Integrated Database Services for Multimedia Presentations

Susanne Boll, Wolfgang Klas, and Michael Löhr
GMD-IPSI
Integrated Publication and Information Systems Institute
Dolivostr. 15
D-64293 Darmstadt, GERMANY
email: {boll, klas, loehr}darmstadt.gmd.de

In: S. M. Chung (editor), Multimedia Information Storage and Management,
to be published by Kluwer Academic Publishers, 1996.

CONTENTS

1

INTEGRATED DATABASE SERVICES FOR MULTIMEDIA PRESENTATIONS

Susanne Boll, Wolfgang Klas, and Michael Löhr *

GMD - Integrated Publication and Information Systems Institute (IPSI)
** Technical University of Darmstadt, Department of Computer Science*
Dolivostr. 15, D-64293 Darmstadt, Germany

ABSTRACT

Advanced multimedia applications call for integrated database system support for the management of multimedia data. Traditional database management systems do not provide appropriate concepts and services for the integrated modeling, management, and interactive presentation of multimedia data. Our approach towards a multimedia database management system offers *presentation independence* to applications, which is in line with traditional database system services like data independence and multi-user support. This new service is realized by the integration of appropriate concepts for continuous object management, the representation and presentation of multimedia data, and the execution of interactive end-user presentations into an open object-oriented database system. The resulting multimedia database system consists of database clients for the interactive presentation of multimedia data, which can be adapted to particular needs of applications, and a database server for the efficient management of and access to multimedia data.

1 INTRODUCTION

In recent years computer systems have been used to present information in increasingly extensive and appealing forms. Information conveying is no longer supported by means of only traditional media types such as text and images but also by media types such as video and audio. The usage of multiple media types makes information presentations more attractive and improves the value of the information presented. Some media types such as text and image are *time-independent* (*discrete*) whereas the values of other media types such as video or audio change over time, and therefore these media types are *time-*

dependent (*continuous*). We follow the definition of [27] and refer with the term *multimedia application* to those applications which include both time-dependent and time-independent media types.

When looking at the different types of multimedia applications one can find different characteristics with regard to the challenging problems. In the following we give examples of some multimedia applications with interesting properties relevant in the framework of our work.

With Video-On-Demand (VOD) applications users access video information from one or more remote servers. Recent developments for VOD systems support to a high degree the time-dependent presentation of a single media type, at the same time they neglect the editing, capturing, and modeling of single media types as well as multimedia compositions. VOD applications show low complexity with respect to presentation functionality as only videos are retrieved from the server and continuous data are delivered uni-directionally to the clients. At the same time VOD applications need high-performance storage and networking systems due to the high data volume of time-dependent media types.

Another class of applications are computer games. They are based on interactivity between the user and the computer and can show high complexity. Games can be played locally or remotely by one or more players. Interactive games use branching techniques or advanced algorithms for interactivity [27]. Games are 'custom made' and lay emphasis on the idea behind the game, the behavior of the interface, stability, and robustness.

Video conferencing applications focus on real-time communication aspects and live synchronization of video and audio streams. The synchronization is inflexible as these applications have to reproduce the data streams at a target system different from the source where the streams are captured. Video conferencing is concerned only with tightly coupled continuous video and audio streams. The realization of advanced video conferencing technology is to a high degree dependent on the development of networking technology.

Multimedia applications such as computer-based training, learning programs, and electronic encyclopedias require complex content modeling and specific presentation techniques reconciled with individual user needs. This also applies to home shopping, ticket reservation, and electronic product catalogues. These applications use both discrete and continuous media types. They focus on the interactive presentation of multimedia information and are reasonable only in a multi-user environment in which several users have access to the multimedia

information. In some cases the information consumers need to have direct contact to the information producer.

The latter applications challenge a multimedia system in many different ways and serve as a basis to derive demands on a comprehensive multimedia system. They also point the way to the trend in multimedia applications observed in [27] towards multi-user environments, interactivity, and bi-directional flow of multimedia information.

In extension to [27] and [21] we define an *integrated multimedia system* as a system that

1. allows for the arbitrary combination of different media types into one multimedia composition by means of specifying temporal, spatial, and semantic relationships,
2. allows for the manipulation and presentation of multimedia compositions as a new single media type,
3. allows at the same time for the presentation and manipulation of single media and parts of media independently of their usage in a multimedia composition,
4. allows for media translations as an optional additional feature of an integrated multimedia system,
5. supports effective management of multimedia information, and
6. provides a system architecture suitable for single-user and multi-user access.

In this chapter we present an approach for the construction of a comprehensive system that fulfills most requirements demanded by an integrated multimedia system and supports a wide range of different multimedia applications. Our approach concentrates on the issues of modeling, storage, retrieval, buffer management, and presentation of multimedia information whereas we base our work on current development in operating systems and communication services. As a starting point we use a database management system and develop concepts needed for integrated multimedia database management systems.

In the following section the usage of database management systems for the handling of multimedia information is motivated. The concepts related to

the modeling and to the execution of a presentation of multimedia data are presented which have to be added to achieve the full functionality of an integrated multimedia system as defined above. Our prototype of a multimedia database management system is presented in section 3 with emphasis on system architecture issues, multimedia composition, and multimedia presentation. Two sample reference applications based on our prototypical implementations are briefly described in 4. In the conclusion we address the prospects for our research interests in the near future.

2 CONCEPTS FOR THE INTEGRATION OF MULTIMEDIA DATA

In this section we elaborate concepts for a system that meets the demands to be fulfilled by an integrated multimedia system as given in section 1. We are mainly concerned with the integration and management of multimedia information, multimedia composition, interactive presentation of multimedia compositions, and the consequences on the architecture of a multimedia database system.

2.1 Management of Multimedia Data

Media of high data volume such as audio and video can be introduced into a multimedia system by integrating continuous data types into programming languages and data models. As a consequence, the need for an efficient management of the continuous data types arises. Many applications use file systems to store multimedia data. It has become clear, however, that neither conventional file systems nor traditional database management systems can effectively create, store, manipulate, and present the data of multimedia applications [17, 26, 24, 6]. The concepts of database management systems nevertheless seem to be reasonable for a uniform handling of multimedia data. Database management system support is of special interest if complex data structures must be managed and several users need to access this data. To provide reliable multi-user access to multimedia data, database management system features such as controlled redundancy, data independence, concurrency control, integrity, data security, data protection, and ad hoc query languages are desirable.

In solving the problem of an efficient management of multimedia data with database systems one can identify the following approaches: the extension of an existing database system, the development of a new database system, and the combination of a traditional database systems with a media-specific database system. The extension of an existing database system is quite obvious as the development of a new multimedia database system from scratch is very complex and wastes the effort already put into the development of existing database systems. The combination of a traditional database system with a media-specific system often lacks integration of conventional database system services with media-specific services.

The modeling of continuous data can be realized in different ways. In traditional database models no direct support for the representation of continuous data like audio and video is available. Obvious solutions in such models are based on *long fields* or *BLOBS* which allow to store big amounts of data in the database. The drawback of such solutions is the fact that the database system stores the data uninterpreted, and, hence, cannot provide any additional service to an application. For example, an application can only retrieve a byte string from the database system and has to determine the proper data format like MPEG or JPEG as well as the appropriate processing techniques on its own. Obviously, the database system does no longer provide data independency to the application.

Object-oriented database systems seem to form a suitable basis for the extension of a database system to a multimedia database system [16, 11]. In object-oriented database systems one can model continuous data by means of abstract data types. That is, one can define classes with attributes and methods for the representation of, e.g., a video data stream. This allows to capture specific semantics associated with video data within the database. For example, a class could explicitly reflect the structure of an MPEG video by means of I-, B-, and P-frames, or by means of scene cuts. Or the class could hide specific details of a data format and provide a kind of standardized delivery format (based on appropriate translation operations) to the application. Although this allows already for more advanced processing of continuous data, the database system itself still does not understand the characteristics of continuous data. For example, the database system still cannot offer appropriate buffer and object management for time-dependent data like video and audio. This is due to the fact that this type of modeling of continuous data is just an application of the data model. No integration of modeling primitives into the data model or system takes place.

Open database systems like Illustra [13] and VODAK [10, 18] allow for the adaptation of the system to specific application needs. Illustra allows to plug-in DataBlades[1], which allow the database system to make internal use, e.g., in query processing, of some functionality encoded by the data types contained in a DataBlade. VODAK [10] is an open object-oriented database system which allows to tailor the data model to specific needs of an application by adding specific classes, data types, and functionality. Although this type of systems provide already quite interesting support for multimedia applications, they still lack integrated support for time-dependent data.

Consequently, another approach towards the integrated handling of multimedia data is the integration of built-in data types for continuous data into an object-oriented data model. Either the database system is extended by built-in data types for the various kinds of continuous data like MPEG1-based video, MPEG2-based video, JPEG video, 16-bit audio, 8-bit audio, etc., or a more generic data type serving as a common basis to most continuous data types is integrated into the system. In contrast to individual built-in data types a generic data type can be more easily integrated with continuous buffer and object management services, which allows the database system to handle time-dependent data according to its characteristics. The generic data type can serve as the basis for the modeling of the various kinds of continuous media by just using the regular modeling primitives like classes, attributes, and methods provided with the data model. Our approach follows this strategy and results in a data model which provides a generic means of handling continuous data. A generic built-in data type for continuous data serves as the basis for the modeling of specific media like audio and video (e.g., [20]). We call the resulting types *media types* in the following.

The modeling of continuous data by means of media types includes not only raw data, i.e., almost unformatted media data. Rather it includes various kinds of meta data like information about data format, coding, compression techniques, length, samples per second, frames per second, sample size, frame size, color table, etc.. Furthermore, applications may require the modeling of additional information based on the content of continuous data, e.g., scene cuts or the description of the content of a video. In addition to this type of structural information media types provide for the definition of media type-specific operations. Examples are operations for the creation of instances of media types (e.g., scanning an image), manipulation of instances (e.g., changing volume, length, and quality of an audio object), or operations needed for the presentation of instances, e.g., the playback of a video data stream.

[1] Trademark of Illustra Information Technologies, Inc.

Instances of media types are called *media objects*. A medium object is stored in the database and represents a single medium, i.e., a video or an audio. Multimedia compositions may be based on several media objects, which are often related to each other by specific temporal or spatial relationships following composition constraints. That is, media objects are only basic building blocks for the composition of multimedia presentations.

In the following, we focus our discussion on the extensions required to allow an object-oriented database system to handle multimedia data in an integrated way as outlined in section 1. The subsequent subsections will discuss details on the representation and the presentation of multimedia compositions.

2.2 Modeling of Multimedia Compositions

One of the demands to be met by an integrated multimedia system is the capability of manipulating and presenting arbitrary combinations of different media types to one multimedia composition. This calls for models which support the description of such multimedia compositions. We call such models *multimedia document models*. Prominent examples of document models are SGML (Standard Generalized Markup Language), HyTime (Hypermedia/Time-based Structuring Language), MHEG (Multimedia and Hypermedia Information Encoding Expert Group), ODA (Office Document Architecture), and OCPN (Object Composition Petri Net) which provide different means for the representation of multimedia compositions. We denote the concrete representation of a multimedia composition in the integrated multimedia system a *multimedia document*. It captures the arrangement of different media objects with temporal, spatial, and semantic interdependencies in a new time-dependent media object. The composition may be constrained by means of a composition scheme which defines the valid types of multimedia documents a designer can construct. The general kind of compositions we are concerned with are *synthetic* or *pre-orchestrated* compositions. That is, the media objects are selected from the database and are arranged prior to a presentation of the composition. Currently, we do not consider the live composition of presentations as it occurs for example in live video conferencing applications.

The media objects stored in the database form the basis for multimedia composition. Each media object contains the data necessary for its representation in the system. Each media object belongs to a media type such as image, audio, and video. The modeling of media objects as presented in section 2.1, however, does not offer sufficient information for a multimedia composition.

