A UML-based Methodology for Hypermedia Design

Rolf Hennicker[1], Nora Koch [1,2]

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

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

Abstract. We propose a methodology for hypermedia design which is based on a UML profile for the hypermedia 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 next 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 presentation model is constructed that can be directly implemented by HTML frames. The different models of the design process are represented by using a hypermedia extension of UML.

The strength of the presented methodology is given by the fact that most steps can be performed in an automatic way thus providing the basis for a generation mechanism for hypermedia design.

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

1 Introduction

Hypermedia development is a new and still evolving discipline. The process of learning how to develop large hypermedia applications has just begun. Hypermedia applications for the Web or CD-ROM are mostly the result of an implementation ad hoc, growing usually from small to large applications and becoming very soon difficult to maintain. Some guidelines and tools are beginning to appear assisting developers of hypermedia applications. But these current practices often fail due to inappropriate techniques, processes and methodologies.

The development of hypermedia systems differs from the developing process of traditional software in several dimensions. People with different skills are involved in the process, such as authors, layout designers, programmers, multimedia experts and marketing specialists. The role of the user is 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 hypermedia applications increments the complexity and risk of "lost in the hyperspace". Web and hypermedia

engineering has to take into account aesthetic and cognitive aspects as well that traditional software engineering environments do not support [11]. The development process tends to be more fine grained, more incremental and iterative. Maintenance is a significant part of the lifecycle of hypermedia applications in contrast to the role played in traditional systems. In addition, security is a concern of most Web applications.

If we restrict ourselves to the design steps, the main differences we can observe between design of traditional and hypermedia software are the heterogeneity of the designer group, the hypertext structure composed of nodes and links, the need of navigational assistance, the multimedia contents and the presentation of this contents e.g. for different browsers. Thus, the design is centered around three main aspects of hypermedia systems. There are the content, navigational structure and presentation. Treating this 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 [6] adapted for hypermedia applications. We propose a design methodology which is based on a UML extension for hypermedia [1]. It consists of three steps that are performed in an iterative design process. The steps are the conceptual, navigational and presentational design. They produce the following artifacts:

- conceptual model
- navigation space model and navigational structure model
- presentation model

The conceptual model is built 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 class diagram.

Based on this conceptual model the navigation space model is constructed. It also is represented as a static class model. A set of guidelines is proposed for modeling the navigation space. A detailed specification of associations, their multiplicity and role names establishes the base for an automatic generation of the 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 hypermedia application showing how navigation is supported by the access structures. Finally, we derive from the navigational structure model and the requirements specification a presentation model based on framesets. This model can be implemented by HTML frames.

Navigational design is a critical step in the design of hypermedia application. Even simple applications with a non-deep hierarchical structure will become complex very soon by the addition of new links. Additional links improve navigability on the one hand but imply on the other hand higher risk to loose orientation. Building a navigation model not only is helpful for the documentation of the application structure, it also allows for a more structured increase of navigability.

The strength of our methodology is given by the fact that most steps can be performed in an automatic way thus providing the basis for a generation mechanism for hypermedia design.

This paper is structured as follows: Section 2 describes the starting points for the modeling process: use cases and a conceptual model. Section 3 gives guidelines to build a navigation space model based on the conceptual model. In Section 4 we present a procedure to derive the navigational structure model from the navigation space model. Section 5 shows how the presentation model is obtained from the navigational structure model. In Section 6 a brief description about related work is given. Finally, Section 7 presents some concluding remarks and an overview of future research.

2 Starting with Use Cases and a Conceptual Model

The design of hypermedia applications builds on the requirements specification in the same way as the design of software applications in general does. Following [6] we propose use cases for capturing the requirements. It is a user-centered technique that forces to define who are the users (actors) of the application and offers an intuitive way to represent the functionality an application has to fulfill for each actor.

As an example to illustrate the design process 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 could be included, such as information about products, documents, events, press releases and job offers. Fig. 1 shows a use case model for the project administration that is part of the use case model of the Web application. The company's, department's and employee's administration can be modeled in a similar way.

