Proc. of the ACM First Int. Conference on Web-Age Information Management (WAIM 2000), 21.-23. Juni 2000, Shanghai, China, Springer, LNCS 1846, 2000, pp. 256-270

HMT: Modeling Temporal Aspects in Hypermedia Applications

Günther Specht

Technische Universität München
Institut für Informatik
Orleansstraße 34, D-81667 München
specht@informatik.tu-muenchen.de

Peter Zoller

Bavarian Research Center for
Knowledge-Based Systems (FORWISS)
Orleansstraße 34, D-81667 München
zoller@forwiss.de

Abstract. Since today large hypermedia applications such as huge web sites or extensive CD-presentations cannot be built without a dedicated design phase any more, advanced methodologies for modeling these kinds of applications become more and more important.
In this paper, we present the essence of the Hypermedia Modeling Technique (HMT), which supports the modeling of truly interactive, adaptive and time-based hypermedia applications on arbitrary platforms. HMT offers new design primitives for the specification of temporal dependencies, for modeling search and update interfaces to the underlying data sources, for specifying access restrictions and for defining advanced hyperlinks with a more powerful link concept than their current WWW counterparts. Thus, HMT excels other methods which focus mainly on information presentation issues. The complete HMT design process consists of a sequence of 6 specification steps, two of which will be discussed in-depth within this paper: first, the conceptual hypermedia design of HMT is introduced, because it serves as the basis for all subsequent design steps. Second, temporal design with HMT is described in detail, since audio, video, slide shows, and time-based animations play a more and more important role in today's applications.

1 Introduction

Due to the technical progress in hard- and software development, the availability and acceptance of hypermedia applications has increased significantly within recent years. As a consequence, the interest in hypermedia models, methodologies and CASE tools has grown accordingly. Especially the success of the World Wide Web has lead to increased research efforts in this area, and several data models and methodologies for hypermedia design have been proposed: HDM [6] has been one of the early examples, followed by other approaches like OOHDM [19], RMM [10,11,12], ARANEUS [2,3], or Strudel [5]. Although based on different data models and design processes, these methodologies have one common characteristic: they focus mainly on presentation issues. Although this might be sufficient for certain classes of applications, next generation approaches should also address additional aspects:

First, time is an important aspect for hypermedia applications, which is not supported by most methodologies so far, although the possibility to initialize and synchronize the different elements is a crucial feature when designing hypermedia applications. Second, a more powerful link concept is needed covering advanced features like span-to-span links or link context as described in the Dexter Hypertext Reference Model [9] and the Amsterdam Hypermedia Model [7]. Third, advanced design primitives should be provided, allowing to model both query interfaces and components for the manipulation and maintenance of the underlying data source (which will be some kind of database in most cases). Last but not least, offering the ability to manipulate data on the server inevitably requires a solid authorization concept. By now, aspects of access restriction are usually handled at implementation level instead of being covered by a modeling methodology.

Based on the Relationship Management Methodology (RMM), we have developed an advanced design methodology in order to close these gaps. The *Hypermedia Modeling Technique* (HMT) extends the capabilities of RMM by offering new and enhanced design primitives for querying and manipulating the underlying data source, for specifying access restrictions at different levels, for defining hyperlinks with extended functionality, and for the specification of temporal dependencies. Additionally, we partially changed syntax and graphical representation of the original design primitives in order to provide a more problem adequate, intuitive and easy to use interface.

Since we want to focus on the aspect of temporal hypermedia design, the other HMT design steps will not be discussed exhaustively in this paper. Please refer to [22] for more information on these issues (based on former works described in [23,20]). The remainder of this paper is organized as follows: Section 2 gives an overview of the HMT design process, section 3 presents a detailed description of the HMT conceptual model, and section 4 discusses the temporal design step of HMT. The paper ends with conclusions in section 5.

2 The HMT Design Process

The HMT design process consists of a sequence of 6 steps, where each step corresponds to a different abstraction level of design specification:

Step 1: Requirements Analysis

Requirements analysis covers aspects like the definition of the application domain, identification of intended users, and specification of system functionality and usage.