When composing multimedia presentations the media objects involved might be used only partly. For example, a multimedia composition may use just a few seconds of a two hour video or only a detail of an image object. Therefore, specific *presentation data* describes the selected media parts in the multimedia document in addition to the original data available with the involved media objects. Furthermore, presentation data supplement the media data in values that are reasonable only for presentation purposes but not for the modeling of the media object in the database. They concern media type-specific information such as desired presentation speed of continuous media objects or the rotation of graphics and general presentation information like the points in time for starting and terminating a presentation.

Multimedia documents must contain or refer to media objects involved in the multimedia composition. It seems reasonable to store media objects and multimedia documents separately. Media objects in an object-oriented database can be referred to by means of unique object identifiers. The multimedia documents then contain for each media object involved the presentation data and a reference to the media object. The object identifiers establish the relationship between media objects in the database and the multimedia document. This makes the multimedia documents independent of the media objects which are used only in the course of a later presentation of the multimedia composition. And this allows for the manipulation of media objects independently of their usage in a multimedia composition, which is again one of the demands to be fulfilled by an integrated multimedia system as presented in section 1. At the same time the separation meets the objectives of database management systems, i.e., to avoid redundancy and to allow for re-usability of media objects, as one media object can be used in different multimedia compositions and in different quality.

The separate management of multimedia documents and media objects, however, causes difficulties as for the consistency of multimedia documents with media objects. If a media object is deleted from the database all references of multimedia documents to the object are no longer valid. Changes to media objects can invalidate the presentation data of multimedia documents if, e.g., a video object of 20 seconds is included in a composition and the same video object is later on cut down to a length of 10 seconds. Changes to media objects involved in multimedia composition should be validated against their usage in multimedia compositions.

For each media object used in a multimedia composition a *presentation object* denotes the combination of both the reference to the respective media object in the database and the presentation data necessary for the object's presenta-

tion. The information available with each presentation object is sufficient for a *single medium presentation*. A multimedia document captures furthermore the actual arrangement of the presentation objects. When creating a multimedia composition the presentation objects can be placed along a time axis and/or be related in space and time to other presentation objects. Special presentation objects are *interactive presentation objects*. They represent media objects for interactions via buttons, sliders, and text input fields. They can be related to one or more presentation objects in the spatial as well as temporal dimensions of a multimedia composition. Additionally, the modeling of interactive presentation objects includes the action which is to be executed in response to a user's interaction. We consider the different relationships between presentation objects in a multimedia composition in more detail in the following. Figure 1 illustrates the dimensions of space, time, and interaction in a multimedia composition.

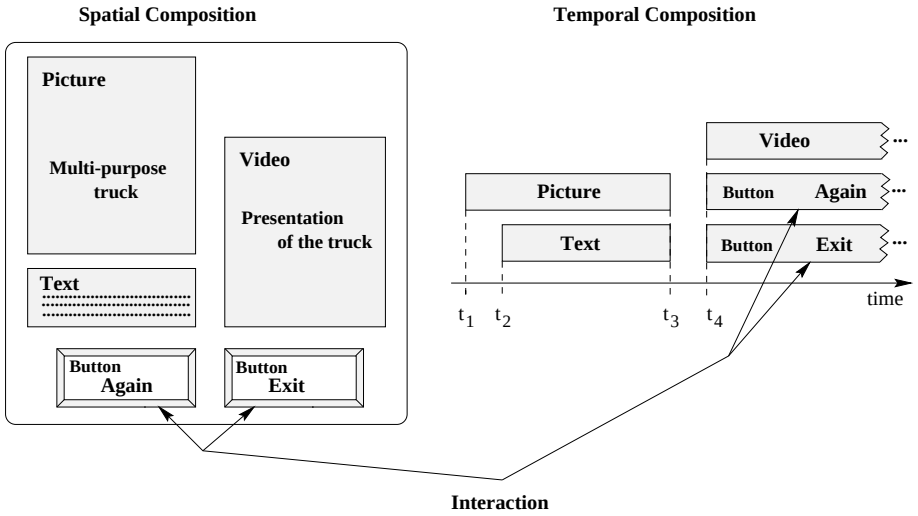


Figure 1 Multimedia composition in space, time, and interaction

Temporal Relationships

To make the presentation objects perceptible in a multimedia presentation both continuous and discrete media are to be presented for a certain period in time. The continuous media change their values over time whereas the discrete media keep their values. The presentation objects are to be placed along a time axis. For each presentation object at least the start, possibly also the end, of its

single medium presentation can be specified as well as alterations to the single medium presentation such as the font size of a text object or the fading out of an audio object. Therefore, a proper modeling of the temporal course of a presentation has to be chosen and captured in a multimedia document. Two fundamental approaches have been proposed:

- Point-based modeling of time: The point-based model represents time by means of points in time. A point in time is of length zero and corresponds to a certain position on the time axis. The set of points in time is completely ordered. For any two points in time one of the relationships *before* ($<$), *after* ($>$), or *equals* ($=$) holds. This is a simple representation of time with a small number of temporal relationships.
- Interval-based modeling of time: *Time intervals* form the basis of the temporal modeling introduced by [2]. The model offers 13 basic time relationships for any pair of time intervals, e.g., two time intervals can *overlap* each other, *start* simultaneously, etc..

The point-based model and the interval-based model are equivalent in so far as each time interval $[a, b]$ equals two points in time, namely the start point a, the end point b of the time interval, and the relationship $a < b$. It is quite obvious to model the temporal course of a multimedia presentation with time intervals, which represent the presentation durations of single media.

The modeling of time with intervals, however, may not be sufficient when it comes to half-open time intervals and indefinite interval relationships. Presentation objects, e.g., for interaction, might have a defined presentation start time but no definite end, as the presentation of interactive presentation objects ends with their selection by a user. Two continuous presentation objects could be simultaneously started without knowing the end of each single presentation. Therefore, enhanced temporal interval models have been proposed to handle open time intervals and indefinite interval relationships [12, 36].

When creating a multimedia composition it might be desirable to change the appearance of presentation objects during their presentation. This alteration can be requested interactively or can be specified in the multimedia composition. The modeling of the temporal course must therefore include the points in time when and in which way a single medium presentation is to be changed. In addition, in case of continuous changes to a presentation object, e.g., fading in or fading out of an audio, the time interval and the quantity of the change

must be specified. The modeling of parameterized operations to change the presentation of a media object is presupposed in this context.

Spatial Relationships

Visual presentation objects need additional information which specifies their spatial position. There can be defined spatial relationships between presentation objects as well. If more than one presentation object is visible in the course of a presentation, presentation objects may overlap. Visual presentation objects may be placed to the left, to the right, above, beneath, etc., of another presentation object. One can think also of relationships such as 'XOR' in the overlapping area.

Interactive Presentation Objects

Multimedia presentations should allow for interactions with users. Therefore, the user may want to use different interactive presentation objects to influence the presentation in various ways. In the following, we present different categories of user interactions with a multimedia presentation system. As from our point of view multimedia compositions should allow for the inclusion of interactive presentation objects we identify in the following those interactions we think to be reasonable in multimedia documents.

- *Scaling actions* change the visual appearance of a multimedia presentation, e.g., alteration of the size of a video window, the volume of an audio, or the color table for images.
- *Movie actions* or *standard interactions* are comparable with the actions on the operating panel of a video player. Standard interactions are *play*, *stop*, *pause*, *resume*, *fast forward*, *fast reverse*, *reverse*, etc.. They are related to speed and direction of multimedia presentations.
- *Decision actions* influence the course of a multimedia presentation. This is achieved with interactive presentation objects that let a user decide at definite points in the presentation how the presentation is to be continued.

Scaling actions need not necessarily be specified in multimedia documents. They refer to single presentation objects and the presentation component can allow for certain scaling operations depending on the modeling of the respective media type in the database.

Standard interactions refer to the entire temporal course of the multimedia presentation. They do not have to be explicitly captured in a multimedia document. Moreover, they belong to the operations that have to be modeled with the data types of multimedia compositions.

Decision actions, however, have to be specified in a multimedia document. They are defined by the designer/user of a multimedia presentation and are therefore called *user-defined interactions*. It has to be specified to which other presentation objects an interactive presentation object refers to and what kind of influence the interaction has on the multimedia presentation. One can think of decision actions that are related to a group of presentation objects as well as actions which influence the entire presentation. The reference can be explicitly modeled or be hidden in the action related to the interaction.

Of special interest is the temporal arrangement of interactive presentation objects in the multimedia composition. The presentation duration of interactive presentation objects is not necessarily determined. Therefore, they cannot be represented by closed time intervals in all cases. Closed time intervals are sufficient only if an interaction is at the users' disposal for a fixed period of time. But interactive presentation objects can also be presented until the user selects it. This is the case with decision interactions such as 'Again' and 'Exit' in the sample presentation shown in Figure 1. The modeling of half-open/indefinite time intervals is imperative for interactive multimedia documents.

Synchronization Relationships

Synchronization relationships can concern the temporal course of the presentation, the spatial arrangement of the presentation objects, or the content of the media objects involved. The *synthetic synchronization relationships* are specified in the multimedia document prior to a multimedia presentation. We are not concerned with *live synchronization* that typically occurs with person-to-person conversational services.

- Temporal synchronization: Synthetic temporal synchronization can be classified into *intra-media synchronization* and *inter-media synchronization*.
  - Intra-media synchronization: For each continuous presentation object the presentation data specifies the desired playback speed, e.g., frames per second. A presentation component for synchronization enforcement has to reproduce the specified speed and continuity.

  - Inter-media synchronization: The specification of temporal relationships between presentation objects should be covered by the temporal modeling of a multimedia composition. These relationships can also be used to specify inter-media synchronization for discrete presentation objects, e.g., the parallel start or end of presentation objects. Continuous inter-media synchronization applies only to time-dependent presentation objects for which deviations in presentation speed are to be continuously controlled and adapted, respectively.

- Spatial synchronization: Spatial relationships can be combined with temporal arrangements. For example, an image can be continuously moved from the upper left to the lower right on the screen during an audio playback.

Quality of Service Parameters

For a multimedia composition *Quality of Service* (*QoS*) parameters can be specified. Multimedia presentations can tolerate deficiencies in presentation quality to a certain degree. The human senses cannot perceive slight deviations in the presentation of continuous media. This is why multimedia presentations have soft real-time requirements. The permitted deviations of the actual presentation quality from the specified presentation quality of the synthetic composition is captured by means of QoS parameters. The different parameters denote the different demands of applications to the components of a multimedia system. At the same time the QoS parameters describe the efficiency of the system. We consider QoS parameters that hold for both single presentation objects and relationships between presentation objects. As we are not concerned with operating system and networking tasks we see QoS parameters only from the viewpoint of multimedia applications. The QoS parameters on the application level must be enforced by a presentation component in cooperation with QoS-models at lower system levels.

Time-independent QoS parameters for single presentation objects concern the quality of the presentation such as the resolution or color depth of an image or a video. For continuous presentation objects an additional permitted deviation *jitter* supplements the intra-media synchronization specification given by a desired playback speed. The QoS parameters for relationships between presentation objects mainly concern the temporal relationships between the objects. A *skew* denotes the accumulated difference in the speed compared with the specified inter-media synchronization.

Standards for Multimedia Document Models

Though a *multimedia exchange format* would be desirable several standards are evolving, e.g., HyTime, MHEG, OCPN, for multimedia document modeling that capture to some extent the aspects presented above.

- HyTime is based on SGML and is used for the description of hypermedia documents [14, 23, 9]. Presentation objects can be placed in a finite coordinate space in time, space, and any other dimension. HyTime, however, offers no description for interactive presentation objects. In reconciliation with SGML it excludes the modeling of formats.
- MHEG [15] is concerned with the composition of time-based media objects. The standard strives to the basis for different multimedia and hypermedia applications. So called MH-objects serve as a level of abstraction between the basic media objects and applications that use the objects. The MHEG standard provides a coded representation for the MH-objects by means of the object-oriented concept of classes. The MH-objects are used to build up complex multimedia documents that allow for synchronization and interaction.
- With OCPN the known petri net model is extended by periods in time assigned to places in the net [19]. The places in the net represent the presentation objects of the composition. The presentation duration of a place corresponds to a closed time interval. OCPN allows for the modeling of all of the 13 temporal relationships between time intervals of the interval-based model. The firing rules of the OCPN and a token flow through the net reproduce the temporal course of a multimedia presentation modeled with OCPN.