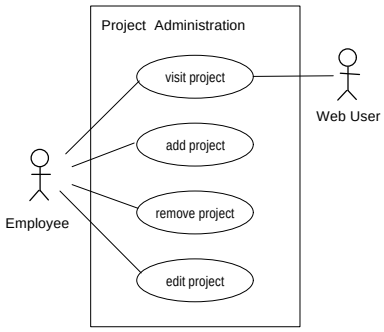

Fig. 1. Use Cases for the Project Administration

The conceptual design of the domain is based on these use cases that include 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. Well-known object-oriented modeling activities are performed during the conceptual modeling, such as:

- to find classes,

- to specify the most relevant attributes and operations,
- to determine associations between classes,
- to define inheritance hierarchies,
- to find dependencies,
- to identify interfaces and
- to define constraints.

The results of these activities is a UML class model of the problem domain. Classes are described through attributes and operations and represented graphically with UML notation [3]. The conceptual model for the Web site of a service company is shown in Fig. 2. Classes and associations defined in this step are used during navigational design to derive nodes of the hypermedia structure. Associations will be used to derive links.

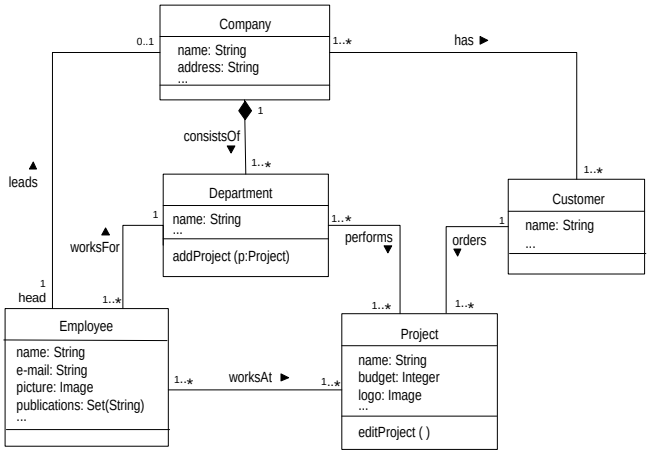

Fig. 2. Conceptual Model

3 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 through navigation in the hypermedia 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 design decisions that are crucial, 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.

3.1 Modeling Elements

For the construction of the navigation space model two modeling elements are used: navigational classes and navigation associations, which express direct navigability. They are the pendant to node and link in the hypermedia terminology.

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 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 bidirected). 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 or equal than one the target class name is used as 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 exception of composition are implicitly assumed to be stereotyped by «direct navigability».

3.2 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.

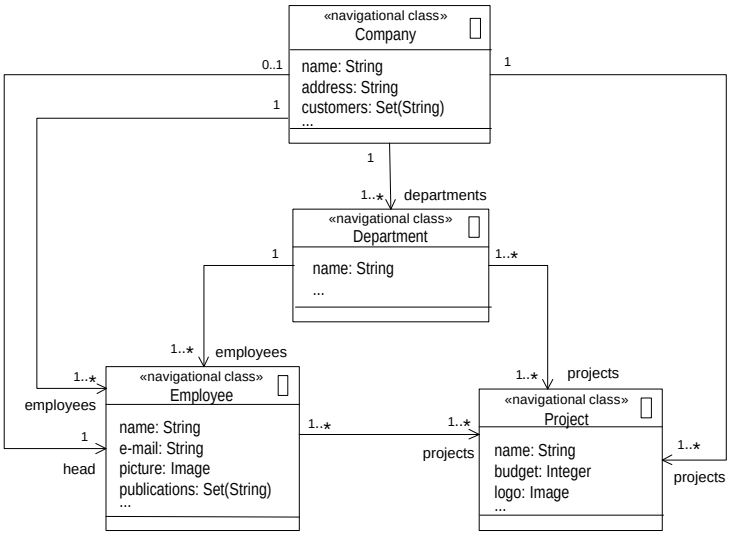

Fig. 3. Navigation Space Model

3.3 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 not relevant for the presentation are excluded in the navigation space model.
3. Often additional associations are added for direct navigation avoiding 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.

4 From a Navigation Space Model to a Navigational Structure Model

The navigation space model tells us which objects can be visited by direct navigations 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.

4.1 Defining Indexes, Guided Tours and Queries

4.1.1 Modeling Elements

For describing indexes, guided tours and queries we use the following modeling elements. Their stereotypes and associated icons stem from [1].

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 member of some index class which is stereotyped by «index» with a corresponding icon. An index class must

be built conform to the composition structure of classes shown in Fig. 4. Hence the stereotype «index» is a restrictive stereotype in the sense of [2]. 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.