Step 2: E/R Design

After the application domain has been specified by the requirements analysis, an E/R model [4] has to be built reflecting objects and relationships from the real world application domain. If a hypermedia application has to be built upon an already existing database, the first two steps in the design process are omitted.

Step 3: Conceptual Hypermedia Design

The core hypermedia application design starts with the conceptual hypermedia design of the application, based on the E/R model developed in the previous step. Aspects like information clustering within the documents (which attributes in which document) and navigation (which links in which documents) are dealt with at this design step, which will be discussed in detail in section 3.

Step 4: Authorizational Design

Access restrictions are specified during the authorizational design phase using RBAC-techniques (Role Based Access Control, see for example [14, 17]). Based upon the conceptual hypermedia model built during the previous design step, access to certain parts of the application can be limited by specifying roles required for access.

Step 5: Logical Hypermedia Design

During the logical design phase, additional properties concerning the logical representation of a document's content are defined. This mainly refers to the spatial and temporal order, synchronization, labels or descriptions of the elements.

Step 6: Layout Design

Finally, all aspects regarding the layout of the later presentation are covered by the layout design step. This step heavily depends on the hypermedia system used for building a presentation, although certain aspects are common to most systems: The presentation layout can be described by defining a background color or image, font type and color, standard headers and footers, or default resolution and frame rate.

Throughout the paper, we will use the sample scenario shown in figure 1. It shows a model containing information about research cooperations, their projects, and national research associations acting as umbrella organizations.

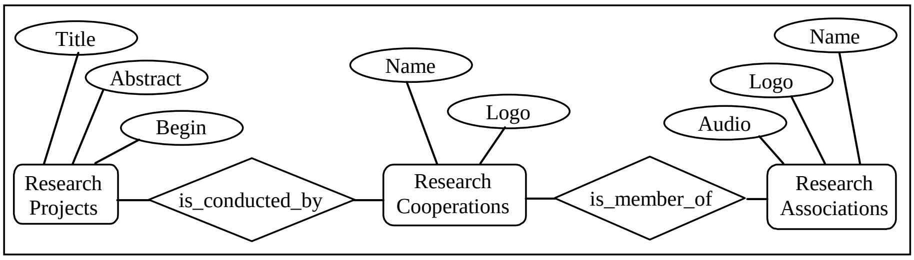


Fig. 1. Sample scenario

3 Conceptual Hypermedia Design

The HMT conceptual design of a hypertext application focuses on two aspects: *Information Clustering* is the process of grouping relevant and related portions of information into one place. *Navigational Design* determines the access paths, by the help of which the user can navigate through the application domain by following links to related information. For both tasks, HMT provides a set of primitives described in the following subsections.

3.1 Domain Primitives

In HMT, the unit of presentation is a *document*. A document usually belongs to one E/R-type (called *base E/R-type*) and is identified by a UID (unique identifier, often a meaningful string like "project_overview"). It may contain *attributes* of the base E/R-type, additional information not contained in the E/R model (*adds*), *structural links*, and *access structures* as described later in section 3.2. It may also contain so called *element-groupings*, which are named sets of attributes, adds, structural links, access structures or other element-groupings allowing to build nested structures.

Special cases are documents or element-groupings without a related E/R-type: These can only contain adds, structural links, element-groupings (again without E/R-type), or a restricted set of access structures (which will be described in the following section). They are typically used for the first page (homepage of an application or other documents where no information from the data source is to be displayed.

