Wissen, M. & Ziegler, J. (2003). A Methodology for the Component-Based Development of Web Applications. In Stephanidis, C.; Jacko, J. (Eds.): Proceedings of 10th Int. Conf. on Human - Computer Interaction (HCI International 2003), Vol. 1, Crete, Greece

A Methodology for the Component-Based Development of Web Applications

Michael Wissen, Juergen Ziegler

Fraunhofer IAO
Nobelstrasse 12
Stuttgart, Germany
{Michael.Wissen, Juergen.Ziegler}@iao.fhg.de

Abstract

This paper presents a universal and integrated methodology for the component-based design of web applications. Modelling methods for design areas relevant in the web context are explicated which are enhancing the concepts of existing design methods by aspects of metadata modelling as well as of view and navigation modelling. Core of the approach is the application of metamodels and modelling techniques as well as tools for prototypical implementation of developed models in a run-capable web application.

1 Introduction

The development of web-based information systems has been technologically supported for some years by a multiplicity of different systems, such as web editors or content management systems. The focus of these technologies lies on the design and flexible development of graphic representations of contents, placing and administering of contents within an editing process as well as the connection to data bases and existing applications. Contrary to conventional applications, web-based applications are often subject to continuous revision regarding their structures as well as their contents which entails an extensive new design of the application when using current procedures and systems. Typical characteristics of current web applications are a mixture of information objects of varying structuring degrees, high modification dynamics and inclusion of external information components and services. Design techniques and development processes have to answer the questions connected with this changed situation and, at the same time, meet the requirements of the widely expanded spectra of diverse designer and developer roles.

2 State of the Art

The use of systematic design methods is yet uncommon in the development of web applications. Studies like Barry & Lang (Barry & Lang, 2001) show that specific methods are hardly used, particularly in the media field, and if so, they have a low standardization degree. Even standard methods like UML are applied as yet only rarely in the web area. Several approaches of methods were proposed for specific aspects of the web development. Some of these methods refer to the modelling of the hypermedial structure of web sites, such as HDM (Garzotti et al., 1993), RMM (Isakowitz & Stohr, 1995), OOHDM (Schwabe & Rossi, 1994) as well as the UML-extension for the modelling of hypermedial systems (Baumeister et al., 1999). However, these methods usually are inadequately integrable with other development perspectives and do not possess sufficient tool support. Moreover, these methods lack, on one hand, the possibility for timely evaluation of efficiency, navigationability and representation of a web application with a generated prototype and, on the other hand, the conceptual support of dynamically changing scope of concepts and contents.

3 Ontology-based Approach for Modelling Web Applications

This paper presents methods and metamodels for the systemization of the construction of web applications and support of the entire development process up to service and maintenance of web applications. Basis of the methodology is a method compound containing concepts and technologies for significant modelling aspects. The concept of the method compound enables software-based tools to support the generation process semiautomatically so that web models generated by the user can be represented prototypically. This permits initial application of mechanisms for efficiency evaluation of structure, navigation, and representation. Anchor point is an ontological description (Lenat & Guha, 1990) of the information base which serves as basis for the systematic design of a navigation and view structure. Based on this, so-called navigational classes enable the automated classification of information objects in individual subjects of the navigation structure with consideration to the different views. Moreover, distributed services available, such as web services, can be integrated in the form of functional components.Construction of a web application site and navigation structure is accomplished based on various components. The web structure is generated with the aid of containers and partitions which have the capacity to be embedded into one another for representation purposes so that also nested web pages can be represented. The method compound is expanded by procedures which provide the developer of a web application with possibilities for modelling and integration of context-dependent information offers. So-called view classes are made available which enable a role-dependent, respectively location-dependent.

The design process of a web application is composed of the following steps: design of metadata model (topic structure), generation of a navigation model, formal analysis of user requirements, derivation of view structure, configuration of external services and definition of graphic representation and interaction.

2.1 Conceptual Model

The resource model serves as basis for the construction of the topic structure. It contains concepts as well as individual information objects in the form of documents, tasks, processes as instances as well as their associations to other instances. This, however, is not sufficient for a defined navigation in the subsequent web application, since it is not possible at this point to access information components directly. But with the aid of an ontology, explorative accesses within the web application can be realized and, moreover, navigation possibilities can be expanded to dynamic content.