So far we have outlined the modeling of multimedia data and multimedia documents. Now we come to the distribution and consumption of multimedia data in a distributed database environment.

2.3 Presentation Services

In addition to the management of both media objects and multimedia documents, we propose to introduce playout facilities into the multimedia database management system. Presentation operations for media objects are introduced to the multimedia database management system with the modeling of media

types. Multimedia documents, however, form a complex media type with operations for their manipulation and interactive presentation that have to be modeled in the database as well.

We believe that a multimedia database management system should provide presentation independence as it provides data independence. From our perspective, this is a natural adjustment of database management functionality to the new requirements of multimedia applications. This allows developers to focus on the application-specific problems and relieves them from the task of encoding presentation functionality or the coupling of a presentation tool.

An integrated multimedia database system should offer integrated manipulation and presentation facilities for multimedia composition. *Playout management*, in general, is the task that deals with the realization of arbitrary pre-orchestrated interactive multimedia presentations. That is, playout management controls the presentation realization phase and is comprised of several subtasks such as the device management, the data stream handling, the enforcement of the synchronization constraints, and the handling of user interactions. The general goal is to achieve that storage, retrieval, buffer management, data transmission, as well as playout management are handled in an integrated way by the database management system [32, 29]. In the following we address architectural issues and the problems which arise with the integration of playout management into a multimedia database system in more detail.

System Architecture

As a consequence of the requirements to a multimedia system architecture we see the necessity of a distributed multi-user system but regard a fully distributed system to be too complex. The problems related to the handling of media objects and to multimedia document management, to media transport in a distributed system as well as to playout management are already complex enough. Therefore, we assume in the framework of our investigations a client/server architecture in which a central database server undertakes the data management tasks. Central management of data relieves the multimedia database management system from a controlled replication of data in the distributed system and preserving the consistency, respectively. The database system tasks of data transport and playout management must be efficiently partitioned in the client/server system to offer fast and comprehensive database support for multimedia applications. We propose to leave all presentation tasks to the clients. In this case the presentation components must have fast access to both multimedia documents and to the involved discrete and continuous media

data which are managed at a possibly remote server. This calls for client/server connections suitable for fast delivery of data streams.

Single Medium Presentation

The realization of presentation operations of the different media types is one of the tasks of a multimedia playout management. The playout management must load the media data, possibly over a slow network, from the database server site to the presentation node. In a distributed system the *end-to-end-delay* for the transmission of media data denotes the time difference between the request for media data at the presentation node and the point in time of delivery of the data to the presentation node. This delay must be taken into account by the playout management when media data has to be presented at a certain point in time. For media data of high volume, additionally, the problem may arise that the data cannot be entirely transported to the presentation node as this would exceed local memory. A continuous communication protocol must solve this problem and must offer continuous data transport throughout the entire presentation of media objects. The playout management must also handle data delivery deficiencies that may occur with unreliable and continuous data transport in a distributed system. This can be achieved by an adaptation of the presentation to the available presentation quality [31, 30]. The presentation of a continuous media object includes the enforcement of intra-media synchronization, that is, to achieve a continuous presentation with no or minimal perceptible discontinuities. Figure 2 illustrates the timing a single medium presentation needs with regard to end-to-end delay and other preparation tasks before the actual presentation can be started.

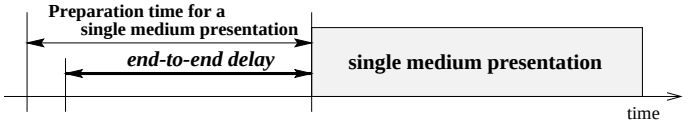


Figure 2 Preparation time and end-to-end delay for a medium presentation

Presentation of Multimedia Compositions

As a multimedia composition is a combination of presentation objects, playout management has to handle the combined presentation of the constituents of the presentation of multimedia compositions. The presentation of multimedia compositions, however, comprises far more tasks than just the single media presentations of the presentation objects involved.

Prior to a multimedia presentation the respective document must be retrieved from the database server. On the basis of the document the playout management reproduces the course of the multimedia composition. One of the most demanding problems of the presentation of a multimedia document is to meet the temporal requirements specified within a multimedia document. This includes the enforcement of the intra-media and inter-media synchronization specifications given by the multimedia document. Concepts have to be elaborated to ensure that the media objects are presented at the right time and in the right sequence, and to avoid discontinuities. For example, the preparation time for the presentation objects of a multimedia presentation can take more time than just the sum of the preparation times of the involved presentation objects. The delivery capacity for the data delivery of objects presented in parallel is limited by the network. Playout management must handle the resource management of the involved input and output devices.

Another important task of playout management is the handling of user interaction. The user may influence ongoing presentations by interactions, e.g. standard interactions or user-defined interactions, that are part of the preorchestrated multimedia document. User interaction can call for data delivery if a decision action causes the fetching of media data from the database server.

In order to fulfill the requirement to handle multimedia documents as a single new media type the presentation must appear as if only one time-dependent medium was presented to the user rather than a collection of several ones. Standard interactions should affect multimedia presentations as if one would operate on a single time-dependent medium. The management and cooperation of the collection of single medium presentations must be transparent to the consumer of the presentation.

As a fixed, predefined presentation quality cannot be guaranteed for single medium presentations all the time in general, the playout management must employ strategies also for the adaptation of the multimedia presentation quality to deviations from the quality specified in the presented multimedia document [31, 32, 30].

3 A MULTIMEDIA-DBMS PROTOTYPE

In this section, we present our approach towards the development of a multimedia database management system undertaken in the AMOS project (Active

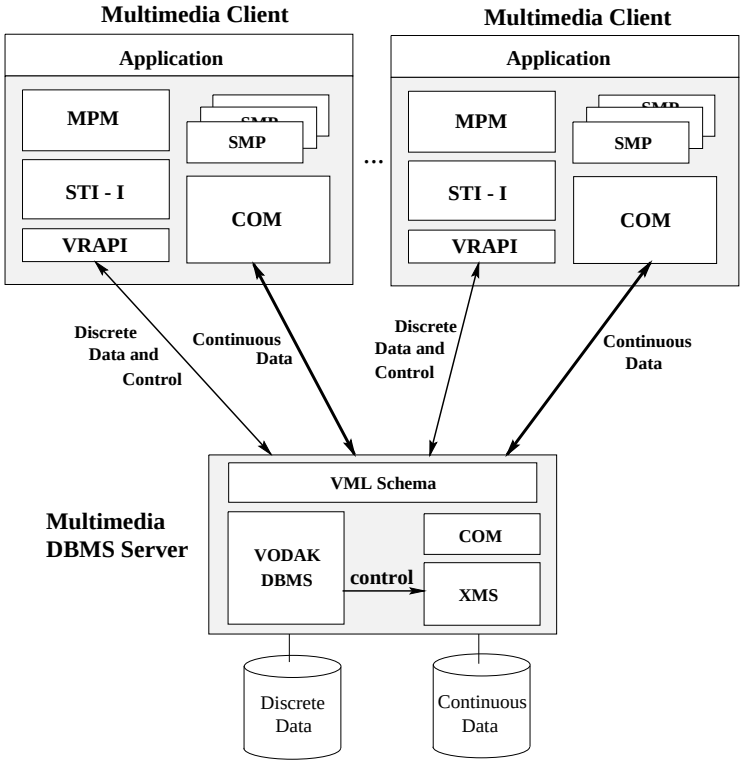
Media Object Stores) at GMD-IPSI. The goal of our development effort is to prototype a multimedia database management system which allows for the integrated management of multimedia data, i.e., integrated storage, retrieval, management, and presentation of multimedia data. Herein, we focus on the modeling and presentation of multimedia compositions in the framework of the new database system service with which we offer *presentation independence* to multimedia applications. We have been developing the object-oriented database management system VODAK which is a distributed database management system based on a client/server architecture. In a first development step we have extended the VODAK Modeling Language (VML) [10, 18, 20] to support media types. We currently extend the database management system with respect to modeling, management, manipulation, and presentation of multimedia compositions at the clients in a distributed environment. During the development process the AMOS system is already used in prototypes and applications ([20, 25, 33, 34], see also section 4). Thus, our concepts and implementations are based on real experiences with multimedia database system applications and they are constantly evaluated against user requirements.

In the following, we first address the architectural issues including the overall functionality of the system. Second, we focus on the functionality of the database server. Third, we discuss details on how to realize presentation support by a database client.

3.1 Architectural Issues

The overall goal of providing presentation independence to applications means that applications become independent of the structures used to represent a multimedia presentation. Hence, details on the modeling of multimedia compositions as well as the concrete execution of the presentation of such compositions have to be hidden from the application programs. In our approach we assign to the database server all the functionality related to the modeling and the management of multimedia data and their composition. Functionality related to the concrete execution of multimedia presentations is assigned to the database system clients.

Figure 3 shows the AMOS multimedia database system architecture, the assignment of tasks to the individual components of the system, and the data flow between multimedia database management system server and the multimedia clients.



Multimedia Client		MMDBMS- Server	
STI - I	**Space-Time-Interaction-Interpreter** Management of time and layout Handling of interactions	**VML Schema**	**Interface to Database** Access to objects, attributes and behaviour
MPM	**Multimedia Playout Manager** Management of SMPs	**VODAK DBMS**	**Object-Oriented DBMS** Management of discrete and continuous data
SMP	**Single Medium Presenter** Presentation of single media (audio, video, animation, etc.)	**COM**	**Continuous Object Manager** Delivery of continuous data
VRAPI	**VODAK Remote API** Delivery of discrete data	**XMS**	**Media specific Server** Storage of continuous data

Figure 3 AMOS multimedia database management system architecture

Multimedia Database Server

The central multimedia database server forms the core of our architecture. It manages the data in the database and controls concurrent access of the multimedia clients to a database. An important aspect of the server environment is the subdivision of the data management tasks into the management of conventional data and the management of continuous data. The VODAK database management system is employed for the management of discrete data. For the efficient management of multimedia data which are of possibly high data volume the architecture offers additional External Media Servers (XMS). These servers manage multimedia data on external storage media. The control over the XMS, however, lies at the VODAK database management system server. The XMS offer only transaction management for concurrent read and write access to external storage media. The Continuous Object Manager (COM) at the server accesses continuous data via the XMS and provides support for multiuser access to continuous data [22]. Access to both discrete and continuous media objects is provided by the methods specified in a VML-schema. The schema represents the objects, their attributes and behaviour accessible by an application. Given this functionality, the database server provides for the modeling and management of multimedia compositions by means of multimedia documents. In addition, the server provides the mapping of multimedia documents into an internal format called *presentation plan* encoding the concrete presentation of a multimedia document to be executed by a database client.

Multimedia Clients

The functionality of presenting and recording multimedia compositions is assigned to the database system clients. Restricting the client's functionality to presentation tasks increases portability of such multimedia clients for heterogeneous platforms.

A Multimedia Playout Manager (MPM) controls the presentation of multimedia components at the client. For each single media object involved in a presentation the MPM activates a Single Medium Presenter (SMP) which presents the corresponding media object at the designated device. The interface between SMPs and MPM is common to all SMPs. This allows to extend the client with appropriate SMPs for additional media types or according to the needs of an application.

The Space-Time-Interaction-Interpreter (STI-Interpreter) drives the MPM and is responsible for the fetching and the interpretation of presentation plans gener-

ated by the server, the temporal control of interactive multimedia presentations, and the handling of user interactions.