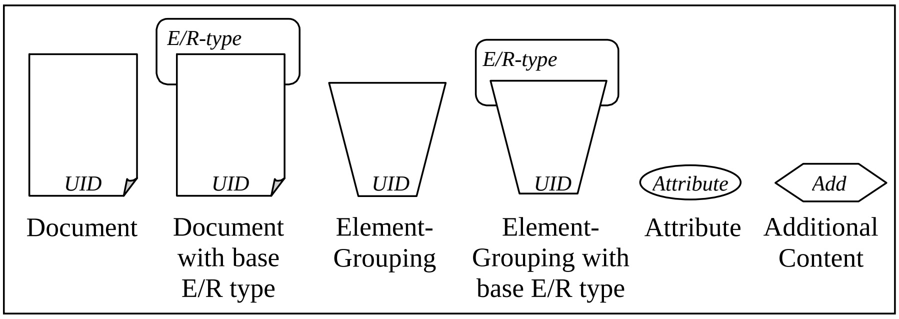


Fig. 2. basic domain primitives

In addition to the basic document primitives described in this paper, HMT also offers a set of *typed documents* like *query documents* or *input documents*, used for building interactive user interfaces for querying or manipulating the underlying data source. These typed documents are labeled with a special symbol in the upper left corner of the corresponding design primitive (for example, query documents are labeled with a question mark). Since temporal modeling with HMT refers mainly to the basic document type, we will not discuss typed documents within this paper.

3.2 Access Primitives

In order to display hyperlinks to or information from other entity types than the base E/R-type, a document or element-grouping can also contain access structures like structural links, TOCs (table of contents), guided tours, or slide shows.

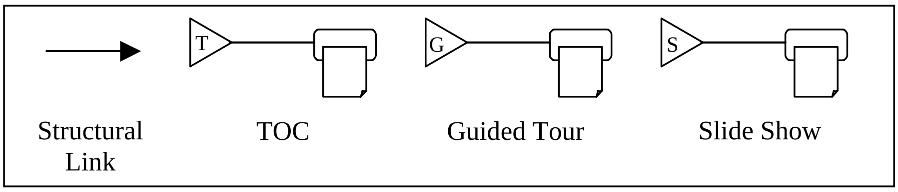


Fig. 3. basic access primitives

A structural *link* originates from an anchor element, which may be an attribute, adds or element-grouping, and points to a target element, which is either a document or an element within a document. This corresponds to span-to-node and span-to-span links as defined in the Dexter Hypertext Reference Model [9]. TOCs, Guided Tours, and Slide Shows reference a (possibly empty) set of entities of another entity type related to the current base E/R-type. These three structures differ in the way they offer access to the information they reference:

TOCs provide a list of descriptions of each referred entity. The contents of this description are defined by the document which is attached to the TOC (see figure 3). A Guided Tour does not display a list of all relevant entities, but shows one single entity of the set together with links pointing to the next element. A Slide Show is like a Guided Tour, but displays the entities automatically one after the other instead of waiting for the user to select the next element. Information about the period of time used to display each entity or the overall time for the Slide Show is specified in the temporal design phase of step 5 (logical design).

The basic link primitive described above causes the current document to be replaced with the target document the link is pointing to. The Amsterdam Hypermedia Model [9] proposes a more powerful approach by introducing the notion of *link context*, describing where the target of a hyperlink is presented and how the source behaves after the link has been selected. HMT deals with this aspect by extending the basic link primitive. Structural links which open a new document are drawn with a small box in the middle of the arrow (see figure 4). If the original document should stop its presentation, the box contains a black square (resembling the stop button of audio or video devices). If the original document should continue its presentation, the box contains a black triangle (resembling the play button of audio and video devices).

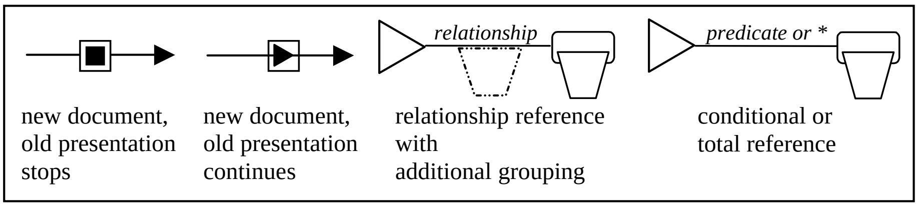


Fig. 4. advanced access primitives

Furthermore, TOCs, Guided Tours, and Slide Shows do not always have to be displayed on a new document (*external* TOC/Guided Tour/Slide Show), but can also be included inside the original document (*internal* TOC/Guided Tour/Slide Show). In this case, the document which describes the contents of the access structure has to be replaced by an element-grouping as shown in figure 4. For referencing the set of target entities, access structures can use three different approaches:

- Relationship reference
  By using the name of a relationship, the access structure references all entities of the target entity type which are related to the current entity by this relationship. If the relationship itself contains attributes, an additional *relationship-grouping* with elements of this relationship can be defined as shown in figure 4.
- Total reference
  If the access structure is labeled with "*", all entities of the target entity type are referenced.

