Systematic Design of Web Applications with UML

Rolf Hennicker[1]

[1]Institute of Computer Science
Ludwig-Maximilians University of Munich
Oettingenstr. 67,
D-80538 München, Germany
{hennicke,kochn}@informatik.uni-muenchen.de

Nora Koch[1,2]

[2] F.A.S.T.
Applied Software Technology GmbH
Arabellastr. 17,
D-81925 München, Germany
koch@fast.de

Abstract. We propose a systematic design method for Web applications which takes into account the navigation space and the presentational aspects of the application. The method is based on a UML profile for the Web domain. Starting with a use case analysis and a conceptual model of the application we first provide guidelines for modeling the navigation space. From the navigation space model we can derive, in a subsequent step, a navigational structure model which shows how to navigate through the navigation space using access elements like indexes, guided tours, queries and menus. Finally, a static and dynamic presentation model is constructed. The different models of the design process are represented by using a Web extension of UML.
The strength of the presented methodology is that most steps can be performed in a semi-automatic way, thus providing the basis for a systematic mechanism for Web design.

Keywords: Unified Modeling Language, Web Application, Design Method, Systematic Development, UML Extension, Web Engineering

INTRODUCTION

Web engineering is a new and still evolving discipline. The process of learning how to develop large Web applications has just begun. Web applications are mostly the result of an ad hoc implementation, growing usually from small to large applications that very quickly become difficult to maintain. Some guidelines and tools are beginning to appear that assist developers of Web applications, but these current practices often fail due to inappropriate techniques, processes and methodologies. The objective is to develop a suitable method that allows high-quality Web applications to be produced through the systematic construction of high-quality models.
People with different skills are involved in the process of Web systems development, such as authors, layout designers, programmers, multimedia experts and marketing specialists. The role of the user is

In Unified Modeling Language: Systems Analysis, Design and Development Issues. (2001). K. Siau and T. Halpin (Eds.). Idea Group Publishing, 1-20.

augmented and makes it more difficult to capture the requirements of the application. The non-linearity of the hyperdocuments as well as the possibility to connect easily to other Web applications increases the complexity and risk of "lost in the hyperspace". Web engineering has also to take into account aesthetic and cognitive aspects that general software engineering environments do not support (Lowe & Hall, 1999). Moreover, the development process tends to be more fine grained, more incremental and iterative. Maintenance is an even more significant part of the lifecycle of Web applications than in traditional systems since Web technologies and user requirements change continuously. In addition, non-functional requirements, such as security have to be addressed by a Web development process.

If we restrict ourselves to the design steps, the main differences we can observe between design of Web solutions and other software applications are the heterogeneity of the designer group, the hypertext structure of nodes and links, the need for navigational assistance, the inclusion of searching and indexing mechanisms, and the presentation of the multimedia contents and this contents e.g. for different browsers. Thus, the design is centered around three main aspects of Web systems: the content, the navigational structure and the presentation. Treating these aspects separately during design will payoff in the maintenance phase.

In this work we concentrate our attention on the analysis and design workflows of an engineering process based on the Unified Process (Jacobson, Booch & Rumbaugh, 1999) adapted for Web applications. The approach we propose consists of a method and a notation. The notation is a UML profile that defines a set of appropriate stereotypes for the Web by using the UML extension mechanism. The stereotypes are used to indicate the descriptive and restrictive properties that the modeling elements have in comparison to standard UML elements.

The method consists of three steps that are performed in an iterative design process. The steps are the conceptual, navigational and presentational design. The artifacts produced as results of the application of the method are models represented by UML diagrams.

They are:

- a conceptual model for the content,
- a navigation space model and a navigational structure model,
- a static and a dynamic presentation model.

The conceptual model is built by taking into account the functional requirements captured with use cases. Traditional object-oriented techniques are used to construct the conceptual model, such as finding classes, defining inheritance structures and specifying constraints. It is represented by a UML class diagram.

Based on this conceptual model, the navigation space model is constructed. It is also represented as a static class diagram showing which objects can be visited through navigation. A set of guidelines is proposed for modeling the navigation space. A detailed specification of (navigation) associations, their multiplicity and role names establishes the base for deriving a navigational structure model. In this step, access structures such as indexes, guided tours, queries and menus are incorporated. The navigational structure model defines the structure of nodes and links of the Web application showing how navigation is supported by the access structures.

Navigational design is a critical step in the design of Web applications. Even simple applications with a non-deep hierarchical structure become complex very quickly by the addition of new links. Additional links improve navigability on the one hand but imply on the other hand higher risk for loss of

orientation. Building a navigation model is not only helpful for the documentation of the application structure, it also allows for a more structured increase of navigability. Navigational models are also represented as UML class diagrams that are built with particular stereotypes.
In the last step of our method we construct a presentational model which consists of two parts: a static model of an abstract user interface based on framesets and a model of the dynamic aspects of the presentation. The dynamic presentation model takes into account multi-window techniques. UML class diagrams and interaction diagrams are used for the representation of these models.
The focus of our methodology is on the structure of the navigation rather than on the dynamics and architecture of Web applications which are considered in Conallen (1999). The strength of our method is the fact that a systematic approach for the construction of the models is described, identifying as many steps as possible that can be performed in an automatic way. This consequently provides the basis for a systematic mechanism for Web design.
This chapter is structured as follows: Section 2 provides a brief description of related work. Section 3 describes the starting points for the modeling process: use cases and a conceptual model. Section 4 gives guidelines to build a navigation space model based on the conceptual model. In Section 5 we present a procedure to derive the navigational structure model from the navigation space model. Section 6 shows how a static presentation model is obtained from the navigational structure model and, in Section 7, the dynamic aspects of the presentation are modeled. Finally, Section 8 gives some concluding remarks and an overview of future trends.

BACKGROUND

Over the last few years many methods for Web design have been proposed; see Lowe and Hall (1999) for an overview. Most of them are not based on the UML, like RMM (Relationship Management Methodology) of Isakowitz, Stohr and Balasubramnian (1995) and OOHDM (Object-Oriented Hypermedia Design Method) of Schwabe and Rossi (1998). They utilize entity-relationship diagrams, OMT or their own notation and techniques.
Recently, some new approaches have proposed UML extensions for the Web domain, such as the development process of Conallen (1999), the extension for multimedia applications in Sauer and Engels (1999) and the UML extension of Baumeister, Koch and Mandel (1999). The first approach is based on the Rational Unified Process – RUP – (Kruchten, 1998) and focuses particularly on the architecture of Web applications. The second extends UML sequence diagrams to model multimedia processes and the third one provides modeling elements for the navigational and presentational design that we have used in our method.
The methodology presented in this chapter differs from the other Web design methods in that it provides guidelines about the activities to be performed systematically by the designer for modeling the navigation space, structure and presentation. The advantage of such guidelines is that they provide a basis for developing case tools for the semi-automatic generation of the Web navigational structure and of presentation templates.
For further details on design and development methods for Web systems see the comparative study of Web development methods in Koch (1999).