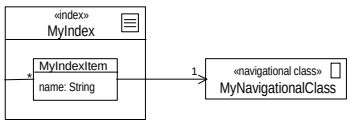

Fig. 4. Index Class

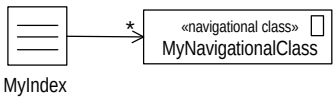

Fig. 5. Shorthand Notation for Index Class

Guided 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}.

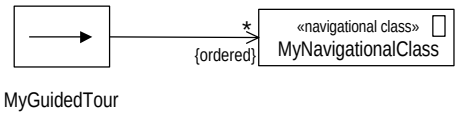

Fig. 6. Guided Tour Class

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 that a query with several result objects must first lead to an index which then allows to select a particular instance of a navigational class.

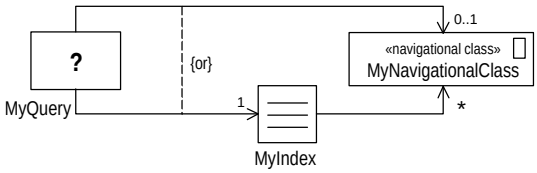

Fig. 7. Query Class

4.1.2 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.

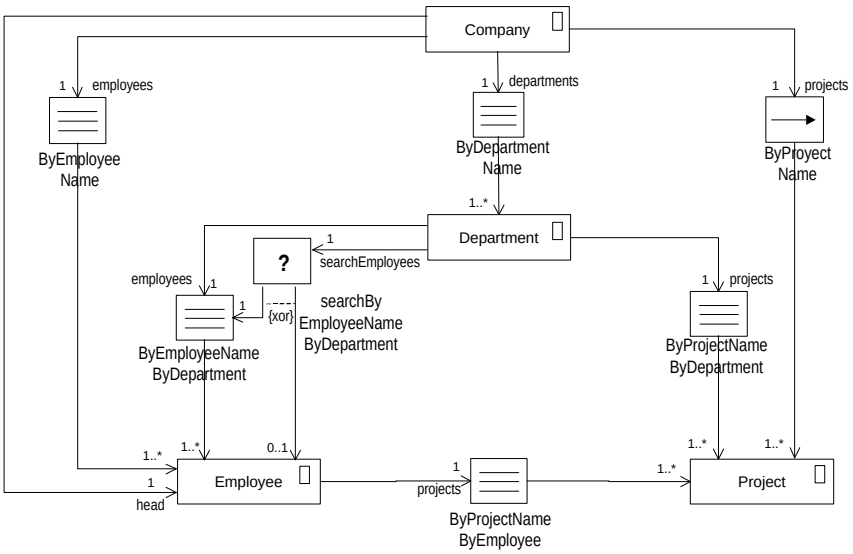

Fig. 8. Navigation Space Model enhanced with Indexes, Guided Tour and Query

4.1.3 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 1 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. Thereby it is important that the roles 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 possible to fully automate also this step by taking as a default design decision always an index according to an attribute with property {key} of the target navigational class.

4.2 Defining Menus

4.2.1 Modeling Elements

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 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 [2].

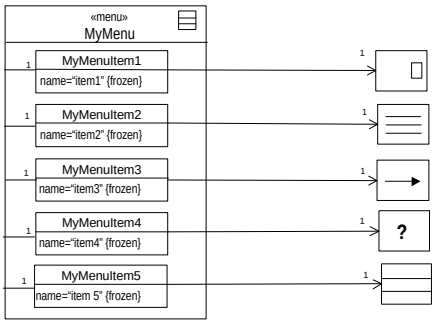

Fig. 9. Menu Class

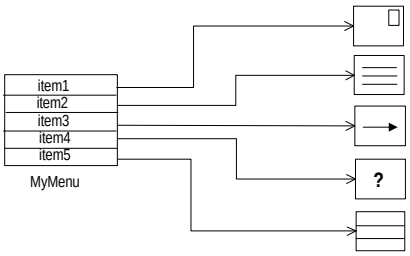

Fig. 10. Shorthand for Menu Class

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. For a convenient notation of menu classes in navigational structure models we will use in the following the shorthand notation shown in Fig. 10.