- Conditional reference
  The access structure can also be labeled with a predicate specifying a subset of the set of target entities. For example, the predicate "begin > 1998-1-1" for a project TOC references all projects that started 1998 or later. Conditional references can be combined with relationship references.

3.3 Sample Application

In the following example, we model the hypermedia presentation of a research association. The starting document (homepage) includes name and logo, some welcome text and an audio track. At the bottom of the document, a standard footer with links to the homepage and the webmaster is displayed. An external TOC leads to a page with information about associated research cooperations.

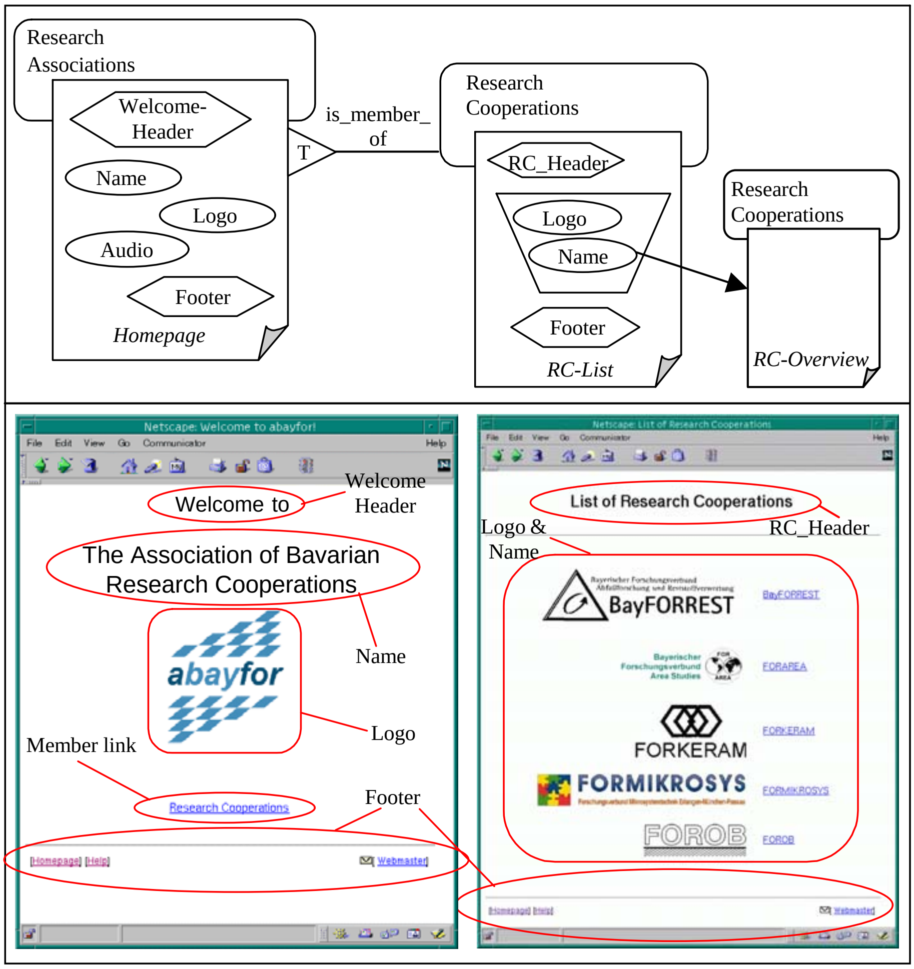


Fig. 5. HMT model and screenshots of the sample application

4 Temporal Design

The temporal specification of a hypermedia presentation is an important aspect which has to be supported by hypermedia modeling methodologies. Not only the duration or sequential execution of the presentation's elements, but also various kinds of interdependencies between the different elements have to be specified.