FROM REQUIREMENTS TO A CONCEPTUAL MODEL

The design of Web applications builds on the requirements specification, just like the design of software applications in general. Following the Unified Process (Jacobson, Booch & Rumbaugh, 1999) we propose use cases for capturing the requirements. They provide a user-centered technique that forces developers to define who are the users (actors) of the application and offer an intuitive way to represent the functionality that an application has to fulfill for each actor.
The conceptual design of the domain is based on these use cases and includes the objects involved in the typical activities users will perform with the application. The conceptual design aims to build a domain model trying to take into account as little as possible of the navigation paths, presentation and interaction aspects. These aspects are postponed to the navigational and presentational steps of the design.

Modeling Elements

The main modeling elements used in the conceptual model are: *class* and *association*. They are represented graphically by the UML notation (Booch, Rumbaugh & Jacobson, 1999). If the conceptual model consists of many classes, it is recommended to group them using the UML *package* modeling element.
Classes defined in this step are used during navigational design to derive nodes of the Web structure. Associations will be used to derive links.

Example

As a running example to illustrate the design process, through the whole chapter we use the Web Site of a service company. This Web site offers information about the company itself, the employees and their relationships to projects, customers and departments. We restrict ourselves in the example to these concepts although many other aspects should be included in an incremental and iterative process, such as information about products, documents, events, press releases and job offers. Fig. 1 shows a use case model for the project administration, which is part of the use case model of the company's Web application. The company's, department's and employee's administration can be modeled in a similar way.

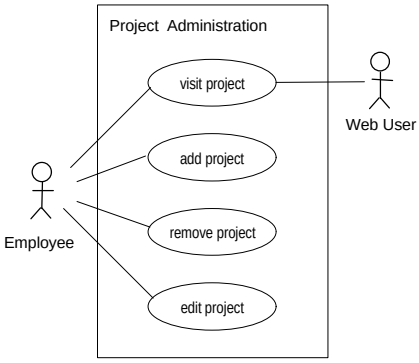

Fig. 1. Use Cases for the Project Administration

The conceptual model for the Web site of the service company is shown in Fig. 2.

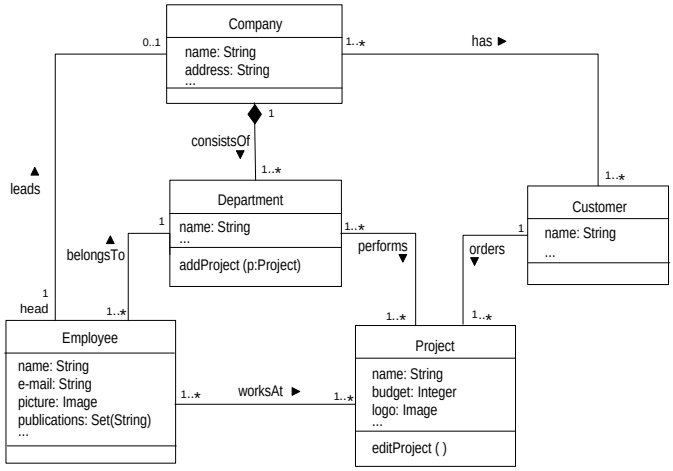

Fig. 2. Conceptual Model

Method

For each use case identified during the requirement analysis a detailed description is provided in terms of (primary and secondary) scenarios, for instance, following the guidelines of Schneider and Winters

(1998). Thus, for Web applications particular emphasis is placed on the information exchanged between the user and the system. The use case descriptions serve as an input for modeling the content of the application.

Well-known object-oriented modeling activities are performed to obtain a conceptual model, such as:

1. finding classes,
2. specifying the most relevant attributes and operations,
3. determining associations between classes,
4. defining inheritance hierarchies,
5. finding dependencies, and
6. defining constraints.

The result of these activities is a UML class model of the problem domain.

FROM A CONCEPTUAL MODEL TO A NAVIGATION SPACE MODEL

In this section we present guidelines to construct a navigation space model from a conceptual model. The navigation space model specifies, *which* objects can be visited by navigation through the Web application. *How* these objects are reached is defined by the navigational structure model that is constructed in the next section. In the process of building the navigation space model the developer takes crucial design decisions, such as which view of the conceptual model is needed for the application and what navigation paths are required to ensure the application's functionality. The decisions of the designer are based on the conceptual model, use case model and the navigation requirements that the application must satisfy.

Modeling Elements

For the construction of the navigation space model two modeling elements are used: navigational classes and navigation associations which express direct navigability.

- *Navigational Class*

A navigational class models a class whose instances are visited by the user during navigation. Navigational classes will be given the same name as conceptual classes. For their representation we use the UML stereotype «navigational class» which is shown in Fig. 3.

- *Direct Navigability*

Associations in the navigation space model are interpreted as representing direct navigability from the source navigation class to the target navigation class. Hence their semantics is different from the associations used in the conceptual model. To determine the directions of the navigation the associations of this model are directed (possibly bi-directed). This is shown by an arrow that is attached to one or both ends of the association. Moreover, each directed end of an association is named with a role name and is equipped with an explicit multiplicity. If no explicit role name is given, the following convention is used: if the multiplicity is less than or equal to one, the target class name is used as the role name; if

the multiplicity is greater than one, the plural form of the target class name is used. In the following diagrams all associations with the exception of composition are implicitly assumed to be stereotyped by «direct navigability».

Example

The navigation space model built with the navigational classes and navigability associations is graphically represented by a UML class diagram. Fig. 3 shows the navigation space model for the Web Site of our service company.