2.2 Component Design and Structuring

In this step, essential elements of a web application are described in components, which are composed into the form of component models. Structure and construction can be modeled using containers and partitions which can be realized, e.g., by frames in the subsequent web application.
Atomic components are concrete objects which cannot be divided further. This includes, e.g., texts, images, multimedia objects, external references, etc. Atomic components are considered instances of their respective atomic class within the conceptual model and, primarily, serve the purpose of unitary viewing of different objects of the navigation model.
The elements describing individual web sites are kept in so-called *containers* which can be embedded into one another so that nested web sites can also be represented. The navigation model, which is the basis for the subsequent representation of the web application, is based on a HiGraph-like approach (Harel et al., 1987). Several subcontainers within a container, which are on the same level, are represented in accordance with an XOR link, i.e., in this case, always exactly one subcontainer is visible. For a parallel representation of different containers, containers are divided into individual *partitions* (cf. Fig. 1).

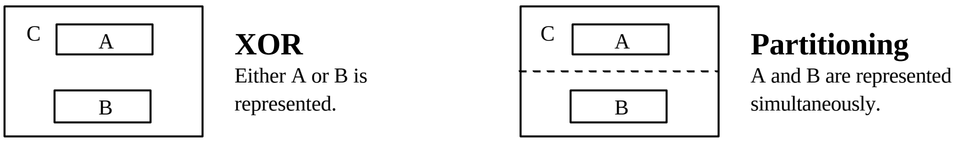

Fig. 1: Containers and Partitions

Atomic classes are considered independent containers which, however, cannot be divided further. If there are several subcontainers on the same level of a container, it needs to be determined which container is visible at first. To do so, a initial state is defined, pictured by an arrow starting from a dot. Containers as well as partitions define the destination area where the contents are to be represented later. With the aid of place holders (dashed component), partial areas of the web application can be represented schematically for better clarity. Detailed modelling of the area coarsely represented by the placeholder, can be done elsewhere.
An alteration of the contents represented on a web page is induced by an *event* which is typically initiated by interaction by a user or caused automatically (e.g., by lapse of time or previous events). Processing the event is performed by a *link* which will prompt an altered representation of a web application. The destination area of a link defines the container, respectively the partition, where changes are to be made. The current context is carried along in a link in form of the source instance. The source instance is the instance of a class represented by the link. Transfer of the context is necessary in order to serve dependencies of the information to be built with the current context. Since the destination area cannot be identified by an atomic component, in addition, a destination class must be known to the link. Based on this and the source instance, it is possible to detect the instance, respectively the set of instances. If the destination class is an atomic class, the source instance is the component in which the link was defined. This component can be either a container or a partition. It is, therefore, possible to refer to arbitrary objects within a web application which are not contained in the ontology. In the graphic representation of the navigation model, links define the transition from one component to another.
In a destination area, either an individual instance of a class can be represented or, if several instances are linked to one class, an overview on the instances available can be displayed with the possibility to view the instances individually at a defined location. For this purpose, *navigational classes* are defined which enable navigation of a set of instances with the aid of an overview. The manner in which this overview is represented, can be determined either by the navigational class

itself or has already been determined in the underlying navigation model. In the latter case, a representation type is selected from a set of predefined navigation models. This set includes, according to Fig. 2, types such as tree, index, sequence, etc. (cf. Ziegler, 1997). If the representation type of the navigational class is detected dynamically, the cardinality of the instance set to be represented, is applied. The set of the instances to be represented can be described with the aid of the instance of the original class and the original relation. The selection of attributes represented is defined in the attribute set of the destination instance and arranged according to a sorting function. The graphic representation of navigational classes consists of two components which are linked by a navigation relation. The navigation relation is a parameterizable link which is initialized by the individual instances in the navigational class when called. It is marked by an open arrowhead.

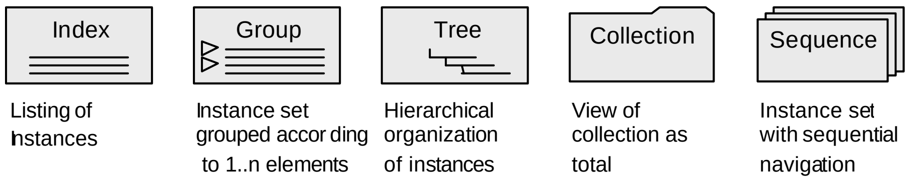

Fig. 2: Selection of different types of navigational classes

The *composition model* defines the compilation of arbitrary components into individual web sites. It specifies, with the aid of containers and partitions, the structure of a web site and the site-inherent navigation. The *navigation model* is composed of the composition models of the individual web sites and an additional higher-ranking navigation structure. This defines the navigation between the individual web sites. Fig. 3 is an example of the navigation model of a portal page. In the entire web application, a header line is to be found in the upper area, directly beneath it is a menu bar with a search field attached to the right in form of an application. Beneath these two components, there is a further partition. It contains in the start condition at first a login field on the left. Upon successful login, it is replaced by a vertical menu with an overlying image. On the right, nothing is represented at first, only upon finalization of the login, the ticker is activated. The lower area represents the main area of the application. It is, at the same time, the destination area for both menu bars and the search field. The different components, which are represented in this area, can now be specified more precisely at another location.