Former approaches were often based on scripting languages, for example Lingo [15]. Although very powerful, these approaches are rather inflexible concerning changes within the presentation, and interdependencies between different elements are often not obvious from the coded specification. Besides, programming skills are required which restricts the number of users for such a methodology. Another classical approach is to use a timeline and to specify absolute positions for each element. This very intuitive technique is rather inflexible regarding changes (all dates "behind" the changed one have to be recalculated), and elements with infinite duration cannot be modeled with this approach, because all dates are absolute points of time. Newer concepts do not rely on exact points of time, but allow to specify temporal relationships between the different elements. Based on the work of Allen [1], several approaches for modeling temporal relations have been proposed [13, 8, 21, 16, 18].

The temporal design concept of HMT is inspired by these works and tailored to the special features of its conceptual model. HMT allows to specify the time each design primitive of a document or grouping is active (that means visible). The scope of a temporal specification is always the surrounding HMT grouping or document, which eases the task of synchronizing complex parts of a presentation.

4.1 Temporal Design Primitives

In order to specify aspects of time, the conceptual HMT design primitives are extended by some temporal design primitives. Each document or grouping defines its own local presentation, which is specified within a dashed box attached to the document or grouping. Within this dashed box, the start of a presentation is marked with a special symbol, which is connected to the first element of the presentation.

◆———	Start of the Presentation
<T> *<name>*	Temporal representation of the HMT Design Primitive *name*, T specifies the duration
n*<T> *<name>*	Temporal representation of the Slide Show Primitive *name*, [T] specifies the periode of time used for each element

Fig. 6. HMT temporal design primitives

Each element is represented by a rectangle containing the element's name. The length of the rectangle is primarily determined by the length of its name and is no indicator for the element's absolute duration. However, relations between elements are to some extent symbolized by the relative position of the elements against each other (for example "element a includes element b", see next section). If the absolute duration of an element has to be specified, the corresponding period of time *T* is written above the rectangle. Two cases have to be differentiated:

If timeless elements like text or images are concerned, the effect is trivial. These elements are simply displayed for the specified period of time and then vanish. In contrast to this, time-based components like audio or video elements already have a predefined duration T_P. If they are assigned a certain time of duration T_S, (specified duration) these elements are either cut off if $T_P > T_S$ or restarted if $T_P < T_S$.

A special kind of element are Slide Shows, because for them a temporal specification is mandatory. Three different approaches are possible:

- If an overall duration *T* is specified, the period of time *t* used to present a single element of the Slide Show is calculated as t = T/n, where n is the number of elements of the Slide Show. This ensures that a given presentation doesn't have to be changed if the number of elements of the Slide Show changes.
- Alternatively, it is also possible to specify the period of time *T* used for a single element of the Slide Show by using the expression n**T*. In this case, changes to the number of elements of a Slide Show influence the overall duration and thus have effects on the synchronization with other elements.
- If all elements of the Slide Show are time-based (like video or audio), then no explicit specification of T is necessary, because each element has its own duration.

Since each element of a document or grouping has its own temporal representation, also access structures like structural links or TOCs can be temporarily restricted and synchronized with other elements, for example video clips or audio tracks.

4.2 Temporal Relations

For interval based approaches, Allen [1] has identified 13 characteristic relations, which can be reduced to 7 if the inverse relations are derived by swapping the corresponding elements. HMT adopts and extends these relations:

First, a new relation *synchronizes* is introduced which is especially suited for time-based elements like video or Slide Shows. Second, a simple arrow denotes a delayed sequential execution of two or more elements, whereas an arrow originating from a bullet is used for specifying parallel execution. The element where the arrows originate from are called *synchronizing* elements, the arrows' targets are the *synchronized* elements. Third, the *equals* relation is left out, because it can be replaced by a combination of the *starts* and the *finishes* relation. Figure 7 shows an overview of the temporal relations used in HMT.

For a formal description of the temporal relations in HMT, we use the following syntax: Let *x* be an HMT element, then *T(x)* denotes the overall duration of *x* (either explicitly specified or implicitly defined in case of time-based elements), *start(x)* and *end(x)* define the starting point and the end point of *x*. Please keep in mind that *start(x)* and *end(x)* are abstract measures and are only used to define the relations' semantics, they are not part of the temporal model of HMT.