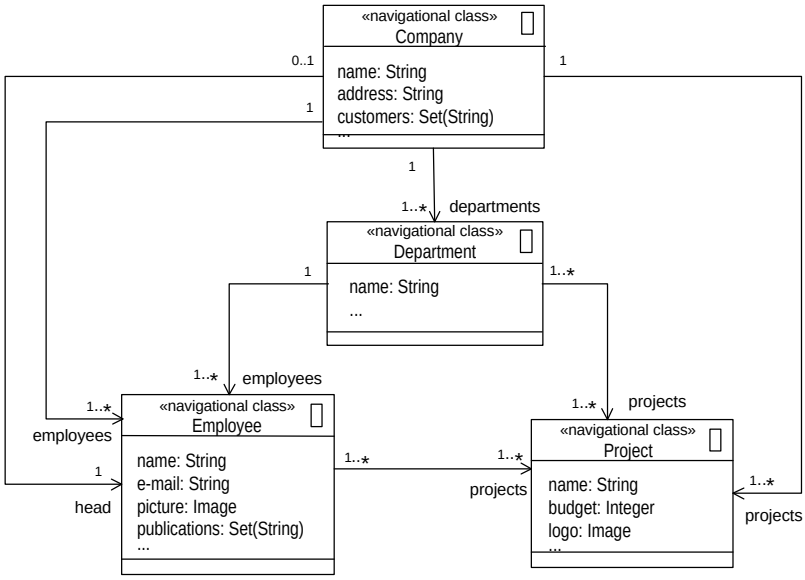

Fig. 3. Navigation Space Model

The Method

Although there is obviously no way to automate the construction of the navigation space model, there are several guidelines that can be followed by the developer:

1. Classes of the conceptual model that are relevant for the navigation are included as navigational classes in the navigation space model (i.e. navigational classes can be mapped to conceptual classes). If a conceptual class is not a visiting target in the use case model, it is irrelevant in the navigational process and therefore is omitted in the navigation space model (like the class Customer in our example).

2. Required information on the omitted classes can still be kept as attributes of other classes in the navigation space model (e.g. the newly introduced attribute customers of the navigational class Company). All other attributes of navigational classes map directly to attributes of the corresponding conceptual class. Conversely, attributes of the conceptual classes that are considered to be irrelevant for the presentation are excluded in the navigation space model.
3. Associations of the conceptual model are kept in the navigational model. Often additional associations are added for direct navigation to avoid navigation paths of length greater than one. Examples are the newly introduced navigation associations between Company and Employee and between Company and Project. Scenarios described by the use case model give the input for the choice of direct navigations.

FROM A NAVIGATION SPACE MODEL TO A NAVIGATIONAL STRUCTURE MODEL

The navigation space model tells us which objects can be visited by direct navigation from other objects. In this section we proceed by describing how the navigation can be performed using access elements like indexes, guided tours, queries and menus. Technically, the navigation paths together with the access elements are presented by a navigational structure model which can be systematically constructed from the navigation space model in two steps: First, we enhance the navigation space model by indexes, guided tours and queries. Then we can directly derive menus which represent possible choices for navigation.

Defining Indexes, Guided Tours and Queries

Indexes, guided tours and queries – so called access primitives – are additional navigational nodes required to access instances of navigational classes. Another access primitive is menu which is treated separately in the next subsection.

Modeling Elements

For describing indexes, guided tours and queries we use the following modeling elements. Their stereotypes and associated icons originate from Baumeister et al. (1999).

- *Index*

An index is modeled by a composite object which contains an arbitrary number of index items. Each index item is in turn an object which has a name and owns a link to an instance of a navigational class. Any index is a member of some index class which is stereotyped by «index» with a corresponding icon. An index class must be built to conform to the composition structure of classes shown in Fig. 4. Hence the stereotype «index» is a restrictive stereotype in the sense of Berner, Glinz and Joos (1999). In practice, we will always use the shorthand notation shown in Fig. 5. Note that in the short form the

association between MyIndex and MyNavigationalClass is derived from the index composition and the association between MyIndexItem and MyNavigationalClass.

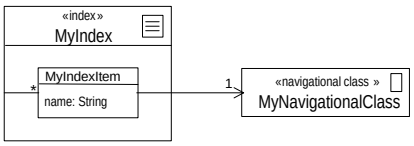

Fig. 4. Index Class

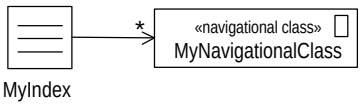

Fig. 5. Shorthand Notation for Index Class

- Gu*ided Tour*.

A guided tour is modeled by an object which provides sequential access to the instances of a navigational class. For classes which contain guided tour objects we use the stereotype «guidedTour» and its corresponding icon depicted in Fig. 6. As shown in Fig. 6, any guided tour class must be connected to a navigational class by a directed association which has the property {ordered}.

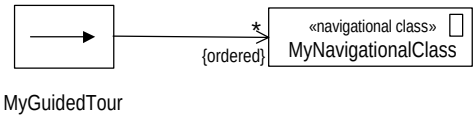

Fig. 6. Guided Tour Class

- *Query*.

A query is modeled by an object which has a query string as an attribute. (This string may be given, for instance, by an OCL select operation.) For query classes we use the stereotype «query» and the icon depicted in Fig. 7. As shown in Fig. 7, any query class is the source of two directed associations related by the constraint {or}. In this way we can model the fact that a query with several result objects must first lead to an index supporting the selection of a particular instance of a navigational class.

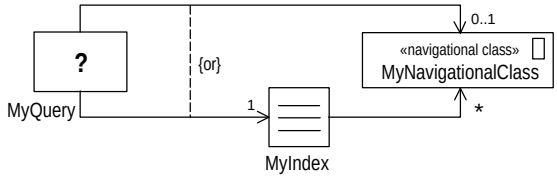

Fig. 7. Query Class

Example

Fig. 8 shows how the navigation space model for the Web Site of our service company can be enhanced by indexes, guided tours and queries. Note that we have included two possible ways to access the employees of a department, by an index and by a query.

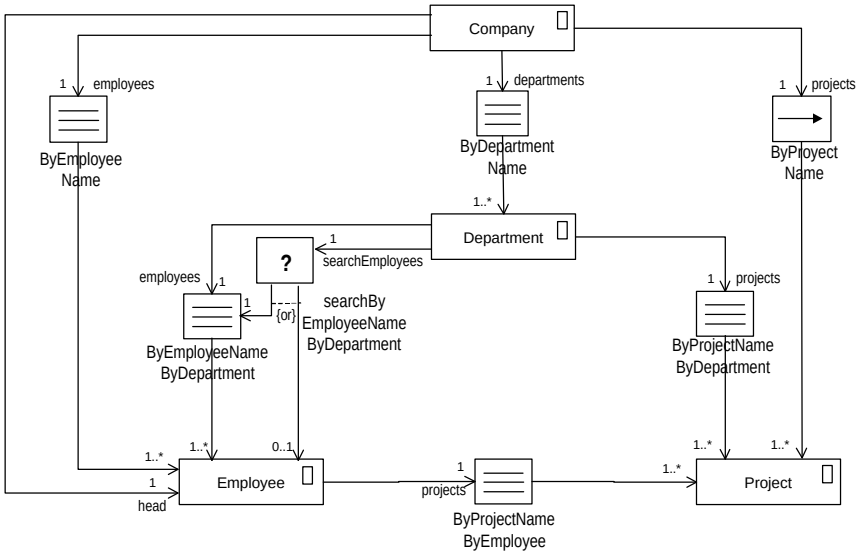

Fig. 8. Navigation Space Model enhanced with Indexes, Guided Tour and Query

The Method

The enhancement of a navigation space model by access elements of type index, guided tour and query follows certain rules which can be summarized as follows:

1. Consider only those associations of the navigation space model which have multiplicity greater than one at the directed association end.
2. For each association of this kind, choose one or more access elements to realize the navigation.
3. Enhance the navigation space model correspondingly. It is therefore important that the role names of the navigation in the navigation space model are now moved to the access elements (compare Fig. 3 and Fig. 8).

