless, their application in real-world projects in particular in the context of a commercial setting is still rare. According to the survey of Barry et al. [Barry and Lang 2003] and our observations, in practice the awareness of academic methods is still rather low and consequently, they are rarely used by practitioners. To gain the badly needed impact on practitioners, there is an urgent need among all web application development approaches for more reference applications which are built with academic methods in order to demonstrate their maturity. Specifically, reaching beyond classical academic web application examples, towards more intensive cooperation with industry is necessary. Furthermore, besides presenting their approaches in terms of scientific publications, web modeling approaches will need to be presented in a way that is more suitable for practitioners to reach them as target group.

Approaches are Continuously Evolving. All of the surveyed web modeling approaches have been presented in numerous publications including refereed papers, articles, books as well as manuals. Over the time, each of the approaches has been subject to extensions, e.g., for supporting business process modeling and customization modeling, as well as to major evolutions, e.g., the introduction of ontologies as a new formalism to specify models and the use of standards enabling model-driven web engineering. From a practitioner's point of view,

these developments might erratically communicate that current web modeling methods are not yet mature enough to be used in practice – maybe one reason whey they are not yet used as extensively one could desire. Additionally, some approaches provide alternative proposals for modeling customization for which it is not clear how they are related to each other and if they can be used in parallel. In this respect, a comprehensive description of the already mature parts of a web modeling approach in terms of a manual for designers might be beneficial.

		Topicality (T)	Modeling Examples (ME)	Applications (A)
data-oriented	WebML	1999-2007	7	✓
	Hera	2000-2007	5	~
hypertext-oriented	WSDM	1998-2006	3	
object-oriented	OOHDM	1994-2006	5	✓
	UWE	1998-2007	4	
	OO-H	2000-2007	6	✓
	OOWS	2000-2007	5	✓

Legend:

✓	supported
	not supported
~	partly supported
n	number of examples
YYYY	year of introduction/ most recent publication

Figure 24. Maturity

4.2 Web Modeling

UML for Content Modeling – Proprietary Solutions for Hypertext Modeling. UML is quite popular for modeling the content level as well as the hypertext level of a web application. While UWE and OO-H are based on UML 1.x, OOHDM uses a UML-like notation. At hypertext level, however, UWE and OOHDM are the only approaches that continue using (stereotyped) UML class diagrams. Indeed, at hypertext level, one can find very different languages and notations for each individual approach. Although it is admitted that web specific concepts are important for the hypertext level and also the presentation level, not basing on a single (standard) formalism might hinder the application of solutions already provided in the model-engineering field like, e.g., model exchange, model transformation, or model checking. Additionally, probably a greater momentum could be gained when standards / standard formalisms are employed.

Presentation Level Seldomly Addressed. When it comes to the supported levels of web applications it becomes obvious that the presentation level is often only a marginal concern as already been diagnosed in [Retschitzegger and Schwinger 2000] some time ago. Although nearly every method provides presentation level support, it is, however, typically omitted in modeling examples. Interestingly, WebML is the only method not providing a presentation model at all, but leaves presentation concerns to the tool. With respect to the other approaches, although desirable when ubiquitous web applications need to be designed, nodes of the hypertext level are often mapped one-to-one to pages from the presentation level, thus not exploiting the chance to provide different presentations on bases of the same hypertext model.

Behavioral Modeling Not Comprehensively Considered. Behavioral modeling is typically not supported comprehensively for all levels. There is, however, a tendency for using behavioral diagrams, including use cases, activity diagrams, concurrent task trees, in the requirements engineering and analysis phases, as well as for describing scenarios of user behavior with, e.g., sequence diagrams. Some form of behavioral modeling is also introduced by approaches providing support for business process modeling or workflow-based web applications. Ubiquitous web applications, however, encounter a greater need for addressing behavioral aspects than static web sites, thus a better support for behavioral modeling at all levels is desired.

Strong Processes. Almost all of the surveyed approaches support the developer with appropriate guidance to developing a web application from requirements engineering to implementation on the basis of their modeling techniques, i.e., the necessary steps, artifacts to be produced within each step, and actors are explained. While most of the approaches propose their own development process, WebML and UWE do base their process on existing work, i.e. Boehm's Spiral model and RUP, respectively. Interestingly enough, all approaches start modeling the web application's content level. Still, the applicability and usability of these processes need to be investigated in real-world projects raising in general demand to empirically evaluate the applicability of the approaches.

		Levels											Features (F)			Phases (Ph)				Development Process (Pr)			
		Supported Levels (L)			Interfaces (I)																		
					C⇔H				H⇔P														
		C	H	P	graphical	text-based	natural language	separated	graphical	text-based	natural language	separated	C	H	P	R	A	D	I	Origin	Steps	Artifacts	Actors
data-oriented	WebML	ER	own		✓	✓			n/a				s	s/b	s	✓	✓	✓	✓	Boehm's Spiral model	7	✓	✓
	Hera	RDFS	own	own	✓	✓			✓	✓			s	s/b	s			✓	✓	own	~6	✓	
hypertext-oriented	WSDM	ORM	own	own	✓				✓				s/b	s	s	✓	✓	✓	✓	own	5	✓	
object-oriented	OOHDM	UML-like	own, UML-like	own		✓			✓				s	s	s/b	✓	✓	✓	✓	own	5	✓	
	UWE	UML CD	UML CD & OCL, UML SM	UML CD UML SD		✓			✓				s	s/b	s/b	✓	✓	✓	✓	RUP	5	✓	✓
	OO-H	UML CD	UML CD & OCL	own	✓	✓					✓	n/a	s/b	s	s	✓	✓	✓	✓	own	1	✓	
	OOWS	CD, SD, SM	own	own	✓				n/a				s/b	s/b	s	✓	✓	✓	✓	own	4	✓	✓

Legend:

✓	supported
	not supported
n/a	not applicable
s/b	structure / behavior
n	number of phases
RUP	Rational Unified Process
ER	Entity Relationship Diagram
UML	Unified Modeling Language
CD	Class Diagram
SD	Sequence Diagram
SM	State Machine Diagram
OCL	Object Constraint Language
ORM	Object Role Modeling
RDFS	Resource Description Framework Schema

Figure 25. Web Modeling

4.3 Customization Modeling

Set of Context Properties Limited and Not-Extensible. All web modeling approaches do support customization with respect to the user context property, thus laying the path to personalization, while other context properties are often not taken into account. The investigation revealed that context properties typically are considered in isolation and that complex adaptations regarding several context properties are rare. Furthermore, the extension of the supported set of context properties is typically not discussed in current web modeling approaches. In this respect, the context model of OO-H represents the only exception considering user, location, device, time, and network context as well as allowing for their extension. Furthermore, except for the WebML and OO-H approaches, there is no support for explicit modeling concepts capturing context information. Typically, it is assumed that context information is updated by some external service. Context captured comprehensively and in combination, however, is the important prerequisite to enable ubiquitous web applications and consequently should not be limited to some selected context properties, only.

Context Chronology and Complex Context not Addressed. Currently, none of the investigated web modeling approaches considers complex contexts as well as basing adaptations on historical context information. Consequently, adaptations have to be specified as a reaction to the sum of simple contexts. Concerning context chronology, some approaches allow for adaptations according to the user's navigation behavior. This historical information about the user is often implicitly available in terms of predicates of a rule language such as in WebML and OO-H. Still, other historical context information, such as the user's location over time, cannot be stored. Unless approaches allow for a proper representation of complex context the full potential of ubiquitous web applications can only hardly be exploited properly.

Set of Adaptation Operations Limited and Not-Extensible. The set of adaptation operations that some web modeling approaches provide is limited to operations that can be performed upon typical concepts of a web application, including add/remove a link, change the style, add/remove a node, and sort some information. Currently, there seems to be no web modeling approach that supports operations on media types such as "resize image" or "shorten text". Of course, it has to be admitted that adaptation operations can be modeled in terms of normal behavior / actions. However, providing a set of generic adaptation operations applicable in a large range of application scenarios would facilitate reuse and also reduce the burden of the developer needing to model such adaptation operations.

Complex Adaptation Operations not Considered. Complex adaptations currently have been realized in the WSDM approach only. The *promoteNode* and *demoteNode* operations have been built upon primitive ones such as *add/remove link*. Nevertheless, none of the approaches provide the necessary modeling means that allow specifying complex adaptation operations on the basis of primitive ones. Likewise before, complex adaptations would allow to better support the modeler in that s/he is enabled to model in terms of concepts better suitable to the problem at hand.

Limited Support for Content, Presentation, and Interface Adaptations. Adaptations at the hypertext level are considered within all of the investigated approaches. With respect to content and presentation level only half of

the approaches provide necessary adaptation operations, while interface adaptation is supported by WSDM and OOHDM, only. More interestingly, approaches supporting presentation adaptations rather operate at a coarse-grained level, e.g., by providing adaptation operations that change the complete style of the web applications presentation. In contrast, the Hera approach allows defining alternative layout managers for parts of the presentation model, thus realizing adaptation at a more fine-grained level.

		Context									Adaptation												
		Properties (P)					Extensibility (CE)	Chronology (C)	Complex Context (CC)	Separation of Context (SC)	Operations (O)	Extensibility (AE)	Complex Adaptation (CA)	Levels (L)			Interfaces (I)		Granularity (G)	Separation of Adaptation (SA)	Phases (CP)			
		User	Location	Device	Time	Network								C	H	P	C⇔H	H⇔P			R	A	D	I
data-oriented	WebML	✓	✓	✓		✓	~	~		~	✓				✓	✓	✓		mi/ma				✓	
	Hera	✓		✓						~				✓	✓	✓			mi	✓			✓	✓
hypertext-oriented	WSDM	✓	✓		✓			~			✓		✓		✓		✓		mi/ma		✓	✓	✓	✓
object-oriented	OOHDM	✓												✓	✓		✓		mi/ma				✓	
	UWE	✓								~	✓				✓				mi/ma	✓		✓	✓	
	OO-H	✓	✓	✓	✓	✓	✓	~		~	✓	~		✓	✓	✓			mi/ma	✓			✓	✓
	OOWS	✓									✓				✓				mi		✓	✓	✓	

Legend:

✓	supported
	not supported
~	partly supported
mi	micro
ma	macro

Figure 26. Customization Modeling

Customization not Comprehensively Considered in all Development Phases. Customization modeling is predominantly considered during design, the exceptions being the OOHDM and the WSDM approaches. It seems that customization is treated as a separate step in the design phase in which an existing web application is extended with customization issues. Instead, customization needs to be considered during all phases in the software development lifecycle. Consequently, web modeling approaches might want to adapt their current development processes in order to appropriately include customization concerns in all phases.

Disregarded Crosscutting Nature of Customization. Current web modeling languages do not allow for modeling customization separately from the rest of a web application model. More specifically, this is due to the missing separation of context and adaptation. Considering separation of context, in general, all web modeling approaches acknowledge the need for defining context information in a separated model. Still, guidelines supporting a developer in constructing a context model are often not available. Some guidance can be found in the WebML approach as well as in the OO-H framework for modeling context information. Nevertheless, none of the approaches currently is able to achieve full separation of context from the rest of the web application models, i.e., typically, concepts from the context model are connected to concepts from the content model in some way or the other, e.g., via associations. As another example, it is often not decidable if a certain concept shall be modeled within the content or the context model, such as the user concept of which some attributes might contribute to the core functionality of the web application and some others contribute to customization. With respect to separation of adaptation, developers typically are required to define adaptations as annotations to existing models. In the UWE, OO-H, and Hera approaches, adaptations at the hypertext level can be captured separately from the rest of the web application models, however. Thus, customization functionality is not intermingled with the rest of the web application. Nevertheless, there is a need for comprehensively capturing customization from all levels of a web application. Furthermore, it is important to know where adaptations do take effect in models, meaning a modeling language needs to specify these subjects of adaptations. This is particularly interesting to know since customization support adds an additional level of complexity to the models. While this is supported by UWE's and Hera's aspect-oriented approaches to modeling customization for the hypertext level, in OO-H the same information is captured within the *Event* and *Condition* parts of its rule language. Concluding, aspect-orientation occurs to represent a suitable mechanism for separately capturing customization. But up to now, aspect-orientation has not been used comprehensively for all levels in a web application model. For example, at the content level aspect-orientation can be used to capture the parts (e.g. attributes) of the user concept that represent context information within a separate aspect, while the application-specific parts remain in the content model.