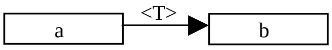
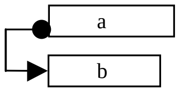
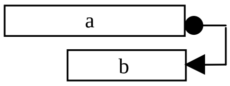
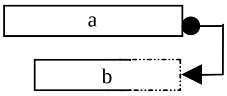
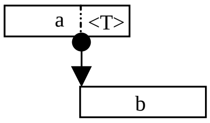
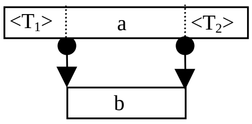
	Immediate sequential execution ("a starts b")
	Deferred sequential execution, T specifies the delay ("a starts b")
	Parallel execution, synchronized beginning ("a starts b")
	Parallel execution, synchronized termination, time-based *bs* are cut off or restarted ("a finishes b")
	Parallel execution, synchronized termination, time-based *bs* are slowed or accelerated ("a synchronizes b")
	Parallel execution, overlapping, T specifies overlapping interval ("a overlaps b")
	Parallel execution, inclusion, T_1 and T_2 specify the delay ("a includes b")

Fig. 7. HMT temporal relations

a meets b: This defines an immediate sequential execution, where *b* is started after *a* has finished. Prerequisite for this relation is that *a* is either a time based element (thus having a predefined duration), or explicitly restricted by specifying *T(a)*.

```
Formal: start(b) := end(a)
```

a before b: Similar to the previous case, *b* is started after *a* has finished. Instead of an immediate execution, the start of *b* is delayed by *T*. Similar to the previous case, element *a* must have a finite duration *T(a)*.

```
Formal: start(b) := end(a) + T
```

a starts b: The elements *a* and *b* are starting simultaneously and are executed in parallel.

```
Formal: start(b) := start(a)
```

a finishes b: This relation synchronizes two elements *a* and *b* in the way that the end of *a* causes *b* to terminate, too. Prerequisite for this case is that *a* has a finite duration (either specified or predefined by time-based elements).

```
Formal: end(b) := end(a)
```

While the synchronization of timeless elements is trivial, time-based elements have to be treated differently. If *b* would normally end before *a* has finished, then *b* has to be restarted in order to bridge the gap until *a* terminates.

```
Formal: T(b)< T(a)-(start(b)-start(a)) ⇒ "restart b"
```

If a should terminate before b, then b has to be stopped.

`Formal: T(b)> T(a)-(start(b)-start(a)) ⇒ "stop b"`

a synchronizes b: For timeless elements, this relation has the same effect as *a finishes b*. But if *b* is a time-based element and has been initiated by some other element *x*, then *b* is slowed or accelerated in order to synchronize with *a*.

`Formal: T(b) := T(a)-(start(b)-start(a))`

a overlaps b: The elements *a* and *b* are executed partially in parallel, where *T* specifies the overlapping interval. Element *a* must have a finite duration *T(a)*.

`Formal: start(b) := end(a) - T`

a includes b: The elements *a* and *b* are executed in parallel, and *a* starts after and ends before *b*. Element *a* must have a finite duration, element *b* may be infinite unless T_2 is specified.

Formal: start(b) := start(a)+T_1, end(b) := end(a)-T_2

4.3 Sample Application

Modifying our sample scenario from section 3.3, our new homepage shall start with a welcome audio track and an introductory slide show, where name and logo of all connected research cooperations are shown (the order of the elements on the homepage and the order of the Slide Show elements are defined in the logical design step, which is left out due to space limitations). This slide show has to be synchronized with the audio track independent of the number of slides to be presented. Afterwards, name and logo of the research association and the standard footer should be presented together with the external TOC on research cooperations.