In step 2 it is the task of the designer to choose appropriate access elements. However, it is important to note that it is also possible to fully automate this step by making as a default design decision the choice of an index according to a selected key attribute of the target navigational class.

Defining Menus

In this step, access primitives of type menu are added to the navigational structure model.

Modeling Elements

The modeling element menu is a stereotyped class that is defined as follows:

- *Menu*

A menu is modeled by a composite object which contains a fixed number of menu items. Each menu item has a constant name and owns a link either to an instance of a navigational class or to an access element. Any menu is an instance of some menu class which is stereotyped by «menu» with a corresponding icon. A menu class must be built to conform to the composition structure of classes shown in Fig. 9. Hence the stereotype «menu» is again a restrictive stereotype according to the classification of stereotypes given in Berner et al. (1999).
Since menu items are assumed to have fixed names the property {frozen} is attached to each name attribute in a menu item class. Nevertheless, the same menu item class may have different instances since there may be menu items with the same name but linked to different objects. To provide a convenient notation for menu classes in navigational structure models we will use in the following the shorthand notation shown in Fig. 10.

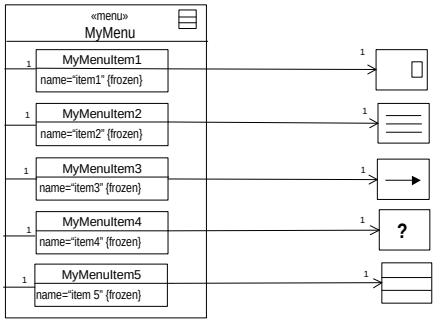

Fig. 9. Menu Class

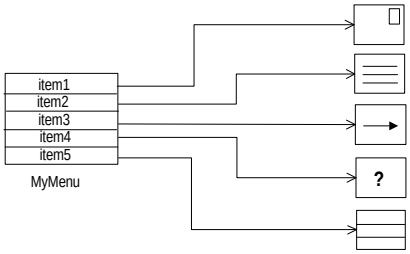

Fig. 10. Shorthand for Menu Class

Example

Fig. 11 shows how the navigational structure model of the previous section is enriched by menus where each menu class is associated by a composition association to a navigational class. Note that the role names occurring in the previous model are now names of corresponding menu items.

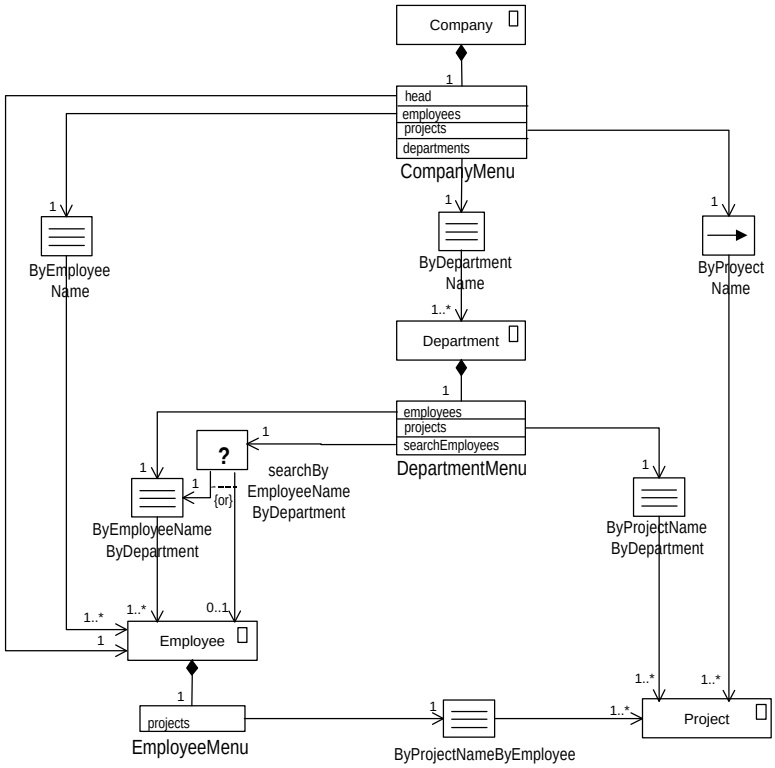

Fig. 11. Navigational Structure Model

The Method

The enhancement of a navigation space model by access elements of type menu follows certain rules which can be summarized as follows:

1. Consider those associations which have as their source a navigational class.

2. Associate to each navigational class which has (in the previous model) at least one outgoing association a corresponding menu class. The association between a navigational class and its corresponding menu class is a composition.
3. Introduce for each role which occurs in the previous model at the end of a directed association a corresponding menu item. By default, the role name is used as the constant name of the menu item.
4. Any association of the previous model which has as its source a navigational class becomes now an association of the corresponding menu item introduced in step 3.

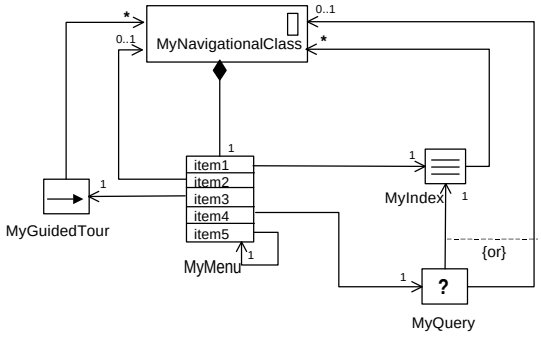

Fig. 12. Navigational Structure Pattern

Navigational Structure Pattern

As a result of our method we obtain a comprehensive navigational structure model of a Web application. It is guaranteed by our method that this model is built to conform to the pattern shown in Fig. 12.