4.2.2 Example

Fig. 11 shows how the navigational structure model of the previous section is

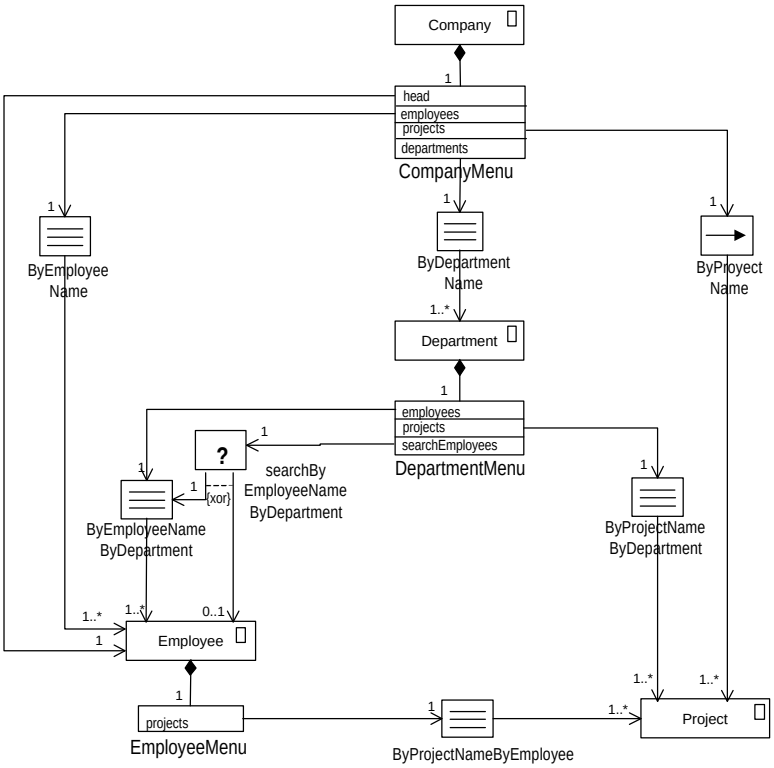

Fig. 11. Navigational Structure Model

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.

4.2.3 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 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 source a navigational class becomes now an association of the corresponding menu item introduced in step 3.

Note that all steps of the above method can be performed in a fully automatic way. As a result we obtain a comprehensive navigational structure model of our application. It is guaranteed by our method that this model is conform to the pattern shown in Fig. 12.

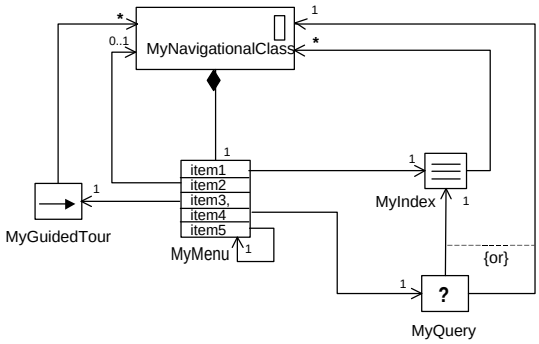

Fig. 12. Design Pattern for Access Structures

5 From a Navigational Structure Model to a 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 similarly 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 construction of user interface prototypes or to the implementation phase which is not in the scope of this paper.

5.1 Modeling Elements

For constructing a presentation model one has to decide which presentational elements will be used for the presentation of the instances of navigational classes and, on the other hand, for the presentation of the access elements. For this purpose we use several presentational modeling elements (with corresponding stereotypes). A top level element for presentation is a frameset which is modeled by a composite object that contains (lower level) presentational objects but may also contain an arbitrary number of nested framesets. A frameset is an instance of a frameset class stereotyped by «frameset» with a corresponding icon. Classes which model framesets must be built conform to the composition structure shown in Fig. 13.

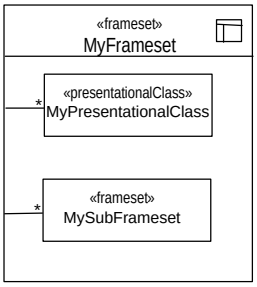

Fig. 13. Frameset

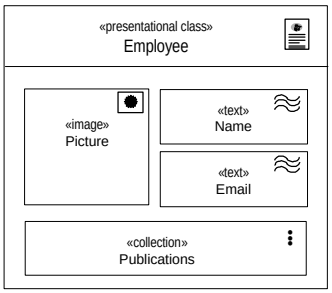

Fig. 14. Presentational Class for the Employee

Presentational objects contained in a frameset are instances of a presentational class which is stereotyped by «presentationalClass» with a corresponding icon. Such objects are containers which comprise basic presentational elements like texts, collections (i.e. lists of texts), images, anchors, anchored collections (i.e. lists of anchors) etc. The basic elements and their stereotypes are defined in [1]. Fig. 14 shows how they can be used for constructing a template for the presentation of employees.