Client/Server Communication

Multimedia clients access the database via the VML-schema only. The data flow that arises from a client's request may be flow of control, flow of continuous media, or flow of discrete data. As a client does not have data management facilities suitable mechanisms are needed to ensure save, fast, and concurrent access to discrete and continuous media objects in the database. The separate management of discrete and continuous media objects at the multimedia database system server is also reflected in the separate streams for discrete and continuous data as can be seen from Figure 3.

The access of multimedia clients to discrete media objects is realized with the VODAK Remote Application Interface (VRAPI). After a connection between a multimedia client and the multimedia database system server has been established the client uses the VRAPI to invoke methods of objects stored in the VODAK database and to receive the respective results. With the VRAPI the full functionality of the object-oriented database system is at the client's disposal.

In the course of a presentation a multimedia client must also have fast access to continuous media objects. As some of these media objects are of large size it is not possible to entirely transport the data from the server to the client at once. The access to continuous media objects in smaller units such as frames and the continuous transport of these units to the client by the continuous object management allows the clients to have access to continuous media objects of high data volume in the course of a presentation. In case a client requests continuous data the server-COM is engaged with the delivery of continuous data. The client-COM serves the corresponding purpose of continuous data transport at the client and frees client applications from special treatment of requests to continuous data. During a multimedia presentation the continuous data needed by the SMPs are delivered by the COM of the client in cooperation with the server-COM. In our current prototypical implementation the communication between the COM modules at the server and at the clients is based on an ATM network protocol.

In the subsequent sections we discuss details on the modeling of multimedia compositions by means of multimedia documents, the internal format of presentation plans, the generation of presentation plans from multimedia documents,

and the interpretation of presentation plans for the purpose of running interactive end-user presentations.

3.2 From Multimedia Documents to Presentation Plans

Based on the architecture explained before we propose a concept for the management and presentation of multimedia documents that is of great flexibility and openness in two directions. First, it allows to store and to present multimedia documents in different formats without any changes to the client/server environment. Second, it increases the portability of the clients as it keeps them independent from the specific format of the multimedia document. In the following we will discuss these concepts to handle multimedia documents in detail.

The VODAK database management system provides powerful modeling capabilities that are sufficient to build up complex document structures, as they are necessary to represent structured documents. This has been proven in the case of SGML documents [5, 3, 1, 4]. Based on the VODAK modeling language (VML) [10, 18] a suitable representation for any structured document model can be found, even if it includes multimedia components. As a consequence, we can offer several models for multimedia documents at the server, e.g., for HyTime [23], MHEG [15], object composition petri nets (OCPN) [19], or for other application specific models.

Besides the modeling of multimedia documents their presentation to the end-user must also be addressed. In fact, most multimedia document models specify how the structures should be interpreted for presentation. Especially HyTime and MHEG specify presentation engines which accomplish these tasks. Our approach does not regard these presentation engines as monolithical entities but splits them into two parts that are characterized as follows:

- The first part of the engine processes the multimedia document structure and extracts all presentation relevant information. The actions that are necessary to accomplish this task depend on the document model and must be provided with every multimedia document model. We assume that this part of the presentation engine is implemented as part of the database schema and, therefore, is located at the server.

- The second part of the presentation engine handles presentation tasks like media playback and user interaction. These tasks are independent from the actual document model and can be provided for all document models in a generic way. As multimedia playback and interaction handling are time critical functions, we decided to locate the responsible components of the presentation engine close to the end-user at the client.

The communication between the client part and server part of the presentation engine calls for an interface that enables the server to delegate presentation tasks to the client and allows the client to return user interactions and information about the state of the ongoing presentation to the server. For this purpose we propose a common, linearized representation of multimedia presentations, which we call *presentation plan*. It is designed to express all presentation relevant aspects of a multimedia document and to be evaluated efficiently by the client components. User interactions that cannot be handled by the client components autonomously are communicated back to the server in order to produce an appropriate reaction.

As mentioned before, our approach is characterized by assigning those aspects to the server that depend on the multimedia document model. In particular this includes the preprocessing of presentation information by database methods. In the following we refer to the part of the server schema that is responsible for this processing as the *document mapper*. As shown in Figure 4 the implementor of a multimedia document model has to provide a model specific mapper. This mapper translates the presentation information contained in the multimedia document into a presentation plan. An example for an analogous mapping is the representation of various high level user interface models like Motif or Open Look based on the X-protocol.

Before we look into the details of multimedia modeling and interpretation, we briefly outline the general scheme of mapping multimedia documents to presentation plans:

The database client, in particular the Space-Time-Interaction-Interpreter (STI-Interpreter), initiates the mapping process at the server by requesting the initialization of a mapper module for the identified multimedia document. The activated mapper is always specific to the document model of the identified multimedia document. For example, if the multimedia document is a HyTime document, the HyTime mapper will be activated. The mapper component then processes the multimedia document and generates a concrete presentation plan. This presentation plan is delivered to the database client, which executes the

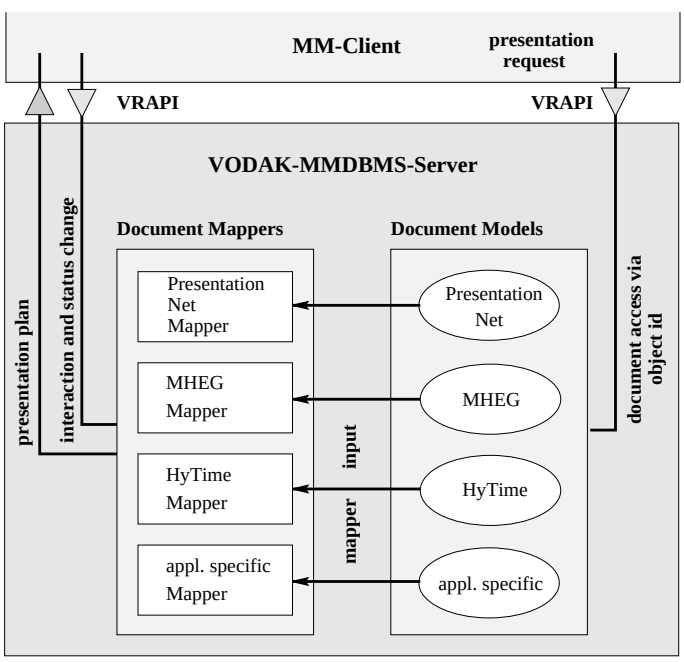


Figure 4 Modeling and preprocessing of multimedia documents at the server

plan for the purpose of an end-user presentation. The client may then request the generation of a subsequent presentation plan depending on user interactions during the presentation.

More details will be given after the discussion of the structure of multimedia documents and presentation plans.

Modeling of Multimedia Documents

Looking at the document models mentioned before, we find that HyTime is a quite complex document model of which we could only implement a subset with reasonable effort. Furthermore, HyTime does not provide a model for interaction. Hence, we would have to design a useful interaction concept for HyTime in addition, if we wanted to have a comprehensive solution. Alternatively, MHEG supports interaction but omits the logical document structure. This is sufficient to exchange multimedia documents in their final form for presentation

but provides no useful basis for editing. Therefore, we decided to illustrate our approach with an extended model of object composition petri nets. OCPNs provide the basis for a reasonable multimedia document model that fulfills several requirements: They can be implemented with foreseeable effort, provide a structure that allows editing and can be easily extended with additional elements that specify interaction and synchronization. Even half open time intervals can be introduced into the model to describe presentations of infinite duration. In the following we denote by *presentation nets* this extended model of OCPN.

Figure 5 shows a sample multimedia presentation that is modeled as a presentation net. Compared to OCPN the notation has been changed slightly. Places are now represented by rectangular boxes instead of circles, and transitions are represented by circles instead of vertical bars. According to the modeling described below we call the places *media nodes* and the transitions *synchronization nodes*. We made several extensions to the original OCPN model. These extensions allow for the specification of media presentations with infinite duration, the description of interaction elements, and the integration of inter-media synchronization.

Figure 6 shows the classes we use to model presentation nets in the database. Every media or synchronization node is an instance of the class `PNNode`. In every node the predecessor and successor nodes are referenced to represent the structure of the network. Incoming tokens are collected in a set. The meaning of `PNNode` objects is generally determined by the content objects they refer to.

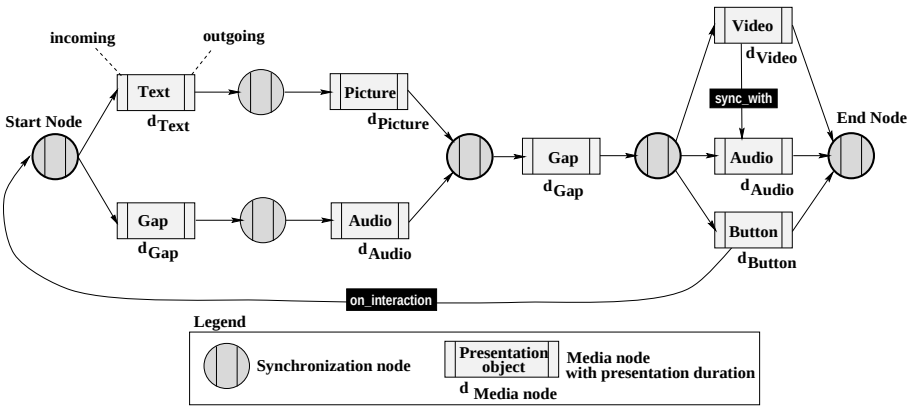


Figure 5 A sample presentation modeled as a presentation net

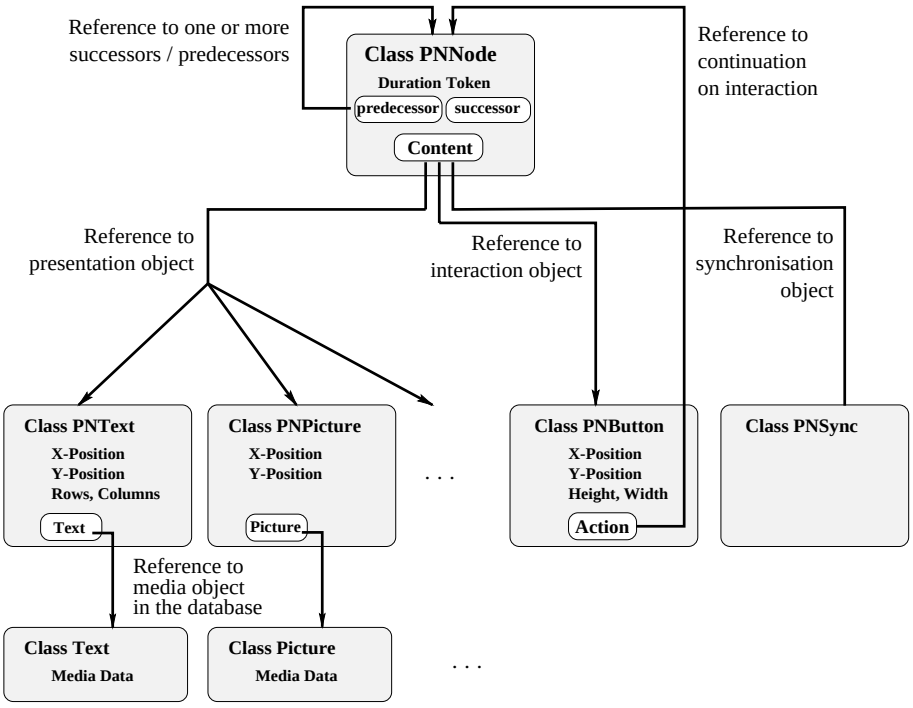


Figure 6 Classes modeling presentation nets in the database

With content objects like `PNPicture`, `PNText`, `PNAudio`, etc. the node acts as a media node. A content object contains the presentation parameters for a media object and a reference to the media object to be presented. Therefore, it is possible to use the same media object repeatedly with different presentation parameters without replicating the media data. The presentation parameters describe aspects like duration, position, and scaling on the screen and allow to specify a subrange of the referenced media object. A content object `PNGap` denotes a special 'media node' that is necessary in order to represent arbitrary temporal relationships. These nodes are interpreted like media nodes but produce no visible or audible result. Their only effect is to delay subsequent nodes by the specified duration. The content object `PNSync` denotes a synchronization node. As synchronization nodes are not presented they need no additional presentation parameters. So far the objects mentioned could be used to build up a network that is equivalent to the OCPN model.