FROM A NAVIGATIONAL STRUCTURE MODEL TO A STATIC PRESENTATION MODEL

The navigational structure model shows how to navigate through the navigation space by using the access elements defined in the previous section. In the next step of our methodology we describe how the information within the navigation space and the access structures are presented to the user. This is done by constructing a presentation model which shows an abstract interface design in a similar way to a user interface sketch. The presentation model focuses on the structural organization of the presentation and not on the physical appearance in terms of special formats, colors, etc. Such decisions are left to the implementation phase which is beyond the scope of this paper. However, the layout of modeling elements in the presentation model may provide hints, for example, about the position and the size of these elements relative to each other.

The following subsections show how a presentation model is derived from the navigation structure model. There are various different types of presentations that can be constructed. Two common alternatives are presented here: a menu-based presentation and a map-based presentation. The last one is also known as tree-structured technique and supports the visualization of the (total or partial) navigation space. It reduces the problem of the "lost in the hyperspace" syndrome. Sometimes both presentation types are combined in one application.

Modeling Elements

For constructing a presentation model one has to decide which presentation elements will be used for the presentation of the instances of navigational classes and which for the presentation of the access elements. For this purpose we use several presentation modeling elements (with corresponding stereotypes):

- *Frameset*

A frameset is a container of presentational objects or other framesets. It is an instance of a frameset class stereotyped by «frameset» with a corresponding icon. (The same stereotype and a similar icon is also used in Conallen (1999). A frameset must be built to conform to the composition structure shown in Fig. 13.

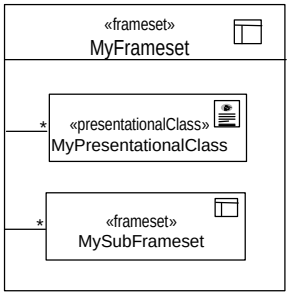

Fig. 13. Frameset

- *Presentational class*

A presentational class models the presentation of a navigational class or an access primitive, such as an index, a guided tour, query or menu. Instances of a presentational class are containers which comprise modeling elements like texts, images, video sequences, audio sequences, anchors, collections (i.e. lists of texts, images, etc.), anchored collections (i.e. lists of anchors), etc. It is stereotyped by «presentational class» with a corresponding icon and follows the composite rules depicted in Fig. 14.

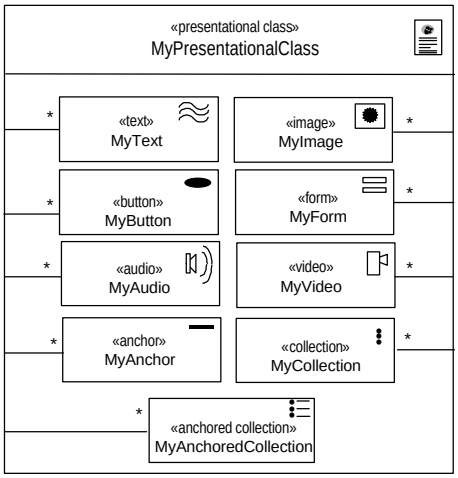

Fig. 14. Presentational Class

- *Text*

A text is a sequence of characters.

- *Anchor*

An anchor is a clickable piece of text which is the starting point of a navigation establishing the relationship to other nodes.

- *Button*

A button is a clickable area which has an associated action. Examples of actions are playVideo, displayImage, stopAudio and runApplet. Note that buttons can also be triggers of navigation.

- *Image, audio* and *video*

Image, audio and video are multimedia objects. An image can be displayed; audio and video can be started, stopped, rewinded and forwarded.

- *Form*

A form is used to request information from the user who supplies information in one or more input fields or selects options from a browser or checkbox.

- *Collection and anchored collection*

Collection and anchored collection are model elements, such as lists of text elements and listsof anchors introduced to provide a convenient representation of composites. It is not specified whether the collection will be laid out horizontally or vertically.

The stereotypes for text, form, button, image, audio, video, anchor, collection and anchored collection as depicted in Fig. 14. Fig. 15 shows how these modeling elements can be used to construct a template for the presentation of employees.

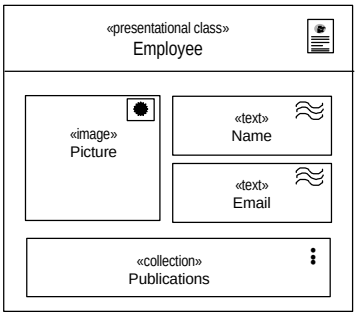

Fig. 15. Presentational Class for Employee

Example for a Menu-based Presentation

Fig. 16 to 19 show part of a presentation model for our sample application. In this example we use framesets to partition the presentation into frames, where the left frame shows always the main menu and the right frame shows the actual content. How this presentation model can be systematically derived from the navigational structure model is explained in Section 5.3. For the moment, it is sufficient to point out that Fig. 16 shows the presentation of the company, Fig. 17 shows the presentation of the head of the company (after having selected Head), Fig. 18 shows the presentation of the department index by

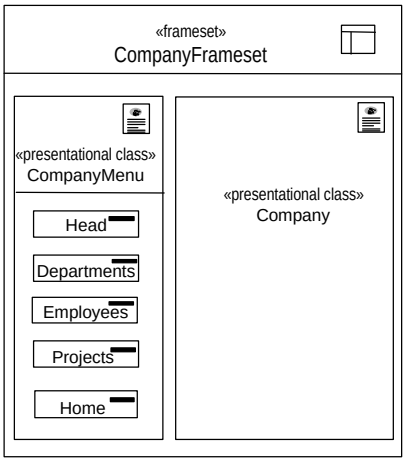

Fig. 16. Frameset for Company

means of a list of anchors (after having selected Departments) and Fig. 19 shows how a selected

department is presented. Thus we have not detailed the presentational class for department which can be done in a similar way to employee (in Fig. 15).