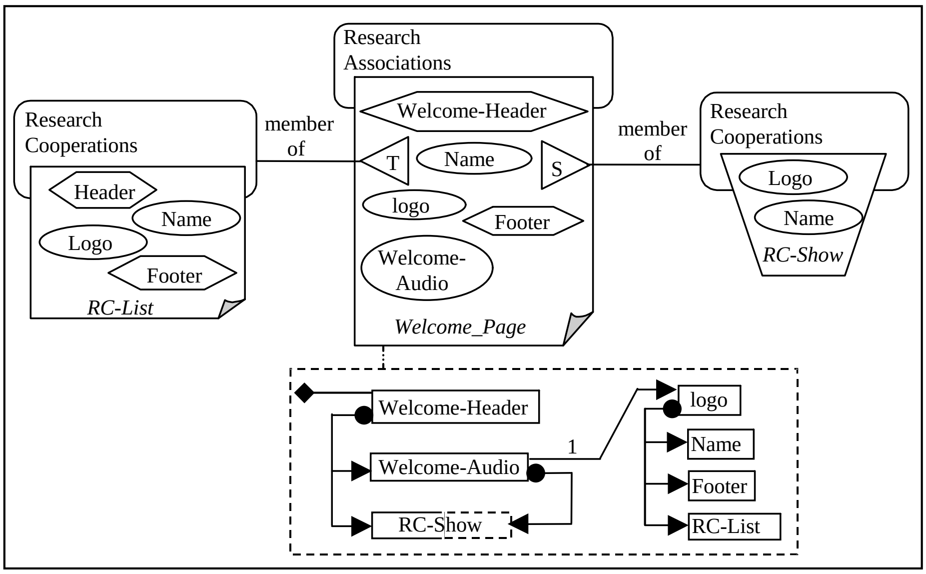
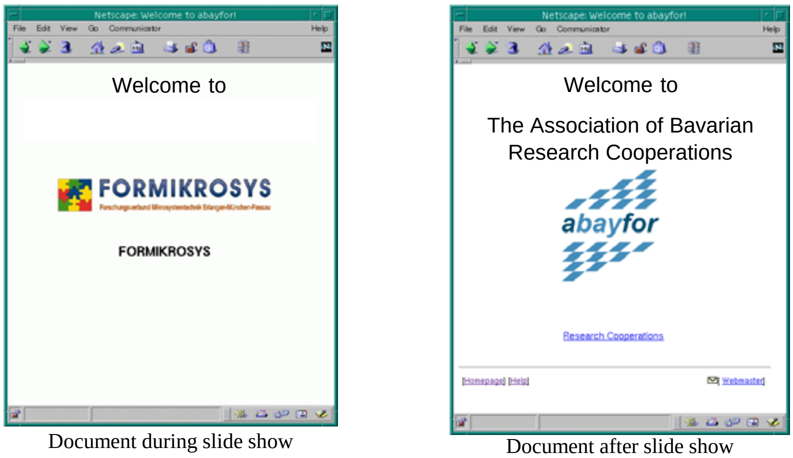


Fig. 8. Temporal specification of the sample scenario

Figure 8 shows the HMT specification, figure 9 shows screenshots of the presentation in different stages.



Document during slide show

Document after slide show

Fig. 9. screenshots of the sample application in different stages

5 Conclusion

In this paper, we have presented the Hypermedia Modeling Technique (HMT), a novel approach to structured hypermedia design. Unlike existing design models and methodologies, HMT covers not only aspects of information presentation, but provides advanced concepts for modeling temporal aspects of hypermedia applications, enhanced hyperlink functionality (link context), interfaces for querying and updating the underlying data source, access restrictions and adaptive documents

In the first part of this paper, we presented an overview of the complete HMT design process, and introduced the HMT conceptual model. Based on an E/R-model of the application domain, the information is clustered within hypermedia documents and groupings. Navigational structures can be defined by using structured links or advanced primitives like TOCs, Guided Tours, or Slide Shows. A sample application has been presented in order to show the usage of the HMT conceptual model.

The second part of our paper discussed in detail temporal hypermedia design in HMT. Each document or grouping is modeled as an independent presentation, and each element (including access structures) gets its individual temporal specification. For synchronization purposes, seven different dependencies between the elements of a hypermedia document have been identified and added to the HMT design primitives. A modified version of the sample scenario has been used to illustrate the usage of the temporal design primitives of HMT.