Objects of the class `PNButton` extend the OCPN model by user interactions. Every `PNButton` object contains a reference to the start node of a presentation net. As long as a `PNButton` object is presented, the user has the possibility to select it on the display. If this happens the presentation is stopped immediately, all tokens are removed from the presentation net, and the presentation is restarted at the referenced start node. In contrast to OCPN we support media nodes with infinite duration. The availability of user-defined interaction elements like buttons and a standard operating panel offered by the MPM guarantee that presentations with infinite duration can always be terminated by user interaction.

To simplify the notation of complex presentation nets *sub-presentation nodes* supplement the node types mentioned before. Sub-presentations refer to complete presentation nets and are substituted by their associated presentation net before the execution of the presentation. One further extension are synchronization links that may be used between two media nodes representing continuous media objects. The presence of synchronization links allows the specification of inter-media synchronization for continuous media that are presented in parallel. In our environment presentation nets can be edited with a generic network editor [37] that has been configured with the definition of presentation nets (see Figure 7). With typed sub-presentations and the constraint checking features of the editor, design rules are supervised while a presentation net is edited.

Presentation Plans

As mentioned before, the database server maps different types of multimedia documents to a common, internal format, i.e., presentation plans, which are executed by the clients. A presentation plan captures all information necessary for the multimedia client to present the multimedia document. The mapping ensures that the temporal and spatial course of the multimedia composition as well as interaction, synchronization, and QoS specifications of a multimedia document are preserved in the presentation plan.

A presentation plan is a linearized, event-based representation of a multimedia presentation that describes the course of a multimedia presentation. As it is sufficient to store each point in time at which 'something happens' the course of a multimedia presentation is captured by *events*. For a time interval in which, e.g., a video object is to be presented, only the start event and the stop event are stored in the presentation plan. For each event the presentation plan contains additional information about the media object involved and its presentation data. If a multimedia document specifies alterations to the presentation of a

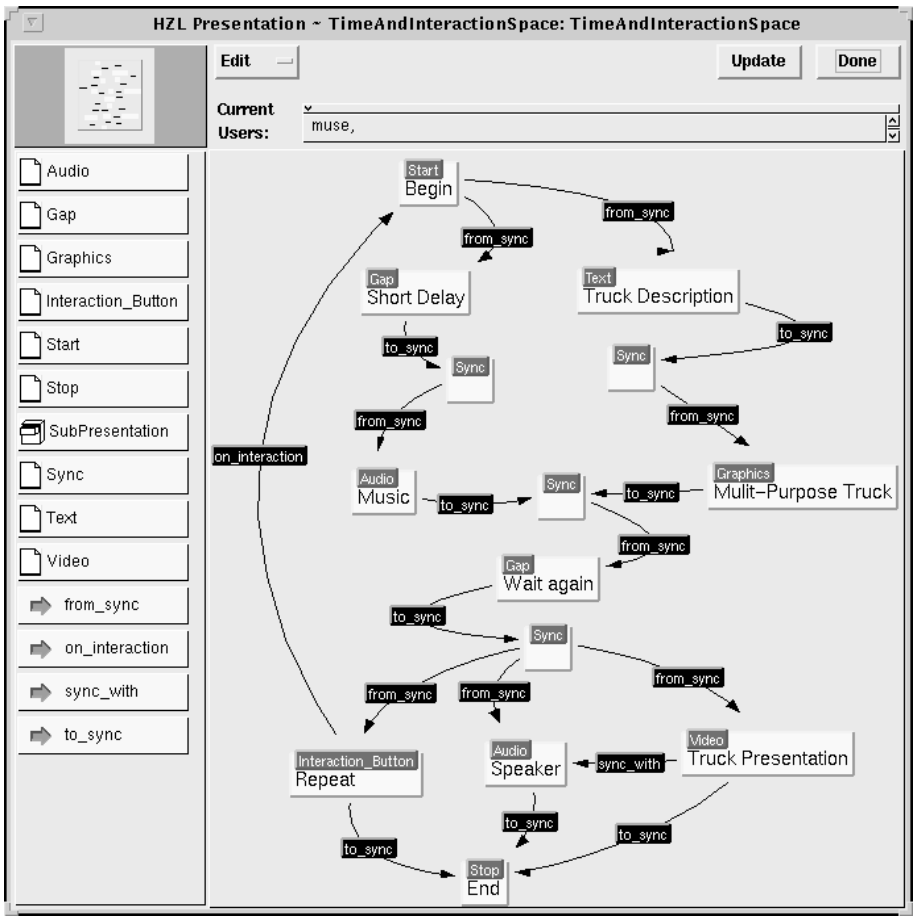


Figure 7 A generic network editor for presentation nets

medium the presentation plan contains additional events for each alteration. All events are stored in a linear list in ascending order corresponding to their occurrence in the multimedia presentation. Time is represented as the temporal distance between the events without referencing a global origin. In the case of two parallel events this temporal distance has a value of 0. Figure 8 shows how the temporal course of a presentation is captured in a presentation plan.

The list elements of a presentation plan are of type `MultimediaEvent` (see Figure 9) and each element stores a point in time, the reference to the respec-

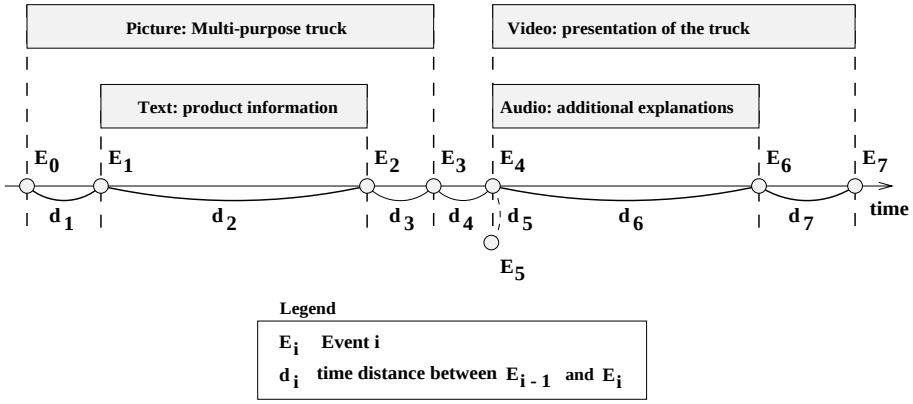


Figure 8 Representation of events in a presentation plan

tive media object, and the necessary presentation data. The attributes of a `MultimediaEvent` contain the information necessary to start, change, or stop a single medium presentation. This includes quality of service parameters and synchronization relationships. All events are represented in a common structure for simplicity.

Bringing Presentation Networks to Life

In the previous paragraphs we described how presentation nets can be represented using an object-oriented database system and how presentation plans look like. Now, we will give some more details on the process of mapping presentation nets to presentation plans.

In addition to the data structures described so far, we implemented a number of database methods. One part of these methods provides editing functionality and supports the network editor used for the design of presentation nets. The other part implements the mapper component of the server which maps presentation nets to presentation plans. For every client which requests the presentation of a multimedia document that is modeled as a presentation net an object of the class `PNMapper` is created. Mapper objects generally offer two basic methods to control the translation of multimedia documents into presentation plans:

```
DATATYPE   MultimediaEvent = [
  EventType:                 INT,     // event type:
                                      //   "start", "stop", "alteration"
  MediumType:                INT,     // medium type
  MediumOID:                 OID,     // medium object id
  PropertyName:              STRING,  // storage
  ReferenceNumber:           INT,     // reference number
  RelativeEventTime:         INT,     // relative time distance
                                      //   to predecessor
  IntraMediumStartTime:      INT,     // details of a
  IntraMediumStopTime:       INT,     //   continuous media object
  IntraMediumStartReference: INT,     // details of a
  IntraMediumStopReference:  INT,     //   discrete media object
  TextColumns:               INT,     // presentation of columns
  TextRows:                  INT,     // presentation of rows
  Width:                     INT,     // width
  Height:                    INT,     // height
  X_Coordinate:              INT,     // vertical screen position
  Y_Coordinate:              INT,     // horizontal screen position
  Z_Coordinate:              INT,     // overlapping
  BasicVolume:               INT,     // basic volume
  BasicSpeed:                INT,     // basic speed

  ButtonAction:              OID,     // oid of the continuation plan
                                      //   in case of interaction

  AlterationType:            INT,     // type of alteration
  AlterationValue:           INT,     // new value

  SynchGroupId:              INT,     // synchronization group
  IsMaster:                  BOOL,    // synchronization role ];
```

Figure 9 Data type for events

1. The method `init_start(object_identifier)` initializes a specific mapper for the multimedia document identified by the parameter of the method.

   If the object identifier refers to a presentation net, an instance of `PNMapper` is initialized. For presentation nets, the object identifier refers always to an entry point of a presentation net, i.e., either to the start node of the net, or to some other node in the net which serves as the starting point for the subsequent generation of a presentation plan. The mapper for presentation nets implements the initialization by inserting a token into the entry node of the presentation net.

   The mapper for any other document model would have to do its own appropriate initialization of the mapping process at this point in time.

2. The method `generate()` generates the presentation plan. This presentation plan contains the information necessary to reproduce spatial and temporal layout, possible interactions, and synchronization as specified by the multimedia document.

   In the case of presentation nets, the entries in the presentation plan are generated by moving tokens through the presentation net. While the presentation net is traversed the time intervals assigned to media and pause nodes are evaluated in order to generate absolute time intervals (see Figure 10) in the presentation plan. The mapper for presentation nets does not pause for the presentation duration of a media node, but only enters the duration into the generated plan.

   A mapper for other document models would have to perform its own kind of mapping at this point. To build up a presentation plan all mappers can rely on the common interface of a class `PresentationPlan`.

There are several possible solutions *when* the generation of a presentation plan could be performed. One possibility is to generate all presentation plans prior to their presentation so that every multimedia document can be presented immediately when requested by the user. Such a solution has the disadvantage of excluding multimedia documents that are created dynamically on a user's request, for example, when a user query results in a multimedia document composed on the fly. Another possibility is the generation of the presentation plan on a user's request, which may delay the start of a presentation and might not make sense for static documents.

In the case of presentation nets we decided to combine both approaches and to generate the presentation plan for a presentation net when a user requests the presentation. We store the plan, however, as a database object that is indexed with the start conditions given by the initialization method. This database object is invalidated when the document changes or when no presentation of the multimedia document is requested for a given period of time.

Another issue is *how* presentation plans are generated. A straight forward strategy is to compute always the complete plan for an entire multimedia document. A useful alternative might be to incrementally compute the presentation plan depending on the progress of the presentation, the user interactions taking place, or the system load at the client and the server. For example, after generating a presentation plan that represents a given duration of the presentation the mapping could be interrupted and the generated plan fragment could be returned to the client. While this fragment is presented at the client the subsequent fragment could be prepared by the mapper and delivered to the client on

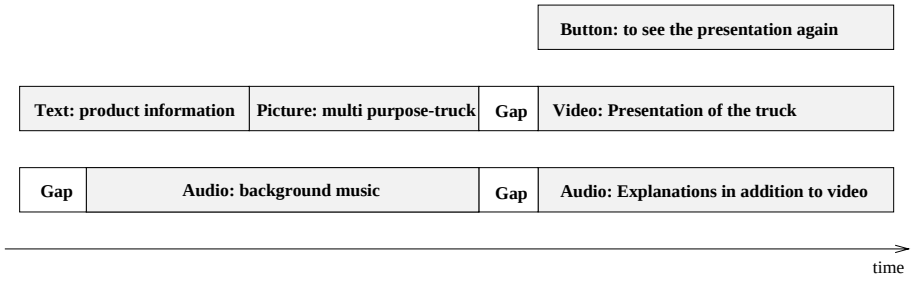


Figure 10 Time-line representation of the presentation example

demand. This strategy becomes necessary as soon as a multimedia document model allows cyclic structures. Applying the straight forward strategy to such structures would lead to an endless loop when the mapper tries to generate a presentation plan that represents the entire document.