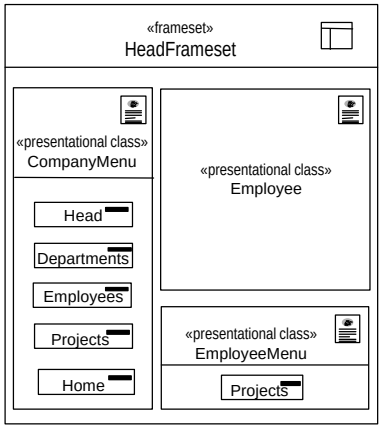

Fig. 17. Frameset for Head

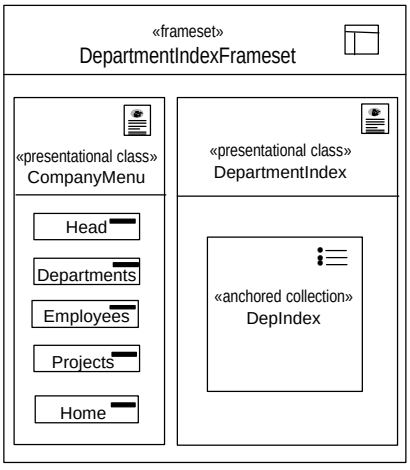

Fig. 18. Frameset for Department Index

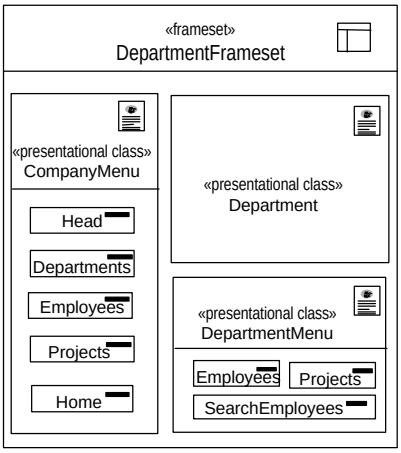

Fig. 19. Frameset for Department

The Method for a Menu-based Presentation

The presentation model consists of a set of presentational classes (Web page). The following rules can be used as a guide:

1. Construct a presentation for each navigational class occurring in the navigational structure model. It has to provide a template for presenting the instances of the class taking into account the given attributes. Stereotyped classes, such as «text», «image», «audio», «video» are used for attributes of primitives types and «collections» are used for lists, etc. For instance, Fig. 15 shows the presentation constructed for the navigational class Employee.
2. Choose one navigational class as a root for navigation and construct a presentational class for the menu of this navigational class (called main menu). In our example we select the class Company with the main menu CompanyMenu shown in Fig. 16. We add an anchor Home, such that it is always possible to go back to the root.
3. Construct a presentational class for each index and menu occurring in the navigational structure model. For the presentation of an index or a menu class we use modeling elements with stereotypes «anchored collection» or «anchor». Examples for menus and indexes are the EmployeeMenu, the DepartmentIndex and the DepartmentMenu included in Fig. 17, Fig. 18 and Fig. 19, respectively.
4. Construct a presentational class for each query and each guided tour. Use forms for representing queries and for guided tours introduce additional menu items ("next", "prev") which allow navigation to the next and previous object within the guided tour.
5. Combine each presentational class (constructed in 1) with the presentational class of its menu (constructed in 3) in a frameset. Eventually, add the main menu to the frameset. Such combinations can be seen in Fig. 17 and Fig. 19.

Example for a Map-based Presentation

Fig. 20 to 23 show a map-based presentation model for our sample application which is an alternative to Section 5.2. In this case the left frame shows the actual navigation tree and the right frame shows the corresponding content. How this presentation model can be systematically derived from the navigational

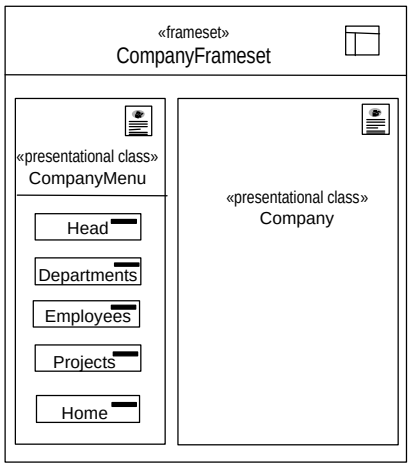

Fig. 20. Frameset for Company

structure model is explained in Section 5.5.

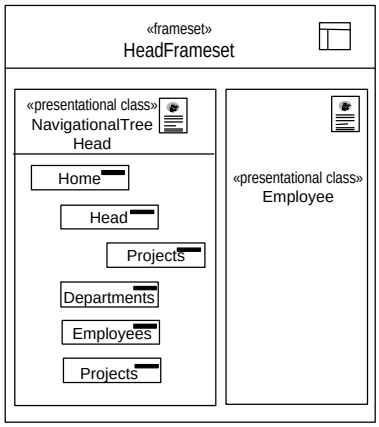

Fig. 21. Frameset for Head

Fig. 20 shows the presentation of the company, Fig. 21 shows the presentation of the head of the

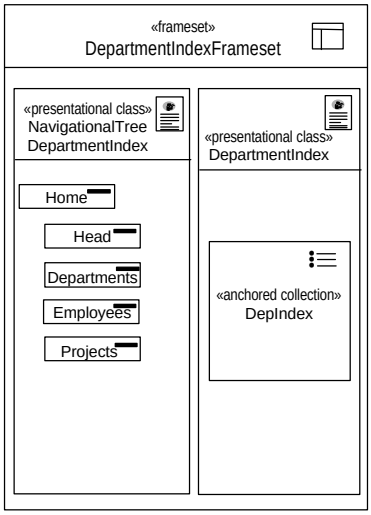

Fig. 22. Frameset for Department Index

company (after having selected Head), Fig. 22 shows the presentation of the department index by a list of anchors (after having selected Departments) and Fig. 23 shows how a selected department (for instance the i-th department Dep_i) is presented.

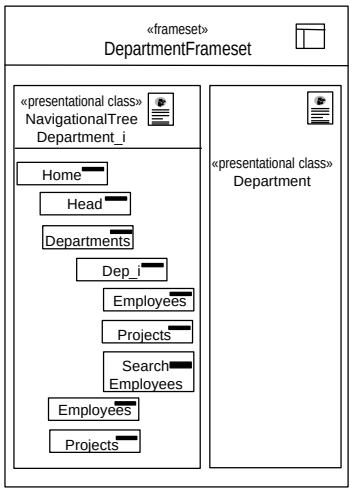

Fig.23. Frameset for Department

The Method for a Map-based Presentation

The map-based presentation method is again based on the use of framesets which allow us to visualize the navigation structure. The idea is therefore to always divide a presentation into two basic parts: one part provides a presentation of the navigation tree (showing the user's actual navigation path and hence the context of navigation) and the other part shows the corresponding content.
On this basis we define the following procedure for deriving a presentation model from a navigational structure model in an entirely systematic way.