5.2 Example

Fig. 15 to 18 show part of a presentation model for our sample application. In this example we use framesets to partition the presentation into frames, whereby 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 structure model is explained in Section 5.3. For the moment it should be sufficient to point out that Fig. 15 shows the presentation of the company, Fig. 16 shows the presentation of the head of the company (after having selected Head), Fig. 17 shows the presentation of the department index by a list of anchors (after having selected Departments) and Fig. 18 shows how a selected department (for instance the i-

th department Dep_i) is presented. Thereby we have not detailed the presentation class for departments which can be done similarly to the case of employees (see Fig. 14).

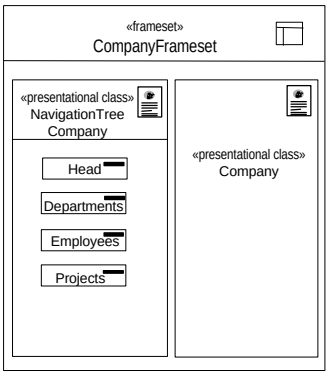

Fig. 15. Frameset for Company

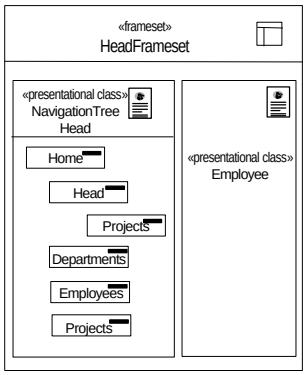

Fig. 16. Frameset for Head

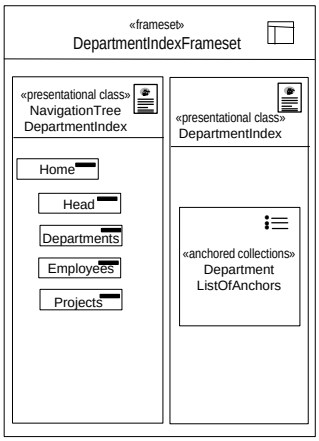

Fig. 17. Frameset for Department Index

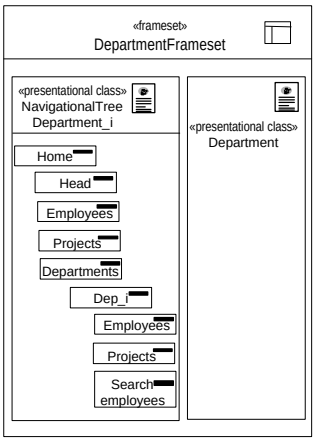

Fig. 18. Frameset for Department

5.3 The Method

Of course there are many possibilities to construct a presentation model for a given navigational structure model. In any case it is essential to define a presentation for each navigational class and one has also to support the navigation structure. The method we propose is based on the use of framesets which allow us to visualize a navigation structure. Thereby the idea is to divide a presentation always 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 for each index class occurring in the navigational structure model. The presentation of a navigational class has to provide a template for presenting the instances of the class which takes into account the given attributes. For instance, Fig. 14 shows a presentation for the navigational class Employee. The presentation of an index class is usually given by a list of anchors. For instance, the presentational class DepartmentIndex in the right frame of Fig. 17 defines a presentation of the index class ByDepartmentName of Fig. 11.
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 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. 15. Since Company is the root of the navigation the corresponding tree is trivial and shows only the menu associated to Company.

   The presentational class NavigationTreeHead in Fig. 16 shows the navigation tree if one moves to the head of the company. Note that the root of this tree is presented 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. 17 shows the navigation tree if one moves from the company to the department index and the presentational class NavigationTreeDepartment_i in Fig. 18 shows the navigation tree if one navigates further to a particular department (for instance, the i-th department Dep_i). Note that then the anchors Employees, Projects and Search Employees are inserted at depth 3 of the tree for presenting the menu associated to a department.
4. Combine the results of step 1 and step 3 accordingly 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. 15 to 18 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 anyway we provide a presentation of the navigation tree which, in particular, 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. 18 would be splitted 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 left side while the right frame could include, for instance, some additional general information on all departments.