4.4 Model-driven Engineering Criteria

From Notations to Languages. Most web modeling approaches have emerged, rather focusing on notations than on using standards for specifying their language. Today, with the rise of model-driven engineering, the semantic web and the general need to produce a running system from the web application models, more and more web

modeling approaches provide formal specifications of their languages in terms of either metamodels, UML-profiles, or ontologies.

		Language Definition (M.L)			Model Transformation Type (M.T)				Platform Description Model (M.P)
		Metamodel	Grammar	Semantic Description	PIM2PIM	PIM2PSM	PIM2Code	PSM2Code	
data-oriented	WebML		DTD				Struts		
data-oriented	Hera			RDFS CC/PP			HTML, WML		
hypertext-oriented	WSDM			OWL	✓		XSLT		
object-oriented	OOHDM	~		RDFS, OWL					
object-oriented	UWE	MOF			✓		J2EE/ Cocoon		
object-oriented	OO-H	MOF	DTD, BNF		✓		PHP		
object-oriented	OOWS	MOF		OWL	✓		J2EE/ .Net		

Legend:	
✓	supported
	not supported

Figure 27. Model-driven Engineering

Model Transformations not Based on MDE Standards. Model transformations are supported by almost all approaches in one way or another. Still, only the UWE web modeling language has recently been extended to better support model transformations in the sense of MDE, i.e., through providing QVT transformations. The employment of standards for model transformations would allow benefiting from existing transformation engines of the MDE field.

Lack of Platform Description Models. None of the approaches provides platform description models and consequently no model transformations from platform-independent to platform-specific models are supported. The employment of platform description models together with transformation techniques could broaden the base of application platforms to be employed and if realized within the accompanying tool, could give the approaches a broader application base.

4.5 Tool Support

Lack of (Extensible) Tool Support. One of the most problematic issues in web modeling is the lack of tool support for the individual approaches. Without proper tool support, however, web modeling methods will not gain acceptance in practice. From the set of surveyed approaches, only four provide tool support that has been made available to the public community. Out of the four tools, WebRatio and VisualWade represent the only commercial tools having left the status of a prototype implementation. None of the available tools is offered under an open-source license, which would attract developers of open-source web frameworks and technologies. Although ArgoUWE, the tool accompanying the UWE method, has been built upon the open-source tool ArgoUML, the extensions made are not open-source themselfes. WebRatio is the only tool offering some built-in extension mechanism for the WebML language, i.e., a plug-in mechanism for so-called custom units.

Customization Modeling not Supported by Tools. Currently, all tools provide support for basic web modelling with some deviations from the original notation, however. Still, modeling support for dealing with customization modeling is only provided by Hera's tool which allows modeling context information within user profiles used to statically generate adapted hypermedia presentations. For the customization modeling extensions of WebML and OO-H, prototype implementations have been reported on in literature but are not made available yet.

Support for Model Transformations Provided. Although not based on MDE standards, all tools offer some form of model transformation, e.g., for generating a hypertext model from the content model, or a presentation model from a hypertext model (ArgoUWE, WebRatio, and VisualWade), or for integrating different models as a prerequisite for code generation (HPG).

Code Generation for a Single Platform Only. Except for the ArgoUWE tool, all approaches provide code generation support. Still, code generation is limited to specific platforms, only. While VisualWade generates PHP code, WebRatio targets J2EE platforms by producing code for the Struts framework. WebRatio is shipped with a Tomcat Servlet Container and provides for simple deployment of the web application. The HPG of Hera, however, does support generating static hypermedia presentations in several formats, including HTML as well

as WML. As a consequence, the employment of a web modeling tool also determines the runtime platform which probably is not desired by the developers.

	Version (T.V)	Standalone Application (T.B)	Open Tool (T.O)	Costs (T.C)	Modeling Support (T.M)		Model Pre-Generation (T.MG)	Consistency Check (T.CC)	Code Generation (T.CG)	Process Support (T.P)		Collaboration (T.Co)
					Web Modeling Support	Customization Modeling				Process of Method	Process Type	
WebML: *WebRatio*	4.3 (acad.)	✓	✓	com	✓		✓	od	✓	~	phases	✓
Hera: *Hera Presentation Generator*	1.3	✓		free	✓	~		wm/ od	✓	✓	wizard	
UWE: *ArgoUWE*	0.16			free	✓		✓	wm		~	step-wise	
OO-H: *VisualWade*	1.2 rev 163	✓		com	✓		✓	od	✓	~	step-wise	

Legend:

✓	supported
	not supported
~	partly supported
n/a	not applicable

free	freeware
com	commercial
od	on demand
wm	while modeling

Figure 28. Tool Support

Limited Process Support. Remarkably, only the HPG tool of Hera in combination with its Model Builders implements the process as described by the supported method. All other approaches have only partial tool support for their defined processes. Typically, the tools do not offer means for supporting developers in producing artifacts (e.g., use cases) from earlier development phases but start with designing the content model. WebRatio allows going back and forth between different models as well as making several changes to either of them without loss of previously modeled artifacts. Instead, ArgoUWE as well as OO-H require the user to follow the web applications' "levels dimension" in first designing the content, hypertext, and finally the presentation level, in a linear fashion.