User interactions can be supported by the mapper without any changes to the interface described so far. If we assume that a user interaction always determines which course the presentation takes in the future the methods `init_start` and `generate` are sufficient. With the presentation plan the client receives the object identifier determining the continuation of the presentation that follows a potential user interaction. If the respective user interaction takes place, the client asks the server for the mapping of the document that is referenced by the object identifier.

In the case of presentation nets, if a button 're-start from the beginning' is offered to the user during a presentation, the presentation plan includes the object identifier of the presentation net which has to be presented after the selection of the button. In the case that the user activates the button, the client just initiates the generation of a continuation via the methods `init_start` and `generate`.

To handle more complex user interactions than interactions via buttons the method `init_start` can be extended by a set of parameters. For example, a query statement could be entered by the user into a text input field. The text string could be passed to the mapper as a parameter of the initialization method, and could then be evaluated as part of the mapping process in order to determine the further presentation.

Mapping database structures to presentation plans is an approach that is very flexible and open for adaptations to specific application needs. It allows to present any kind of data stored in the database. The only requirement is to have available an appropriate mapper module which generates presentation plans using the common interface of the system class `PresentationPlan`. In fact, in our prototypical implementation we can integrate arbitrary mappers as part of an object-oriented database schema. This allows us to offer services like graphical database browsers or graphical query interfaces with little additional effort.

3.3 Visualization of Presentation Plans

After we discussed the structure of presentation plans and their generation from multimedia documents, we now look at the presentation tasks to be performed at the multimedia clients. Figure 11 shows the components of a multimedia client and the data streams between the client components and the database system server. The multimedia presentation tasks are split up into the actual presentation of the multimedia composition and the overall control of the presentation including time management and interaction handling.

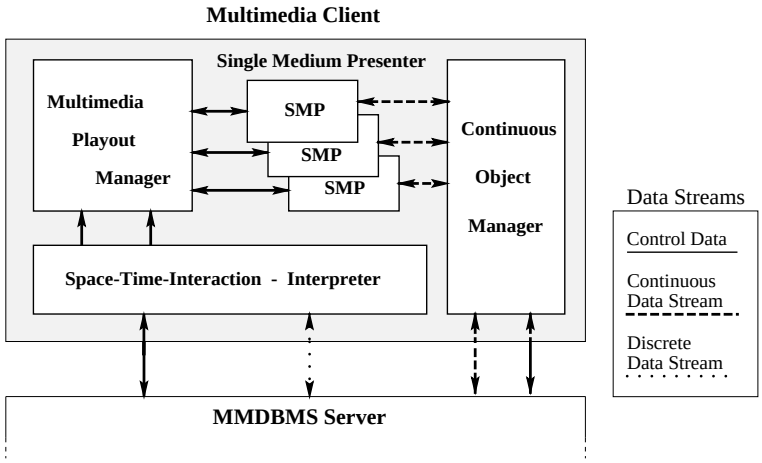


Figure 11 Database components of a multimedia client in the AMOS system

The Multimedia Playout Manager (MPM) forms the playback component of the client and allows for the presentation of different media objects. Part of the tasks of the MPM is to control input and output devices such as mouse, monitor,

and speakers. For each media object involved in the presentation the MPM creates and manages a Single Medium Presenter (SMP) which can prepare, start, suspend, resume, change, and stop the presentation of its associated single media object. The continuous media data for single medium presentations are delivered from the COM at the server via the COM at the client to the respective SMP. The Space-Time-Interaction-Interpreter (STI-Interpreter) is the connecting link between the presentation of media objects by a client and the management of media objects and multimedia documents by the database server. It drives the MPM and controls the course of a multimedia presentation. Prior to a presentation the STI-Interpreter requests the respective presentation plan from the database server and forwards individual presentation instructions to the MPM.

Figure 12 shows the partitioning of the playout management tasks into control mechanisms, i.e., the interpretation of presentation information, the time management, and the complex interaction handling, carried out by the STI-Interpreter, and presentation mechanisms, i.e., the actual playback of single media objects including control of input and output devices carried out by the MPM in cooperation with the SMPs.

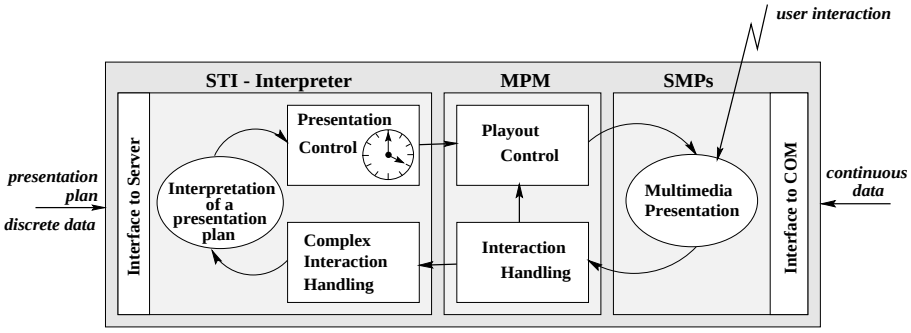


Figure 12 Cooperation of STI-Interpreter, MPM, and SMPs

Once a presentation plan is received by the client the STI-Interpreter starts the interpretation of the plan. As a result of the interpretation it generates detailed instructions for the playback of the presentation. The time management is handled by the presentation control of the STI-Interpreter, which forwards presentation instructions to the MPM according to the course of the presentation. The MPM immediately executes the instructions received from the STI-Interpreter with best effort supported by the SMPs and the continuous data delivery of the COM. As the MPM presents also media objects for interaction such as buttons

it forwards user interactions, if necessary, to the STI-Interpreter. In case of a complex interaction the STI-Interpreter initiates the interaction handling. It enforces the respective action, e.g., pausing of the entire multimedia presentation in cooperation with the MPM and the SMPs.

In the following, we discuss the tasks of interpreting a presentation plan, managing time, and controlling a multimedia presentation performed by the STI-Interpreter. We also look at the tasks of running a presentation by the MPM and the interaction handling in more detail.

Interpretation of Presentation Plans

The STI-Interpreter sequentially processes a presentation plan that contains the events of the multimedia presentation in the order of their occurrence in the presentation. Figure 13 illustrates how the temporal course of a multimedia presentation is reproduced.

In a first step, each event in the presentation plan is processed by the STI-Interpreter. For each event one *prepare* instruction and one *trigger* instruction are generated for the MPM but not yet forwarded to the MPM. The logical sequence of instructions generated is mapped to a real-time schedule. This schedule reflects the detailed timing for the preparation and execution of the actions associated with the events. In a second step, the STI-Interpreter forwards the prepare instructions and the trigger instructions according to the schedule to the MPM. The instructions finally cause the MPM to control the SMPs accordingly. The STI-Interpreter performs these two steps in parallel. The mapping of the logical sequence of events given in the presentation plan onto a real-time axis of the presentation is illustrated in Figure 13.

Time Management

The basic idea behind the time management is the separation of the preparation tasks for single medium presentations from the actual triggering of the presentation. Prior to the start of a media object, e.g., a video, the respective data have to be loaded from the database and a presentation device, e.g., a window, has to be prepared. This can be done in a preparation phase prior to the actual presentation. We intend to put as much work load as possible into the preparation phase in order to allow for fast and efficient execution of a single medium presentation when triggered. The scheduling of preparation instructions takes the dynamically changing end-to-end delays in a distributed

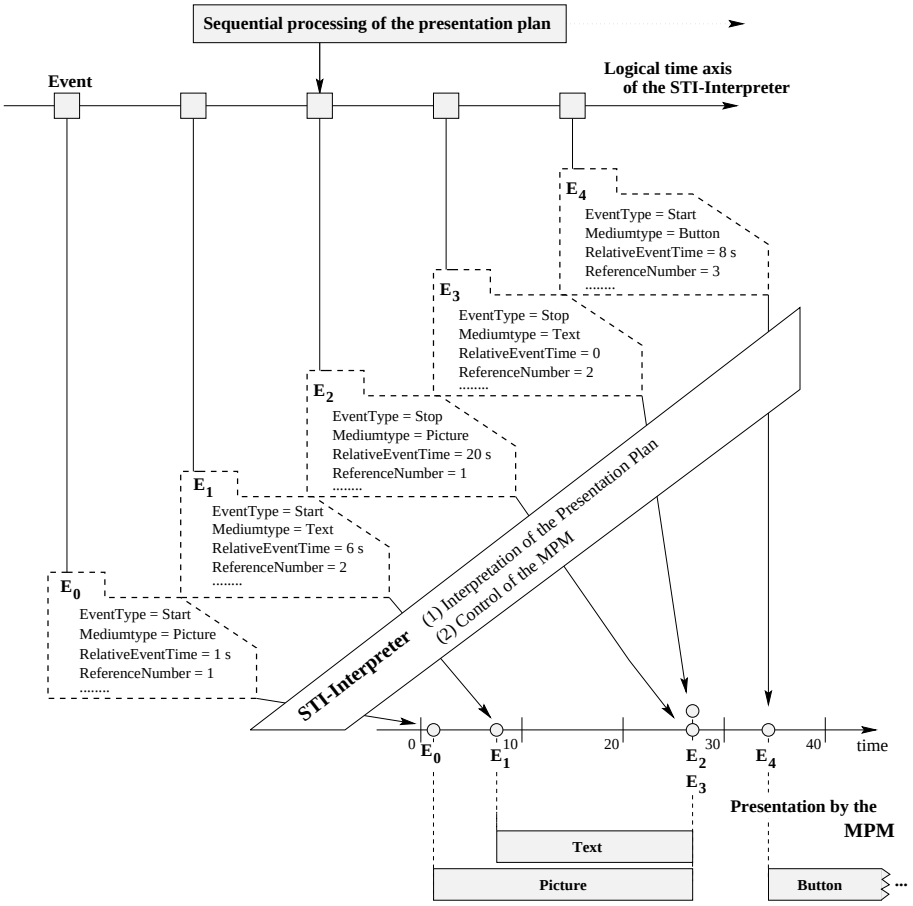


Figure 13 Mapping of the logical sequence of events to a real-time axis

environment into account. Figure 14 illustrates the *prepare-and-trigger* concept we apply.

On receipt of a prepare instruction the MPM creates or activates, respectively, an SMP which executes the operations implementing the preparation of the event. If two events occur at the same point in time they are prepared by two different instructions, but triggered by only one instruction. This ensures that events scheduled for the same point in time can be triggered in parallel. This concept can be seen in Figure 15 that shows two events, namely the end of the

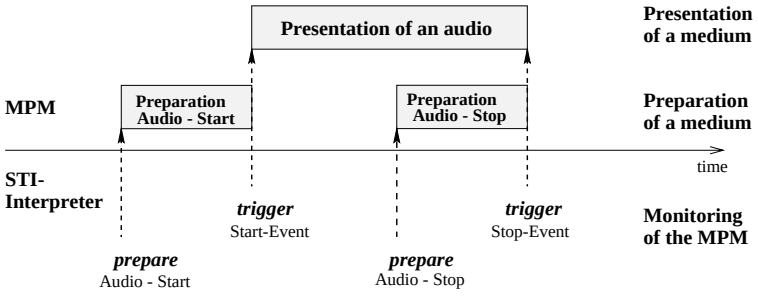


Figure 14 Reproduction of the temporal course of a presentation

presentation of an audio and the beginning of the presentation of a picture, that are triggered in parallel.