6 Related Work

During the last few years many methods for hypermedia and Web design have been proposed; see [10] for an overview. Most of them are not based on the UML, like RMM (Relationship Management Methodology) [5] and OOHDM (Object-Oriented Hypermedia Design Method) [13]. They utilise entity-relationship diagrams, OMT or their own notation and techniques.

Recently, some new approaches propose UML extensions for the hypermedia domain, for instance the development process of Conallen [4], the extension for multimedia applications in [12] and the UML extension of [1]. The first approach is based on the Rational Unified Process (RUP) [9] and focuses particularly on the architecture of Web applications. The second one extends UML sequence diagrams to model multimedia processes. The third one provides modeling elements for the navigational and presentational design that we have used in our methodology.

However, a guideline of activities to be performed systematically by the designer for modeling the navigation space, structure and presentation as in our approach is missing by almost all hypermedia design methods. The advantage of such a guideline is that it can be used by a Case Tool for the automatic generation of the hypermedia navigational structure and of presentation templates.

For further details on design and development methods for hypermedia systems see the comparative study of hypermedia development methods in [7].

7 Conclusions and Future Work

In this paper we have presented a methodology for hypermedia design that uses a UML profile for the hypermedia domain. The strength of our approach is given by the semi-automatic generation of the navigational structure and presentation of the application.

We describe how to build step by step:

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

The design steps presented here are part of a development process [8] based on the Unified Process [6] that covers the whole lifecycle of hypermedia applications.

The objectives of our future work are to include the modeling of the dynamic and database aspects related to hypermedia applications. The methodology will be refined and tested for the design of more complex applications, such as Web applications that require database transactions and user activities not restricted to browsing. A next step will be the construction of a Case-Tool for the semi-automatic development of hypermedia and Web applications based on our methodology.

Acknowledgement

We would like to thank Luis Mandel and Hubert Baumeister for common work related to the UML extension [1]. We thank also Andy Schürr and other participants of the GROOM 2000 Workshop for helpful suggestions and comments.

References

1. Baumeister, H., Koch, N., Mandel L.: Towards a UML extension for hypermedia design. InProceedings «UML»'99, France, R., Rumpe, B. (eds), LNCS, Vol. 1723. Springer-Verlag (1999) 614-629.
2. Berner S., Glinz M., Joos S.: A classification of stereotypes for object-oriented modeling languages. InProceedings «UML»'99, France, R., Rumpe, B. (eds), LNCS, Vol. 1723. Springer-Verlag (1999) 249-264.
3. Booch G., Rumbaugh J., Jacobson I.: The Unified Modeling Language: A User Guide. Addison Wesley (1999).
4. Conallen J.: Building Web Applications with UML. Addison-Wesley (1999).
5. Isakowitz T., Stohr E., Balasubramanian P.: A methodology for the design of structured hypermedia applications. Communications of the ACM, 8(38) (1995) 34-44.
6. Jacobson I., Booch G., Rumbaugh J.: The Unified Software Development Process. Addison Wesley (1999).
7. Koch N.: A comparative study of methods for hypermedia development. Technical Report 9901, Ludwig-Maximilians-University Munich (1999).
8. Koch N.: Hypermedia systems development based on the Unified Process. Technical Report 0003, Ludwig-Maximilians-University Munich (2000).
9. Kruchten P. *The Rational Unified Process: An Introduction.* Addison Wesley (1998).
10. Lowe D., Hall W.: Hypermedia & the Web: An engineering approach. John Wiley & Sons (1999).
11. Nanard J., Nanard M. Hypertext design environments and the hypertext design process. Communication of the ACM, August 1995, Vol 38 (8), (1995) 49-56.
12. Sauer S. and Engels G.: Extending UML for Modeling of Multimedia Applications. In Proceedings of the IEEE Symposium on Visual Languages – VL´99, IEEE Computer Society (1999) 80-87
13. Schwabe D., Rossi G.: Developing hypermedia applications using OOHDM. In Proceedings of Workshop on Hypermedia Development Process, Methods and Models, Hypertext´98 (1998).