Lack of Collaboration Facilities in Tools. Collaborative work on web application development is currently insufficiently supported. WebRatio, however, as a first step in that direction, foresees the employment of a CVS.

5. RELATED SURVEYS

In an effort to shed light on the different approaches to web application development, some surveys have already been presented. In the following, these surveys are distinguished according to their specific goals and foci into closely related work representing customization modeling surveys and more widely related work representing web modeling surveys focusing on general web modeling criteria, development processes, requirements engineering, and support for modeling rich internet applications.

5.1 Customization Modeling Surveys

Barna et al. provide a comparison of four approaches, amongst them Hera, OOHDM, and UWE also investigated in the present survey [Barna et al. 2003]. The approaches are investigated according to their specific design models for content, hypertext, and presentation levels as well as their support for customization design, though the focus is rather on personalization. The discussion is supported using a simple running example of a virtual art gallery.

In a further evaluation but already some time ago, Kappel et al. compare the customization modeling capabilities of OOHDM and WebML with respect to supported context, granularity of adaptations, and the degree of customizability [Kappel et al. 2001].

Similar to the above mentioned evaluations, this survey's focus is on investigating the support of modeling customization in current web modeling languages but in contrast, specifically considers also the model-driven development of UWAs including tool support. Besides this difference in goals, this survey is also different in terms of comprehensiveness by surveying seven recent web modeling approaches which provide means for customization modeling and if available their tool support. Furthermore, the evaluation is supported with a running example consisting of five different customization scenarios which is used to better explain the general modeling concepts of each approach and in particular the provided means for customization modeling. The evaluation is based on a well-defined as well as fine-grained catalogue of more than 30 criteria, which allows a detailed investigation of each approach with respect to general web modeling characteristics, customization

modeling characteristics, as well as model-driven engineering and tool support. In contrast, the above mentioned related works provide less than five criteria or no explicit description in terms of a catalogue at all.

5.2 Web Modeling Surveys

General Web Modeling. In Schwinger and Koch, an introduction into modeling web applications is given, including a brief overview of eleven web modeling approaches based on a set of twelve criteria, amongst them one criterion evaluating support for customization modeling as well as code generation [Schwinger and Koch 2006]. In contrast, this survey focuses on web modeling approaches providing support for modeling customization. It also differs in that this survey applies a more detailed criteria catalogue as well as a running example while it includes the relevant approaches also surveyed in [Schwinger and Koch 2006].

Development Processes. Nora Koch has evaluated eleven approaches, amongst those only OOHDM and WSDM that are still evolving in [Koch 1999]. That survey specifically focuses on the approaches' development process, supported development phases, modeling techniques and notations used, as well as tool support. In the end, the characteristics of the Rational Unified Process (RUP) [Kruchten 2000] are presented as well as a discussion on how some approaches support parts of RUP. Customization, however, is not a focus as in the survey of that work.

In Woukeu et al. eight approaches are investigated, having in common with this survey the WebML, OOHDM, and WSDM web modeling approaches [Woukeu et al. 2003]. The evaluation is focused on the supported development process, i.e., their phases, as well as the modeling techniques used. Furthermore, each approach's concepts from the hypertext level are listed. Finally, each approach is evaluated if it allows to model read-only or read-write web applications so that the survey differs considerably in terms of focus.

Our survey, aiming at a fine grained set of criteria, has adopted some of the criteria in the works of Nora Koch and Woukeu et al. Furthermore, where appropriate, the criteria have been endowed with a clear definition including a measurement scale or they have been refined. Such a refinement generally means the decomposition of a criterion into several criteria.

Requirements Engineering. In Escalona et al., the scope is requirements engineering for web applications [Escalona and Koch 2004]. A comparison of ten web modeling approaches is provided including WSDM, OOHMD, UWE, and WebML, which are also investigated within this survey. In particular, the types of requirements, the activities and techniques employed during requirements elicitation, specification and validation, and the methodologies' focus on the requirements process, techniques, or artifacts have been evaluated.

In contrast to the work of Escalona et al, this survey is rather concerned with the design level means of today's web modeling languages. Nevertheless, the surveyed approaches are investigated with respect to their support of a requirements engineering phase and more particularly, if customization modeling is already considered during requirements engineering. Consequently, this evaluation is complementary to the one of Escalona et al.

Support for Rich Internet Applications. Preciado et al. compare fifteen representatives from web modeling, multimedia and hypermedia methodologies according to their applicability to model rich internet applications [Preciado et al. 2005]. Again, five of the approaches are also evaluated within this survey, namely, UWE, OO-H, WebML, WSDM, and OOHMD. The set of ten evaluation criteria include multimedia modeling, personalization modeling, and tool support, and are rated according to a weighted measurement scale consisting of four degrees of coverage. Again this survey's focus is different to the one of Preciado et al. which investigates their selection of approaches concerning their applicability to model rich internet applications. Customization modeling, however, is only a marginal concern supported in the work of Preciado et al. with one criterion, only, which in this survey is evaluated in much more detail.

6. SUMMARY

This chapter has presented the state-of-the-art in model-driven development of ubiquitous web applications. More specifically, an in-depth comparison of seven web modeling approaches currently supporting the development of ubiquitous web applications has been provided. An evaluation framework has been designed on the basis of a detailed and well-defined catalogue of evaluation criteria. The actual evaluation by means of this criteria catalogue is supported by a modeling example, i.e., a tourism information web application, used to provide an initial insight into each approaches' concepts for modeling customization as well as to facilitate their comparability. More specifically, a set of five customization scenarios has been defined, to be modeled with each approach. The per-approach evaluation is complemented with a report on lessons learned, summarizing the

approaches' strengths and shortcomings, thus pointing towards possible future developments. This survey points out limitations of current web modeling languages with respect to the model-driven development of ubiquitous web applications lacking of a proper MDE foundation in terms of metamodels as well as missing tools allowing to model customization. Furthermore, the proposed customization mechanisms are often limited, since they neither cover all relevant context factors in an explicit, self-contained, and extensible way, e.g., within a dedicated context model, nor allow for a wide spectrum of extensible adaptation operations. Furthermore, the provided customization mechanisms frequently do not allow dealing with all different parts of a web application in terms of its content, hypertext, and presentation levels as well as their structural and behavioral features. Finally, current web modeling approaches may be extended to better consider the crosscutting nature of customization by not providing the necessary means to comprehensively capture customization separately from all levels of a web application model.