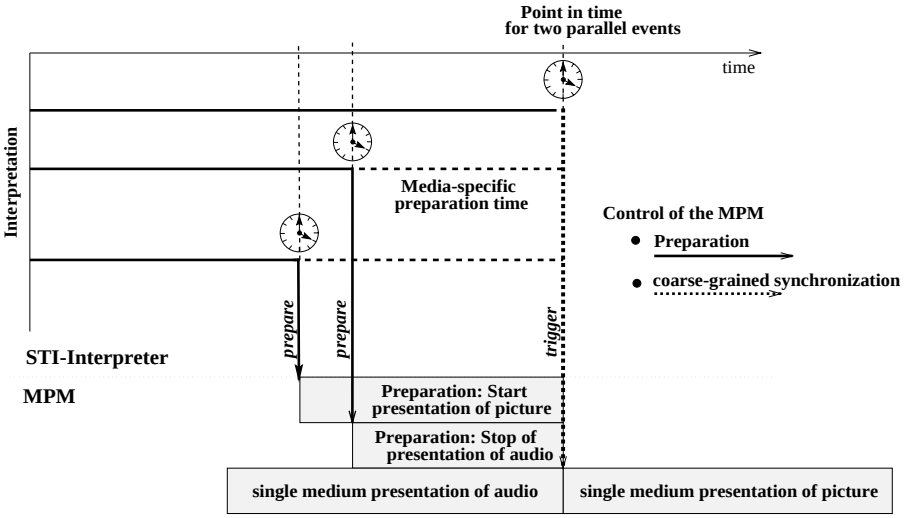


Figure 15 Triggering of parallel events by the STI-Interpreter

Playback of Presentations

The MPM manages a collection of SMPs each of them in charge of presenting a media object. For each instruction to prepare the start of a single medium presentation the MPM creates a media specific SMP. The latter is an independent presentation thread which carries out all single medium presentation tasks such

as start, stop, pause, and resume of the presentation as well as alterations to the single medium presentation. During the preparation phase for the start of a single medium presentation the SMP is responsible for pre-loading the media data from the remote multimedia database system server. After having finished the preparation of an event the SMP waits for the triggering of the event. Once a trigger instruction for events is received by the MPM the respective SMPs execute the actions that were prepared beforehand. The SMPs for continuous single medium presentations autonomously realize the time-dependent presentation according to the specified presentation speed. User interaction is provided by specific SMPs which support, for example, the selection of a button, the input of text, or the recognition of speech input.

Synchronization Enforcement

During a presentation the system has to enforce *coarse-grained* as well as *fine-grained* synchronization. The STI-Interpreter is responsible for the coarse-grained synchronization and, hence, enforces the temporal modeling of the presentation plan, i.e., the scheduling of start and stop events. This is achieved by the proposed prepare-and-trigger concept for sequential and parallel events as illustrated in Figures 14 and 15.

The fine-grained synchronization concerns both inter-media and intra-media synchronization of continuous media objects and is enforced by the MPM and the SMPs. SMPs that present continuous media objects support intra-media synchronization. That is, they regularly control the continuity and the playback speed of the continuous single medium presentation. The MPM enforces the inter-media fine-grained synchronization. During a presentation the MPM compares the actual presentation quality regarding inter-media synchronization with the quality specified in the presentation plan on a regular basis. In case of quality deficiencies the MPM employs strategies for the correction of asynchrony such as dropping or duplication of data units [31, 32, 30].

Interaction Handling

The interaction handling includes the processing of *simple* and *complex* interactions. Simple interactions are interactions which can be handled by an SMP itself. For example, a user can change the volume of an audio, move or resize a picture on the screen, or unhook a continuous single medium presentation from the remainder of a presentation. The more functionality an SMP provides the more simple interactions can be offered to a user.

Complex interactions are those which cannot be handled by the SMPs themselves. Data on the occurrence of complex interactions are forwarded to the STI-Interpreter via the MPM for further processing (see also Figure 12). We distinguish two special types of complex interactions, *user-defined interactions* and *standard interactions*.

Examples for user-defined interactions are interactions which result in the change of the presentation plan for the remainder of a presentation because of a decision action such as 'continue at some other media node in the presentation net', or 'jump to a specific, user-defined position in the presentation'. This requires the server to generate and to deliver a new presentation plan. Figure 16 illustrates the affect of user-defined interactions on a presentation. Figure 16(a) shows the execution of some presentation without any user interaction. Figure 16(b) illustrates the affect of selecting buttons during the presentation. First, a user selects the button 'Repeat Trailer', which stops the current presentation and starts again with the section beginning with the trailer. Subsequently, the user selects the button 'Repeat scene 2', which causes the system to change the presentation plan again in order to start presentation again with the beginning of scene 2.

Standard interactions are default generic interactions offered by the MPM to the user for the purpose of controlling an entire presentation. Examples are interactions concerning an entire presentation, e.g., fast forward, reverse, or coloring of all constituent media objects. These interactions are not reflected in presentation plans. Figure 17 shows our current standard operating panel with the standard interactions start, stop, pause, resume, and fast forward. The available interactions depend on the actual state of the presentation. Some more standard interactions such as reverse, fast reverse, and random positioning in the multimedia presentation by means of a slider, will be added in the near future.

The processing of complex interactions comprises the following three steps:

1. Adjusting plan interpretation: Depending on the semantics of the interaction the status of the interpretation of a presentation plan is adjusted, i.e., the interpretation is stopped, or resumed with a new or updated presentation plan. For example, if an interaction to stop an entire presentation occured, the plan interpretation is stopped too. If the playback was changed to fast forward, the plan interpretation is continued with an updated presentation plan reflecting the new playback speed.

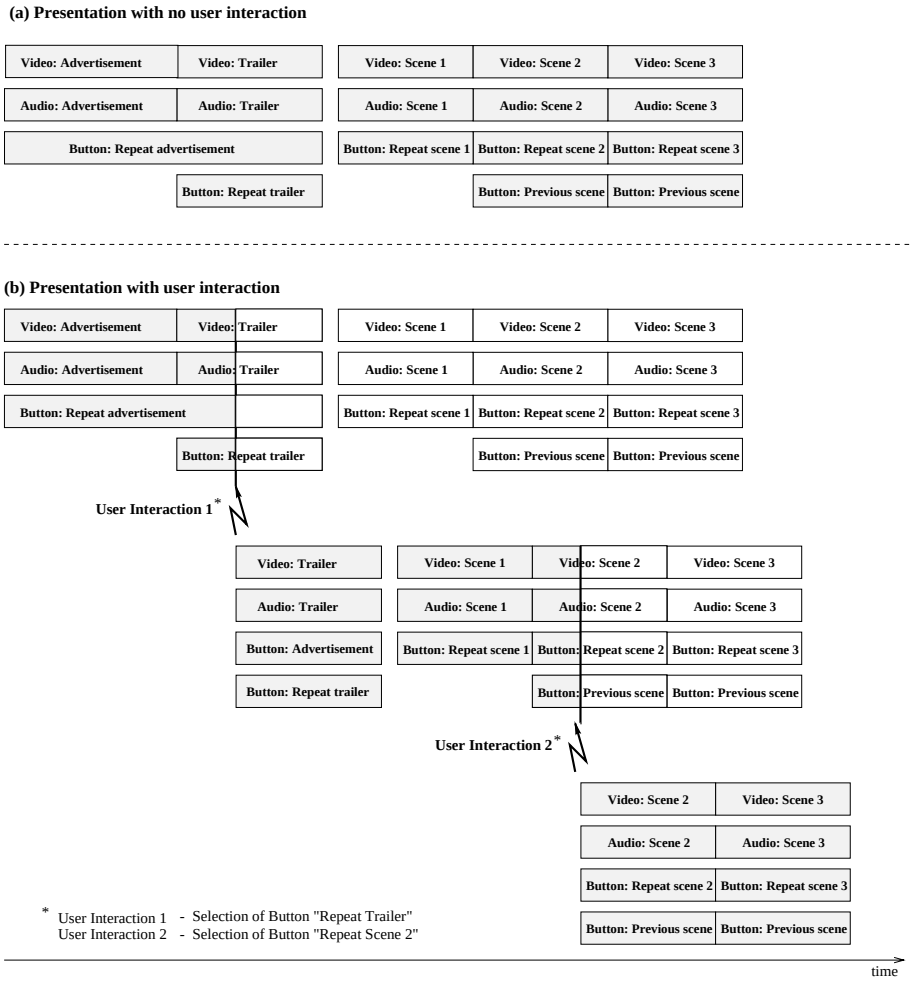


Figure 16 User defined Interactions

Figure 17 Standard operating panel

2. Adjusting control of MPM and time management: After adjusting the plan interpretation, the STI-Interpreter verifies the real-time schedule of instructions originally planned to be sent to the MPM. For each instruc-

tion scheduled the STI-Interpreter checks whether this instruction and its scheduling are still valid or have been invalidated by the interaction. For example, if an interaction to stop an entire presentation occured, all instructions scheduled for the future are invalidated. If an interaction requested to pause a presentation, the trigger instructions are invalidated, the scheduling of prepare instructions is kept and executed accordingly.

3. Adjusting the presentation: As a consequence to the interaction, the presentation has to be adjusted according to the semantics of the interaction. For example, if an interaction to stop an entire presentation occured, the MPM is instructed to stop the playback of each involved SMP. If an interaction to jump to another position within a presentation has been registered, the MPM is instructed to stop the ongoing playback of all SMPs and to continue the presentation at another position according to a new presentation plan.

4 APPLICATIONS

In this section we describe two reference applications which use the prototypical implementation of the concepts described so far. The first example shows how multimedia concepts and database management systems concepts can be used to support the design and evaluation of complex technical systems. The second example briefly describes a multimedia online service providing information on events. One could easily identify further application scenarios which involve the storage, processing, and retrieval of multimedia data. These include the quite natural and general application domain of multimedia document management and processing (e.g., [3, 33]), multimedia mail systems including voice-mail systems (e.g., [35, 34]), teleconferencing applications, and kiosk information or point-of-sale systems.

4.1 Product Documentation for a Systems Engineering Environment

At the Technical University of Darmstadt the research group MUSE[2] was formed to provide a generic, computer-based environment that will assist in

[2]MUSE is an acronym for **Mu**ltimedia-Supported **S**ystems **E**ngineering. The project is sponsored by the 'Deutsche Forschungsgemeinschaft' (German research foundation) under grant number He 1170/5-1,2,3 .

the process of modeling and validating complex technical systems. Several groups from different research directions in computer science and mechanical engineering contribute to the project. The modeling process supported by the MUSE tools results in virtual prototypes of technical systems to be designed. Simulations of operational states of technical systems based on these virtual prototypes are an important means for validation [8].

Our subproject is concerned with the development of a hypermedia database and desktop. These components serve as the organizational environment that allows to access and manage all data that is produced in the MUSE environment. Moreover, it integrates the various tools a systems engineer uses. The generic media support of our database system is used to store and to present raster images, drawings, videos and audio attachments which result from the design and testing of the developed product. An additional medium which occurs in this context is the recording and playback of simulation inputs and results. Recorded simulations can be presented in animated form and offer considerable more possibilities for user interaction than any other medium [7]. Composite media presentations are mainly used for documentation purposes. An example is the synchronized reproduction of a simulation result as a multimedia presentation. Several animated views of the model can be shown in several windows while spoken comments, which were recorded during the simulation, are played back. Figure 18 shows the presentation of a multimedia document concerning the rear wheel steering of a multi-purpose truck, which was activated from the hypermedia desktop of MUSE.