2.3 View Model

Different views on existing information are created with the aid of view classes, with the objective of generating an information supply customized to user requirements. Their mission is representing defined characteristics of an instance of a class under consideration of constraints. A resource class can encompass several view classes with which the different views on the underlying contents can be realized. Similar to OOHDM, a SQL-like query language would be the obvious choice for the definition of the view classes.

2.4 Representation Model

The representation model serves the transformation of the conceptual schema on a lower-level representation and basically contains the information necessary for graphic representation. This includes exact information on position and graphic design of the contents and interaction interfaces to be represented. These information items are assigned to the individual view classes using style sheets and retrieved at runtime.

3 Implementation and Future Development

With the aid of parameterizable components, which generate, based on defined metadata and navigational models on runtime, dynamic and context-sensitive navigation structures, models generated by the user can be realized in a runtime environment for test and verification possibilities. For this, processes for automatic generation of navigation structures based on the developed models are designed and implemented. Their high reusability rate enables the implementation of a modeled web application in a run-capable system.

This paper describes the procedure for systematic and universal development of web applications. Special attention has been paid to a most easy perceivability of the models and especially to the support possibilities by software-based tools which are to relieve the user in generation and verification of the models to the greatest possible extent. In a further step, the component-oriented approach could be extended to model components available in the web which are characterized by an integrative description. It would offer the developer of a web application the advantage of being able to access already existing self-generated or external solutions for certain components of his application.

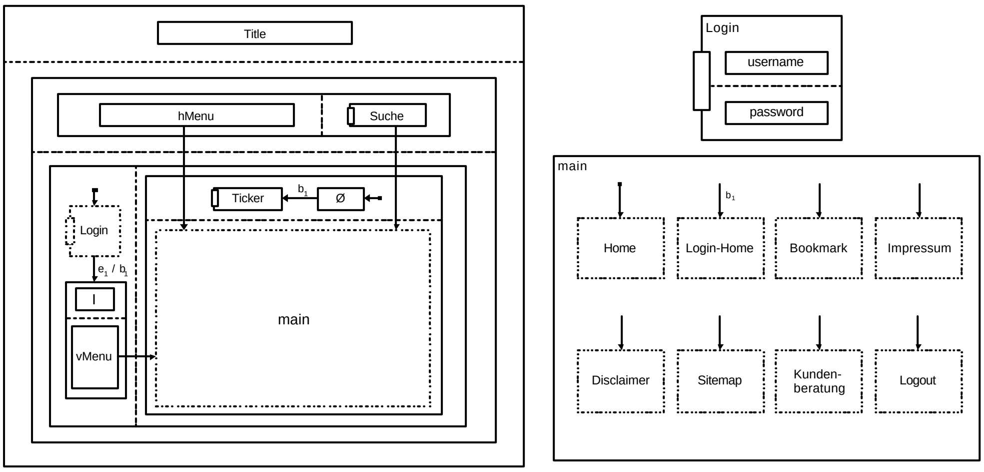

Fig. 3: Navigation model (Excerpt from a portal)

References

Barry, C.; Lang, M. (2001): A Survey of Multimedia and Web Development Techniques and Methodology Usage. IEEE Multimedia. 8(3), 52-60

Baumeister, H.; Koch, N.; Mandel, L. (1999): Towards a UML extension for hypermedia design. In UML'99, Fort Collins, USA

Garzotti, F.; Paolini, P.; Schwabe, D. (1993): HDM – A model-based approach to hypermedia application design. ACM-Transactions on Information Systems, 11/1

Harel, D.; Pnueli, A.; Schmidt, J. P.; Sherman, R. (1987): On the formal semantics of statecharts. Proc. 2nd. IEEE Symposium on Logic in Computer Science, Ithaca, N.Y.

Isakowitz, T.; Stohr, E. A.; Balasubramanian, P. (1995): RMM: A Methodology for Structured Hypermedia Design. Communications of the ACM, 38/8:34-44

Lenat, D. B.; Guha, R. V. (1990): Building large knowledge-based systems. Addison-Wesley

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

Ziegler, J. E. (1997): ViewNet - Conceptual design and modelling of navigation. In S. Howard, J. Hammon & G. Lingaard (Eds.), Human-Computer Interaction: Interact'97, London: Chapman & Hall