7. ACKNOWLEDGEMENTS

We would like to thank Andreas Schönbeck for his contributions to prior versions of this work.

8. REFERENCES

Abrahao, S. M., Fons, J., Gonzalez, M., and Pastor, O. 2002. Conceptual Modeling of Personalized Web Applications. In Proc. of the 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH 2002), Malaga, Spain. LNCS 2347. 358– 362.

Barna, P., Frasincar, F., and Houben, G.-J. 2002. Specification Framework for Engineering Adaptive Web Applications. In Proc. of the 11th International World Wide Web Conference, Web Engineering Track (WWW 2002), Honolulu, Hawaii, USA.

Barna, P., Frasincar, F., and Houben, G.-J. 2006. A Workflow-driven Design of Web Information Systems. In Proc. of the 6th International Conference on Web Engineering (ICWE 2006), Palo Alto, CA, USA. ACM, 321–328.

Barna, P., Frasincar, F., Houben, G.-J., and Vdovjak, R. 2003. Methodologies for Web Information System Design. In Proc. of the International Conference on Information Technology: Computers and Communications (ITCC 2003), Las Vegas, NV, USA. IEEE Computer Society, 420–424.

Barry, C. and Lang, M. 2003. A comparison of 'traditional' and multimedia information systems development practices. Information & Software Technology 45, 4, 217–227.

Baumeister, H., Knapp, A., Koch, N., and Zhang, G. 2005. Modelling Adaptivity with Aspects. In Proc. of the 5th International Conference on Web Engineering (ICWE 2005), Sydney, Australia. LNCS 3579. 406–416.

Baumeister, H., Koch, N., and Mandel, L. 1999. Towards a UML Extension for Hypermedia Design. In Proc. of the 2nd International Conference on the Unified Modeling Language (UML 1999), Fort Collins, CO, USA. LNCS 1723. 614–629.

Bozzon, A., Comai, S., Fraternali, P., and Carughi, G. T. 2006. Conceptual Modeling and Code Generation for Rich Internet Applications. In Proc. of the 6th International Conference on Web Engineering (ICWE 2006), Palo Alto, CA, USA. ACM, 353–360.

Brambilla, M., Celino, I., Ceri, S., Cerizza, D., Valle, E. D., and Facca, F. M. 2006. A Software Engineering Approach to Design and Development of Semantic Web Service Applications. In Proc. of the 5th International Semantic Web Conference (ISWC 2006), Athens, GA, USA. LNCS 4273. 172–186.

Brambilla, M., Ceri, S., Fraternali, P., and Manolescu, I. 2006. Process Modeling in Web Applications. ACM Trans. Softw. Eng. Methodol. 15, 4, 360–409. Cachero, C., Gomez, J., and Parraga, A. 2001. Migration of Legacy Systems to the Web. In VI Jornadas de Ingenier´ıa del Software y Bases de Datos (JISBD 2001), Almagro (Ciudad Real). 601–614.

Cachero, C., Gomez, J., Parraga, A., and Pastor, O. 2001. Conference Review System: A Case of Study. In 1st International Workshop on Web-Oriented Software Technology (IWWOST 2001), Valencia, Spain.

Cachero, C., Gomez, J., and Pastor, O. 2000. Object-Oriented Conceptual Modeling of Web Application Interfaces: the OO-HMethod Abstract Presentation Model. In Proc. of the 1st International Conference on Electronic Commerce and Web Technologies (EC-Web 2000), London, UK. LNCS 1875. 206–215.

Casteleyn, S., Garrigos, I., and Troyer, O. D. 2005. Automatic Runtime Validation and Correction of the Navigational Design of Web Sites. In 7th Asia-Pacific Web Conference on Web Technologies Research and Development (APWeb 2005), Shanghai, China. LNCS 3399. 453–463.

Casteleyn, S., Plessers, P., and Troyer, O. D. 2006. On Generating Content and Structural Annotated Websites Using Conceptual Modeling. In Proc. of the 25th International Conference on Conceptual Modeling (ER 2006), Tucson, AZ, USA. LNCS 4215. 267–280.

Casteleyn, S., Troyer, O. D., and Brockmans, S. 2003. Design Time Support for Adaptive Behavior in Web Sites. In Proc. of the 18 th Symposium on Applied Computing (SAC 2003), Melbourne, FL, USA. 1222–1228.

Casteleyn, S., Woensel, W. V., and Houben, G.-J. 2007. A Semantics-based Aspect-Oriented Approach to Adaptation in Web Engineering. In Proceedings of the 18th Conference on Hypertext and Hypermedia (HT 2007), Manchester, UK. 189–198.

Ceri, S., Daniel, F., and Facca, F. M. 2006. Modeling Web Applications reacting to User Behaviors. Computer Networks 50, 10, 1533–1546.

Ceri, S., Daniel, F., Facca, F. M., and Matera, M. 2007. Model-Driven Engineering of Active Context-Awareness. World Wide Web, Volum 10, Issue 4, December.

Ceri, S., Daniel, F., and Matera, M. 2003. Extending WebML for Modeling Multi-Channel Context-Aware Web Applications. In Proceedings of the Workshop on Mobile Multi-channel Information Systems, in conjuntion with WISE 2003, Rome, Italy. 615–626.