References

[1] Allen, J.F.: "Maintaining Knowledge about temporal intervals*". Communications of the ACM* 1983/26, Vol. 11, pp. 832-843

[2] Atzeni, P., Mecca, G., Merialdo, P.: "To Weave the Web". *Proc. 23rd International Conference on Very Large Databases (VLDB'97)*, Athens,1997, pp. 206-215

[3] Atzeni, P., Mecca, G., Merialdo, P.: "Design and Maintenance of Data-Intensive Web Sites". *EDBT98*, Valencia, 1998, pp. 436-450

[4] Chen, P.S.: “The Entity-relationship Model – Towards a Unified View of Data”. *ACM TODS*, Vol. 1, No. 1, 1976.

[5] Fernandez, M., Fiorescu, D., Kang, J., Levy, A., Suciu, D.: "STRUDEL: A Web-site Management System". *ACM SIGMOD 1997*

[6] Garzotto, F., Paolini, P., Schwabe, D.: “HDM- A Model-based Approach to Hypermedia Design”. *ACM Transactions on Information Systems*, 11, 1 (1993), pp. 1-26.

[7] Hardman, L., Bulterman, D.C.A., Rossum, G.: "The Amsterdam Hypermedia Model". *Communications of the ACM*, 1994/2, Vol.37, pp. 50-62

[8] Hoschka, P.: "Synchronized Multimedia Integration Language". http://www.w3.org/TR/WD-smil-971109

[9] Halasz, F., Schwartz, M.: "The Dexter Hypertext Reference Model". *Communications of the ACM*, Feb. 1994, Vol. 37, No. 2, pp. 30-39

[10] Isakowitz, T., Kamis, A., Koufaris, M.: “Extending the capabilities of RMM: Russian Dolls and Hypertext”. *Proc.30th HICSS*, 1997.

[11] Isakowitz, T., Kamis, A., Koufaris, M.: “Reconciling Top-Down and Bottom-Up Design Approaches in RMM”. *Proc. Workshop on Information Technologies and Systems (WITS97)*, Atlanta, GA, Dec. 1997.

[12] Isakowitz, T., Stohr, E., Balasubramanian, P.: “RMM: A Methodology for the Design of Structured Hypermedia Applications”. *Comm. of the ACM*, 38(8), pp. 34-44.

[13] Jourdan, M., et al.: "Authoring Environment for Interactive Multimedia Documents". INRIA Rhone-Alpes, Montbonnot, France, http://opera.inrialpes.fr/OPERA

[14] Lupu, E., Sloman, M.: "Reconciling Role Based Management and Role Based Access Control". *Proc. 2nd ACM RBAC Workshop*, Fairfax, VA, USA, Nov. 1997

[15] Macromedia: Director 4.0 *User's Guide*.

[16] Rossum, G., Jansen, J., Mullender, K.S., Bulterman, D.: "A Presentation for Portable Hypermedia Documents". *Proceedings of the ACM Multimedia ´93*, pp.183-188

[17] Sandhu, R., Coyne, E.J., Feinstein, H.L., Youman, C.E.: "Role-Based Access Control Models". *IEEE Computer*, 29(2), Feb. 1996, pp. 38-47

[18] Song, J., Kim, M.Y., Ramalingam, G.: "Interactive Authoring of Multimedia Documents". *Research Report RC 20369*, T.J. Watson Research Center, IBM Research Division Yorktown Heights, NY

[19] Schwabe, D., Rossi, G., Barbosa, S.: “Systematic Hypermedia Design with OOHDM”. *Proc. Hypertext 96*, Washington, Mar. 1996.

[20] Sommer, U., Zoller, P.: “WebCon: Design and Modeling of Database Driven Hypertext Applications”. *Proc. 32nd HICSS*, 1999.

[21] Yu, J.: "A simple, intuitive hypermedia synchronization model and its realization in browser/java environment". Technical Note 1997-027, Digital Equipment Corporation Systems Research Center, Palo Alto, CA, October 1997

[22] Zoller, P.: "HMT: Modeling interactive, adaptive Hypermedia Applications". To appear in : *Siau, K., Rossi, M. (editors): Modelling Methodologies for the next Millenium*, Idea Group Publishing, spring 2000.

[23] Zoller, P., Sommer, U.: “WebCon: A Toolkit for an Automatic, Data Dictionary Based Connection of Databases to the WWW”. *Proc. 1998 ACM Symposium on Applied Computing*, Atlanta, 1998, pp. 706-711