1. Construct a presentation for each navigational class and model each index class occurring in the navigational structure in a similar way as in Section 5.3.
2. Choose one navigational class as a root for navigation. In our example we select the class Company.
3. For each navigational class and for each index class consider all possible paths (in the navigational structure model) from the root class to the actual class. For each path construct a presentation of the corresponding navigation tree.
   Let us explain this step in more detail by considering our example. We start with the class Company. The corresponding navigation tree is represented by the presentational class NavigationTreeCompany in the left frame of Fig. 20. Since Company is the root of the navigation the corresponding tree is trivial and shows only the menu associated with Company.
   The presentational class NavigationTreeHead in Fig. 21 shows the navigation tree if one moves to the head of the company. Note that the root of this tree is represented by the anchor Home for going back to the company. The anchor Projects is inserted at depth 2 of the tree to present the menu associated to employees (remember that the head is indeed an employee).

The presentational class NavigationTreeDepartmentIndex in Fig. 22 shows the navigation tree if one moves from the company to the department index and the presentational class NavigationTreeDepartment_i in Fig. 23 shows the navigation tree if one navigates further to a particular department (for instance, the i-th department Dep_i). Note that in this case the anchors Employees, Projects and Search Employees are inserted at depth 3 of the tree for presenting the menu associated with a department.

4. Combine the results of step 1 and step 3 according to framesets. Any frameset has two parts where the right frame contains the presentation of the navigational class or index class (constructed in step 1) and the left frame represents the navigation tree (constructed in step 3) corresponding to one possible navigation to this class.
   In our example, this leads to the framesets shown in Fig. 20 to 23 which of course have to be completed by taking into account all other possible navigations shown in the navigational structure model in Fig. 11. In particular, Fig. 11 contains also guided tours and queries whose presentation is not detailed here. The idea is to present a guided tour simply by two additional anchors Next and Prev (with an obvious meaning) which extend the menu of the corresponding navigational class. As a straightforward presentation of queries one will usually choose forms.

In step 3 we must ensure that there is only a finite set of navigation paths from the root class to each navigational or index class. For this purpose we assume that the given navigational structure model has no cycles, i.e. forms a directed acyclic graph. This is not a proper restriction since in any case we provide a presentation of the navigation tree that allows us to move backwards. Concerning the presentation of a navigation tree, it is obvious that in practice the depth of the tree must be limited. For a convenient representation of such trees one may also use several frames, for instance a top frame and a left frame. In this case the left frame in Fig. 23 would be split into a top frame which contains the main menu and into a left frame presenting the subtree with the anchor Dep_i as a root.
Let us note that there is also a variant of the above procedure which treats the presentation of indexes differently. With this variant the department index would be included in the navigation tree on the lefthand side while the right frame could include, for instance, some additional general information on all departments. This variant is shown in Fig. 24.

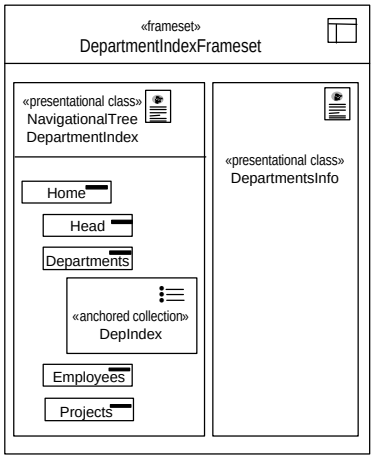

Fig. 24. Department Index Frameset (Variant)

DYNAMIC PRESENTATION MODEL

In addition to the static presentation model we model the dynamic aspects of the presentation by a window flow model that describes the behavior of the presentational objects, i.e. the changes on the user interface when the user interacts with the system. The construction of a window flow model is mainly recommended when a multiple-window technique is chosen. It specifies when windows are open, closed and when they coexist.
Control flow between windows can be represented by interaction models showing which windows are open and which frameset is displayed in each window at a certain moment. For this purpose we use UML sequence diagrams.

Modeling Elements

The following modeling element *window* is used for describing user interactions with the user interface of the Web application:

- *Window*

A window is the area of the user interface where framesets or presentational objects are displayed. A window can be moved, resized and reduced to an icon. It includes at least two buttons, one to transform the window into an icon and one to close the window. Fig. 25 depicts a stereotyped class for windows.

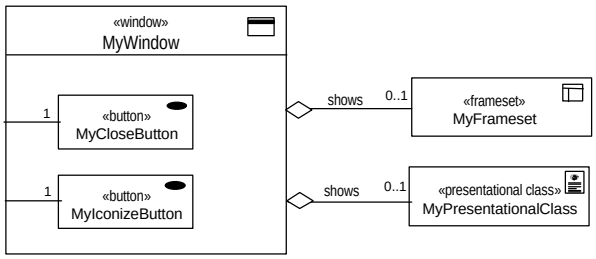

Fig. 25: Window

Example

Fig. 26 shows part of a window flow model for the sample application corresponding to the search of employees by department. The window flow model describes the dynamics of the map-based presentation of Section 5.4. It shows an abstract representation since the eventual implementation will include more objects, in particular control objects that collaborate in this interactive process (Conallen, 1999).

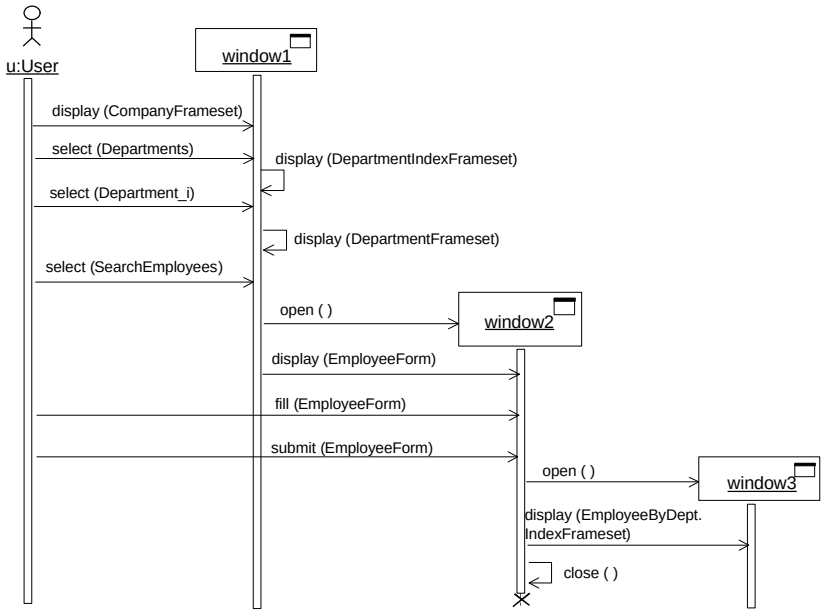

Fig. 26: Window Flow Model

The Method

The steps to build a window flow model cannot be automated as the developer has to decide how many windows are used, when and which additional windows are opened and closed. The following is a set of guidelines to assist the developer in the modeling of the window flow model based on the navigational structure model and the static presentation model:

1. Set the context for the interaction model, i.e. define which navigation path of the navigation structure diagram will be modeled . A navigation path is always related to a use case. In Fig. 26 it is the use case search employee by department that is part of the employee's administration.
2. Decide how many windows are needed in this context. Three windows are used in our example. Represent the user and these window objects in the horizontal dimension.
3. Specify a display message for each presentational object and frameset that should be presented to the user (in a window). The parameter of the display message is the corresponding frameset (described in Section 5).
4. Include a select message for each user action that selects an anchor or a button. The anchor or button names are the parameters of the message.
5. Specify a fill and a submit message for each user action that consist of supplying data in a query form. This form is the parameter of the message.
6. Include a message for each open and each close of a window.
7. Use "balking" to specify the period of time that a window is visible.

UML sequence diagrams are used to represent the window control flow. Note that the representation does not include additional classes needed in the implementation to facilitate the communication between windows.

FUTURE TRENDS

Let us return to the first sentence of this chapter – Web engineering is a new and still evolving discipline. The methodology presented in this chapter is part of this Web engineering discipline and is not expected to be an exception.

There are many open issues to be addressed and integrated, such as database and distributed systems aspects, changing technologies in the Web field or new versions of the UML. Another aspect that will play an important role in the future development of hypermedia systems is the personalization of applications, i.e. adaptation of the presentation or navigation according to the user interests, knowledge or preferences. Adaptation can be based on the information provided by agents that observe the users' behavior. A future methodology for Web design has also to scope with the design of multi-modal interfaces including speech for example. Synchronization problems also have to be solved in these kinds of flexible Web applications.

Tools for UML are developing fast, but there is an enormous scope for improvement; indeed there is widespread dissatisfaction with the state of the art. There are good drawing tools, but these need to increase their capabilities to provide automatic verification of models, to support the use of patterns, and to allow for constraint specification with OCL.

There is still much work needed to improve the Web engineering discipline. This will be done mostly in little steps through the continuous adjustment of methodologies, techniques, notations and tools.

CONCLUSIONS

In this chapter we have presented a methodology for the design of Web applications that uses a UML profile for the Web domain. The methodology consists of a notation and a method that extends previous approaches of Baumeister et al. (1999) and Hennicker & Koch (2000). The deliverables of our method are models represented by UML diagrams. Some of the modeling elements occurring in such diagrams are defined by stereotypes using the UML extension mechanism. The definition of many new stereotypes causes extra effort to read the diagrams, but once one gets use to them the diagrams are more meaningful in terms of Web design. The advantages of the methodology are the use of the UML, the consideration of specific Web aspects in design of Web applications through the definition of specialized modeling elements and the creation of the tailored models to express navigation and presentation.

The methodology describes how to build:

- the navigation space model based on the conceptual model,
- the navigational structure model from the navigation space model, and
- the static and dynamic presentation model from the navigational structure model.

The strength of this approach is that for each model a detailed list of construction steps is provided. We have therefore tried to identify as many steps as possible that can be performed automatically, for instance, when constructing the navigational structure model from the navigation space model. In addition, the method describes how templates for the Web application can be systematically generated from the navigational structure model. However, there are still several steps where decisions of the designers are essential. This concerns, in particular, the construction of the navigation space model based on the conceptual model (2.9).
The design steps presented here are part of a development process (Koch, 2000) based on the Unified Process (Jacobson et al., 1999) that covers the whole lifecycle of Web applications. The methodology still requires validation and testing for a wide spectrum of Web applications. Moreover it has to be extended to model the dynamic and database aspects related to Web applications, in particular for e-commerce applications. A further next step will be the construction of a case tool that supports our methodology.

ACKNOWLEDGEMENT

We would like to thank Luis Mandel and Hubert Baumeister for common work related to the UML extension. We wish to thank Martin Wirsing for the various discussions, reviews and comments. We thank also Andy Schürr and other participants of the GROOM 2000 Workshop for helpful suggestions.

REFERENCES

Baumeister, H., Koch, N., & Mandel L. (1999). *Towards a UML extension for hypermedia design. In Proceedings «UML»'99, France, R., Rumpe, B. (Eds)*, LNCS, Vol. 1723. Springer-Verlag, 614-629.

Berner S., Glinz M., & Joos S. (1999). *A classification of stereotypes for object-oriented modeling languages. In Proceedings «UML»'99*, France, R., Rumpe, B. (Eds), LNCS, Vol. 1723. Springer-Verlag, 249-264.

Booch G., Rumbaugh J., & Jacobson I. (1999). *The Unified Modeling Language: A User Guide.* Addison Wesley.
Conallen J. (1999). Building Web Applications with UML. Addison-Wesley.

Hennicker R., & Koch N.(2000). A UML-based methodology for hypermedia design. *In Proceedings UML2000*, LNCS, Springer-Verlag.

Isakowitz T., Stohr E., & Balasubramanian P. (1995). A methodology for the design of structured hypermedia applications. *Communications of the ACM*, 8(38), 34-44.

Jacobson I., Booch G., & Rumbaugh J. (1999). *The Unified Software Development Process.* Addison Wesley (1999).

Koch N. (1999). A comparative study of methods for hypermedia development. Technical Report 9901, Ludwig-Maximilians-University Munich.

Koch N. (2000). Hypermedia systems development based on the Unified Process. Technical Report 0003, Ludwig-Maximilians-University Munich.

Kruchten P. (1998). *The Rational Unified Process: An Introduction.* Addison Wesley.

Lowe D., & Hall W.(1999). *Hypermedia & the Web: An engineering approach.* John Wiley & Sons.

Nanard J., & Nanard M. (1995). Hypertext design environments and the hypertext design process. *Communication of the ACM,* August 1995, Vol 38 (8), 49-56.

Sauer S., & Engels G. (1999). *Extending UML for modeling of multimedia applications.* In Proceedings of the IEEE Symposium on Visual Languages – VL´99, IEEE Computer Society, 80-87.

Schneider G., & Winters J. (1998). *Applying Use Cases: A Practical Guide.* Addison-Wesley Object Technology Series.

Schwabe D., & Rossi G. (1998). *Developing hypermedia applications using OOHDM.* In Proceedings of Workshop on Hypermedia Development Process, Methods and Models, Hypertext´98.