Ceri, S., Daniel, F., Matera, M., and Facca, F. M. 2007. Model-driven Development of Context-Aware Web Applications. ACM Transactions on Internet Technology 7, 1.

Ceri, S., Fraternali, P., Bongio, A., Brambilla, M., Comai, S., and Matera, M. 2003. Designing Data-Intensive Web Applications, 1st ed. Morgan Kaufmann.

Chen, P. P. 1976. The Entity-Relationship Model - Toward a Unified View of Data. ACM Transactions on Database Systems 1/ 1, 9–36.

Cowan, D. D. and de Lucena, C. J. P. 1995. Abstract Data Views: An Interface Specification Concept to Enhance Design for Reuse. IEEE Trans. Software Eng. 21, 3, 229–243.

Escalona, M. J. and Koch, N. 2004. Requirements Engineering for Web Applications – A Comparative Study. Journal of Web Engineering 2/3, 193–212.

Fiala, Z., Frasincar, F., Hinz, M., Houben, G.-J., Barna, P., and Meißner, K. 2004. Engineering the Presentation Layer of Adaptable Web Information Systems. In Proc. of the 4th International Conference on Web Engineering (ICWE 2004), Munich, Germany. LNCS 3140, Vol. 3140. 459–472.

Fiala, Z., Hinz, M., Houben, G.-J., and Frasincar, F. 2004. Design and Implementation of Component-based Adaptive Web Presentations. In Proc. of the Symposium on Applied Computing (SAC 2004), Nicosia, Cyprus, 2004. 1698–1704.

Fons, J., Pelechano, V., Albert, M., and Pastor, O. 2003. Development of Web Applications from Web Enhanced Conceptual Schemas. In Proc. of the 22nd International Conference on Conceptual Modeling (ER 2003), Chicago, IL, USA. LNCS 2813. 232–245.

Frasincar, F., Barna, P., Houben, G.-J., and Fiala, Z. 2004. Adaptation and Reuse in Designing Web Information Systems. In Proc. of the International Conference on Information Technology: Coding and Computing (ITCC 2004), Las Vegas, Nevada, USA. 387–291.

Frasincar, F. and Houben, G.-J. 2002. Hypermedia Presentation Adaptation on the Semantic Web. In Proc. of the 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH 2002), Malaga, Spain. LNCS 2347. 133–142.

Frasincar, F., Houben, G.-J., and Barna, P. 2006. HPG: the Hera Presentation Generator. Journal of Web Engineering 5/2, 175–200.

Frasincar, F., Houben, G.-J., and Vdovjak, R. 2001. An RMM-Based Methodology for Hypermedia Presentation Design. In Proc. of the 5th East European Conference on Advances in Databases and Information Systems (ADBIS 2001), Vilnius, Lithuania. LNCS 2151. 323–337.

Garrigos, I., Casteleyn, S., and Gomez, J. 2005. A Structured Approach to Personalize Websites Using the OO-H Personalization Framework. In Proc. of the 7th Asia-Pacific Web Conference on Web Technologies Research and Development (APWeb 2005), Shanghai, China. LNCS 3399. 695–706.

Garrigos, I., Cruz, C., and Gomez, J. 2007. A Prototype Tool for the Automatic Generation of Adaptive Websites. In Proc. of the 2nd International Workshop on Adaptation and Evolution in Web Systems Engineering (AEWSE 2007), in conjunction with ICWE 2007, Como, Italy. CEUR Workshop Proceedings.

Garrigos, I. and Gomez, J. 2006. Modeling User Behaviour Aware WebSites with PRML. In 3rd Workshop on Web Information Systems Modelling (WISM 2006), in conjunction with ICWE 2006, Palo Alto, California, USA.

Garrigos, I., Gomez, J., Barna, P., and Houben, G.-J. 2005. A Reusable Personalization Model in Web Application Design. In 2nd Workshop on Web Information Systems Modelling (WISM 2005), in conjunction with ICWE 2005, Sydney, Australia.

Garrigs, I., Gomez, J., and Cachero, C. 2003a. Modelling Adaptive Web Applications. In Proc. of the IADIS International Conference WWW/Internet, Algarve, Portugal. 813–816.

Garrigs, I., Gomez, J., and Cachero, C. 2003b. Modelling Dynamic Personalization in Web Applications. In Proc. of the 3rd International Conference on Web Engineering (ICWE 2003), Oviedo, Spain. LNCS 2722. 472–475.

Gomez, J., Bia, A., and Parraga, A. 2005. Tool Support for Model-Driven Development of Web Applications. In 6th International Conference on Web Information Systems Engineering (WISE 2005), New York, NY, USA. LNCS, 3806. 721–730.

Gomez, J., Cachero, C., and Pastor, O. 2000. Extending a Conceptual Modelling Approach to Web Application Design. In Proc. of the 12th International Conference on Advanced Information Systems Engineering (CAISE 2000), Stockholm, Sweden. LNCS 1789. 79–93.

Gomez, J., Cachero, C., and Pastor, O. 2001. Conceptual Modeling of Device-Independent Web Applications. IEEE MultiMedia 8, 2, 26–39.

Grün, Ch, Schwinger W., Pröll B., Retschitzegger W., Werthner, H. 2006. Pinpointing Tourism Information onto Mobile Maps – A Light-Weight Approach. In Proceedings of ENTER 2006, 18-20 January 2006, Lausanne, Switzerland.

Halpin, T. 2001. Information Modeling and Relational Databases. Morgan Kaufmann.

Hester, A., Borges, R., and Ierusalimschy, R. 1998. Building Flexible and Extensible Web Applications with Lua. Journal of Universal Computer Science 4, 9, 748–762.

Houben, G.-J., Frasincar, F., Barna, P., and Vdovjak, R. 2004. Modeling User Input and Hypermedia Dynamics in Hera. In Proc. of the 4th International Conference on Web Engineering, Munich, Germany. LNCS 3140. Springer, 60–73.

Isakowitz, T., Stohr, E. A., and Balasubramanian, P. 1995. RMM: A Methodology for Structured Hypermedia Design. Communications of the ACM 38, 8, 34–44.

Jacyntho, M. D., Schwabe, D., and Rossi, G. 2002. A Software Architecture for Structuring Complex Web Applications. J. Web Eng. 1, 1, 37–60.

Kappel, G., Pröll, B., Reich, S., and Retschitzegger, W. 2006. An Introduction to Web Engineering. In Web Engineering - Systematic Development of Web Applications, G. Kappel,

B. Pröll, S. Reich, and W. Retschitzegger, Eds. Wiley, 1–21. Kappel, G., Pröll, B., Retschitzegger, W., and Schwinger, W. 2003. Customisation for Ubiquitous Web Applications a Comparison of Approaches. International Journal of Web Engineering and Technology 1, 1, 79–111.

Kappel, G., Pröll, B., Retschitzegger, W., Schwinger, W., and Hofer, T. 2001. Modeling Ubiquitous Web Applications - A Comparison of Approaches. In Proc. of the 3rd International Conference on Information Integration and Web-based Applications & Services (IIWAS 2001), Linz, Austria.

Koch, N. 1999. A Comparative Study of Methods for Hypermedia Development. Tech. Rep. 9905, Ludwig-Maximilians-University Munich, Germany.

Koch, N. 2001. Software Engineering for Adaptive Hypermedia Systems: Reference Model, Modeling Techniques and Development Process. Ph.D. thesis, Ludwig-Maximilians-University Munich, Germany.

Koch, N. 2007. Classification of Model Transformation Techniques used in UML-based Web Engineering. IET Software 1, 3, 98–111.

Koch, N. and Kraus, A. 2002. The expressive Power of UML-based Web Engineering. In Proc of the 2nd International Workshop on Web-oriented Software Technology (IWWOST 2002), in conjunction with ECOOP 2002, Malaga, Spain. 21–32.

Koch, N. and Kraus, A. 2003. Towards a Common Metamodell for the Development of Web Appliactions. In Proc. of the 3rd International Conference on Web Engineering (ICWE 2003), Oviedo, Spain. LNCS 2722. 497–506.

Koch, N., Kraus, A., Cachero, C., and Melia, S. 2004. Integration of Business Processes in Web Application Models. Journal of Web Engineering 3, 1, 22–49.

Koch, N. and Wirsing, M. 2002. The Munich Reference Model for Adaptive Hypermedia Applications. In Proc. of the 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH 2002), Malaga. LNCS 2347. 213–222.

Koch, N., Zhang, G., and Escalona, M. J. E. 2006. Model transformations from requirements to web system design. In Proc. of the 6th International Conference on Web Engineering (ICWE 2006), Palo Alto, CA, USA. ACM, 281–288.

Kraus, A. and Koch, N. 2002. Generation of Web Applications from UML. Models using an XML Publishing Framework. In Proc. of the 5th World Conference on Integrated Design and Process Technology (IDPT 2002), Pasadena, CA, USA.

Kruchten, P. 2000. The Rational Unified Process: An Introduction. Addison-Wesley. Manolescu, I., Brambilla, M., Ceri, S., Comai, S., and Fraternali, P. 2005. Model-Driven Design and Deployment of Service-Enabled Web Applications. ACM Transactions on Internet Technology 5, 3, 439–479.

Moreno, N., Fraternalli, P., and Vallecillo, A. 2007. WebML modelling in UML. IET Software 1, 3, 67–80.

OMG, O. 2002. Meta Object Facility (MOF) Specification 1.4. http://www.omg.org/docs/formal/02-04-03.pdf.

OMG, O. 2005. Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification. Final Adopted Specification. ptc/05-11-01.

OMG, O. M. G. 2003. MDA Guide Version 1.0.1. http://www.omg.org/docs/omg/03-06-01.pdf.

Pastor, O., Abrahao, S., and Fons, J. 2000. OOWS: An Object-Oriented Approach for Web- Solutions Modeling. In Proc. of the International Conference on Information Society (ICIS 2000), Ljubljana, Slovenia.

Pastor, O., Fons, J., Pelechano, V., and Abrahao, S. 2006. Conceptual Modelling of Web Applications: The OOWS Approach. In Web Engineering: Theory and Practice of Metrics and Measurement for Web Development, E. Mendes and N. Mosley, Eds. Springer, 277–302.

Pastor, O., Insfran, E., Pelechano, V., Romero, J., and Merseguer, J. 1997. OO-METHOD: An OO Software Production Environment Combining Conventional and Formal Methods. In Proc. of the 9th International Conference on Advanced Information Systems Engineering (CAISE 1997), Barcelona, Catalonia, Spain. LNCS 1250. 145–158.

Paterno, F. 2000. Model-Based Design of Interactive Applications. ACM Intelligence 11, 4, 26–38.

Plessers, P., Casteleyn, S., and Troyer, O. D. 2005. Semantic Web Development with WSDM. In Proc. of the 5th International Workshop on Knowledge Markup and Semantic Annotation (SemAnnot2005), in conjunction with ISWC 2005, November 6-10, Galway, Irland. CEUR Workshop Proceedings.

Preciado, J. C., Trigueros, M. L., Sanchez, F., and Comai, S. 2005. Necessity of methodologies to model Rich Internet Applications. In Proc. of the 7th IEEE International Workshop on Web Site Evolution, Budapest, Hungary. 7–13.

Rojas, G., Valderas, P., and Pelechano, V. 2006. Describing Adaptive Navigation Requirements of Web Applications. In Proc. of the 4th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH 2006), Dublin, Ireland. LNCS 4018. 318–322.

Rossi, G. and Schwabe, D. 2006. Model-Based Web Application Development. In Web Engineering: Theory and Practice of Metrics and Measurement for Web Development, E. Mendes and N. Mosley, Eds. Springer, 203–333.

Rossi, G., Schwabe, D., de Lucena, C. J. P., and Cowan, D. D. 1995. An Object-Oriented Model for Designing the Human-Computer Interface Of Hypermedia Applications. In Proc. of the International Workshop on Hypermedia Design (IWHD 1995), Montpellier, France. 123–143.

Rossi, G., Schwabe, D., and Guimaraes, R. 2001. Designing Personalized Web Applications. In Proc. of the 10th International World Wide Web Conference (WWW 2001), Hong Kong, China. 275–284.

Rossi, G., Pastor, O., Schwabe, D., Olsina, L., (eds.) 2007. Web Engineering: Modelling and Implementing Web Applications, Human-Computer Interaction Series, Springer.

Schauerhuber, A., Schwinger, W., Wimmer, M., Retschitzegger, W., and Kappel, G. 2007. A Survey on Web Modeling Approaches for Ubiquitous Web Applications. Tech. rep., Vienna University of Technology. Oct.

Schauerhuber, A., Wimmer, M., Kapsammer, E., Schwinger, W., and Retschitzegger, W. 2007. Briding WebML to model-driven engineering: from document type definitions to meta object facility. IET Software 1, 3, 81–97.

Schwabe, D., de Almeida Pontes, R., and Moura, I. 1999. OOHDM-Web: An Environment for Implementation of. Hypermedia Applications in the WWW. ACM SIGWEB Newsletter 8, 2, 18–34.

Schwabe, D., Guimaraes, R., and Rossi, G. 2002. Cohesive Design of Personalized Web Applications. IEEE Internet Computing 6, 2, 34–43.

Schwabe, D. and Rossi, G. 1994. From Domain Models to Hypermedia Applications: An Object-Oriented Approach. In International Workshop on Methodologies for Designing and Developing Hypermedia Applications, Edinburgh.

Schwabe, D. and Rossi, G. 1998. An Object Oriented Approach to Web-Based Applications Design. Theory and Practice of Object Systems 4, 4, 207–225.

Schwabe, D., Rossi, G., and Barbosa, S. D. J. 1996. Systematic Hypermedia Application Design with OOHDM. In Proc. of the 7th Conference on Hypertext (HT 1996), Washington DC. 116–128.

Schwabe, D. and Salim, C. S. 2002. Integrating Knowledge Management Applications in the Enterprise The Xerox Knowledge Portal Project. Knowledge and Process Management 9, 3, 190–201.

Schwinger, W. and Koch, N. 2006. Modeling Web Applications. In Web Engineering – Systematic Development of Web Applications, G. Kappel, B. Pröll, S. Reich, and W. Retschitzegger, Eds. Wiley, 39–64.

Torres, V., Fons, J., Pelechano, V., and Pastor, O. 2004. Navigational modeling and the semantic web. an ontology based approach. In Proceedings of the Joint Conference 10th Brazilian Symposium on Multimedia and the Web & 2nd Latin American Web Congress, (WebMedia & LA-Web 2004), Ribeirao Preto-SP, Brazil. IEEE Computer Society, 94–96.

Torres, V., Pelechano, V., Ruiz, M., and Valderas, P. 2005. A Model Driven Approach for the Integration of External Functionality in Web Applications. The Travel Agency System. In Proc. of 1st Workshop on Model-Driven Web Engineering, in conjunction with ICWE 2005), Sydney, Australia. 1–11.

Troyer, O. D. and Casteleyn, S. 2004. Designing Localized Web Sites. In Proc. of the 5th International Conference on Web Information Systems Engineering (WISE 2004), Brisbane, Australia. LNCS 3306. 547–558.

Troyer, O. D. and Leune, C. J. 1998. WSDM: A User Centered Design Method for Web Sites. Computer Networks 30, 1-7, 85–94.

Valderas, P., Fons, J., and Pelechano, V. 2005. Using Task Descriptions for the Specification of Web Application Requirements. In Anais do WER05 - Workshop em Engenharia de Requisitos, Porto, Portuga. 257–268.

Valverde, F., Valderas, P., Fons, J., and Pastor, O. 2007. A MDA-based Environment for Web Applications Development: From Conceptual Models to Code. In International Workshop on Web-oriented Software Technology (IWWOST 2007), in conjunction with ICWE 2007, Como, Italy.

van der Sluijs, K., Houben, G.-J., Broekstra, J., and Casteleyn, S. 2006. Hera-S – Web Design Using Sesame. In Proc. of the 6th International Conference on Web Engineering (ICWE 2006), Palo Alto, CA, USA. 337–344.

Vdovjak, R. and Houben, G.-J. 2002. Providing the Semantic Layer for WIS Design. In Proc. of the 14th International Conference on Advanced Information Systems Engineering (CAISE 2002), Toronto, Canada. LNCS 2348. 584–599.

Vilain, P., Schwabe, D., and de Souza, C. S. 2000. A Diagrammatic Tool for Representing User Interaction in UML. In Proc. of the 3rd International Conference on the Unified Modeling Language (UML 2000), York, UK. LNCS 1939. 133–147.

W3C, 2004a. Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies 1.0. http://www.w3.org/TR/CCPP-struct-vocab/.

W3C, 2004b. RDF Primer. (XML) 1.1 (Second Edition). http://www.w3.org/TR/rdfprimer/.

W3C, 2004c. OWL Web Ontology Language Overview. http://www.w3.org/TR/owl-features/.

Retschitzegger, W. and Schwinger, W. 2000. Towards Modeling of Data Web Applications A Requirements' Perspective. In Proc. of Americas Conference on Information Systems (AMCIS 2000), Long Beach, USA.

Woukeu, A., Carr, L., Wills, G., and Hall, W. 2003. RethinkingWeb Design Models: Requirements for Addressing the Content. Tech. Rep. ECSTR-IAM03-002, University of Southampton.