4.2 Multimedia Online Information Service

Due to substantial progress in integrating network technology and multimedia information systems specific multimedia teleservices are emerging. One specific example from this application domain is a multimedia database system supported archiving teleservice regarding actual events like art performances, cinema movies, tours of music groups, etc.. Our group at GMD-IPSI developed several prototypes illustrating the potential of this technology including a *Calendar of Event Teleservice* prototyped in the context of the GAMMA project [33]. It allows for the storage of multimedia event descriptions offered by content providers, for the search and retrieval of multimedia event descriptions in order to get details on an event, to pre-view a show or movie, or to order tickets. Figure 19 illustrates with a screen dump how a multimedia presentation driven by a multimedia database system client looks like for a pre-view of some circus show. At the bottom of the window one can see some standard inter-

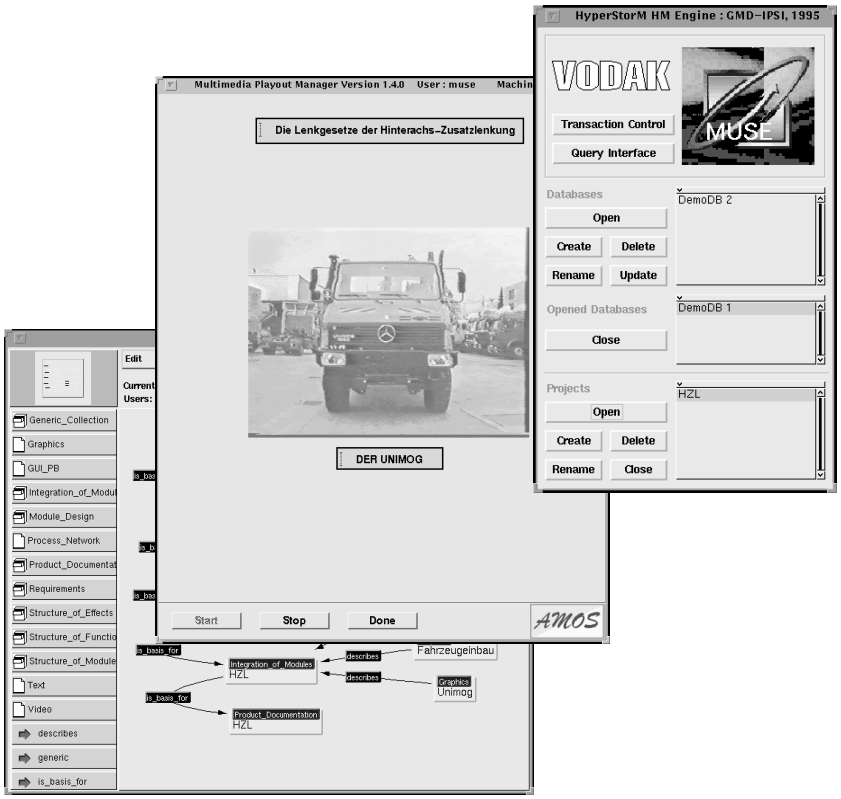


Figure 18 Presenting multimedia product information on the MUSE desktop

action buttons *Start*, *Stop*, *Pause*, *Resume*, and *Quit* driven by corresponding Single Medium Presenters (SMP). On the left side one can see an image (video frame) presented by another SMP, and to the right there are additional, user-defined interaction buttons which allow the user to choose between an English or French presentation of the event.

Similar, Figure 20 shows a snapshot of a re-play of a chess tournament. The players are presented by either small images or video clips, the moves of each player are encoded by text below the images, each individual move gets explained by a textual comment below the board as well as by an audio annotation, and the chess board animates the individual moves. The images, the text

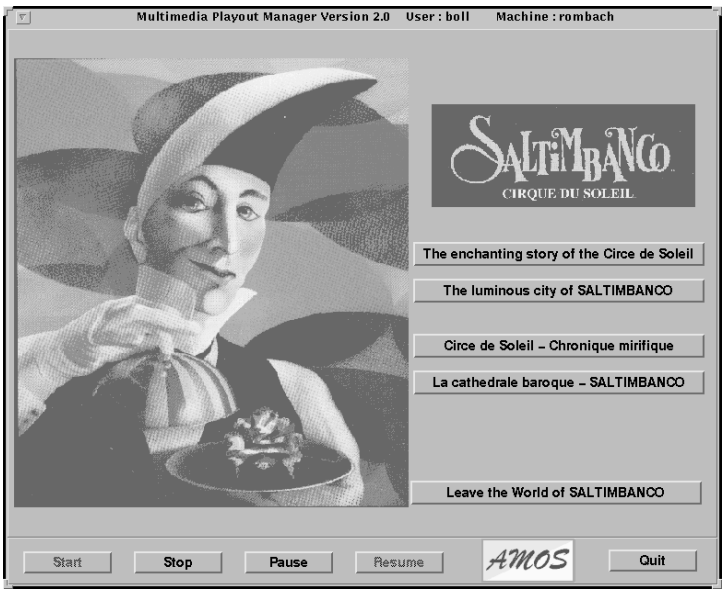


Figure 19 Snapshot of a multimedia presentation

windows, and the audio annotations each are driven by corresponding Single Medium Presenters. The chess board is a good example of a complex Single Medium Presenter which encodes the functionality of a chess board and the legal moves. The only information passed from the Multimedia Playout Manager (MPM) to that Single Medium Presenter are the encoded moves, but no visual information is passed through the system. The visual presentation is determined by the Chess Board Single Medium Presenter. With this example one can easily illustrate some benefits of our approach. For example, if somebody does not like the style of the chess board, he/she could ask the system to choose another Chess Board Single Medium Presenter which follows his/her preferred look and feel for the board. The only requirement for another Single Medium Presenter is that it follows the standardized interface between Single Medium Presenters and Multimedia Playout Manager. The possible choices could be pre-coded by the presentation designer within the presentation plan generated by the server, or the server could ask the end-user about his/her preferences before generating the presentation plan. In our current implementation, we do not support the latter option.

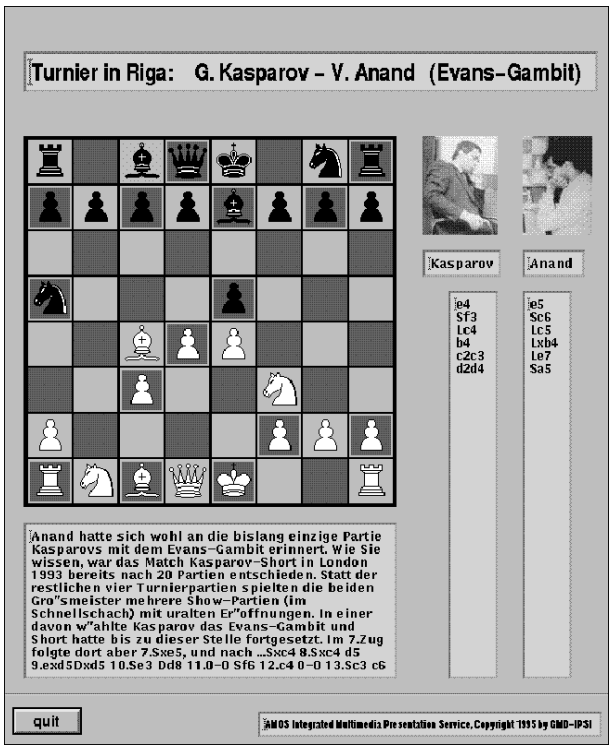


Figure 20 Snapshot of a multimedia presentation

5 CONCLUSION AND PERSPECTIVES

As we have seen from our discussion, extending database systems with specific services for the handling of multimedia data allows for an integrated support of multimedia data for applications. The benefits of our approach are

- the availability of a new database system service called *presentation independence* offered to applications in line with other database system services like data independence, consistent data management, multi-user support, etc.,
- the support for the modeling, storage, manipulation, and presentation of arbitrary multimedia compositions,

- the handling of multimedia compositions in an integrated way like the handling of other multimedia and conventional data, and
- the availability of a configurable presentation engine which comes with the database client, i.e, users and designers can chose appropriate SMPs from Single Media Presenter libraries according to their own preferences or to the requirements related to presentation styles (look and feel).

In the framework of our projects we work also on dedicated SGML database system technology [5, 3, 1] and plan to integrate that technology with the work described herein. This also applies to the work on a graphical authoring tool based on a generic version of SEPIA [28] which allows a designer to create presentation networks as described in the previous section.

Our further work includes the integration of other multimedia document models. So far our prototype supports the modeling of multimedia compositions by means of presentation nets that are based on extended OCPN. The implementation of document models for MHEG and/or HyTime according to the architecture shown in Figure 4 will follow. Currently, we look into the extension of the modeling capabilities of presentation networks, into the possibility of using the continuous communication protocol also for the delivery of presentation plans, i.e., multimedia documents, from the server to the client, and into alternative realizations of the continuous object management at the server. Further implementations will also include some additional standard interactions such as the random positioning in a composition by means of a slider. In addition, we look at the realization of the concepts presented for multimedia database clients by means of Java-applets. This is of special interest for the realization of the SMPs as this will allow end-users to download their favorite SMPs for some particular medium from any SMP-library available on the Wold Wide Web in the future.

Acknowledgements

We would like to thank our colleagues and students in the AMOS group for their contributions to the ideas and the implementations presented in this chapter.

REFERENCES

[1] K. Aberer, K. Böhm, and C. Hüser. The prospects of publishing using advanced database concepts. In *Proc. of the International Conference on Electronic Publishing, Document Manipulation, and Typography, EP94, Darmstadt, Germany*, pages 469–480. John Wiley & Sons, Ltd., 1994.

[2] J.F. Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11):832–843, November 1983.

[3] K. Böhm and K. Aberer. An Object-Oriented Database Application for HyTime Document Storage. In *Proceedings of the Conference on Information and Knowledge Management (CIKM94)*. Gaithersburg, MD, December 1994.

[4] K. Böhm, K. Aberer, and C. Hüser. Introducing D-STREAT - The Impact of Advanced Database Technology on SGML Document Storage. ⟨*TAG*⟩, 7(2):1–4, February 1994.

[5] K. Böhm, K.Aberer, and E. Neuhold. Administering Structured Documents in Digital Libraries. In *Advances in Digital Libraries*. Lecture Notes in Computer Science Vol. 916, Springer Verlag, 1995.

[6] K. Böhm and T.C. Rakow. Metadata for Multimedia Documents. In *SIGMOD Record (Special Issue on Meta-data for Digital Media)*, number 4 in SIGMOD Record. ACM, December 1994.

[7] Klaus Böhm, Hanno Wirth, and Werner John. Prototype Validation in Virtual Environments – Vision and First Implementation. In Joachim Rix, Stefan Haas, and Jose Teixeira, editors, *Virtual Prototyping – Virtual Environments and the Product Development Process*. Chapman & Hall, September 1995. ISBN 0 412 72160 0.

[8] M. Deegener, W. John, B. Kühnapfel, M. Löhr, G. Lux, and H. Wirth. A Basic Architecture for the Development of a Distributed Interactive Simulator. In F. Breitenecker and I. Husinsky, editors, *Proc. EUROSIM '95*, pages 357–362, Wien, 11.–15. Sep 1995. Elsevier.

[9] S.J. DeRose and D.G. Durand. *Making Hypermedia Work*. Kluwer Academic Publishers, 1994.

[10] DIMSYS. `VODAK V 4.0` *User Manual*. Technical Report (Arbeitspapiere der GMD) 910. GMD Sankt Augustin, 1995.

[11] Asuman Dogac, M. Tamer Özsu, Alexandros Biliris, and Timos Sellis, editors. *Advances in Object-Oriented Database Systems*, volume 130 of *NATO ASI Series F*. Springer Verlag, Berlin, 1994.

[12] N. Hirzalla, B. Galchuk, and A. Karmouch. A Temporal Model for Interactive Multimedia Scenarios. *IEEE Multimedia*, pages 24–31, Fall 1995.

[13] Illustra Information Technologies, Inc. *Illustra - Technology: WWW URL: http://www.illustra.com*, 1996.

[14] ISO/IEC. *Information Technology - Hypermedia/Time-based Structuring Language (HyTime)*, 1992. ISO/IEC IS 10744.

[15] ISO/IEC. *JTC 1/SC 29, Coded Representation of Multimedia and hypermedia Information Objects (MHEG) Part 1, Committee Draft 13522-1*, June 1993. ISO/IEC 10031.

[16] Won Kim. *Modern Database Systems - The Object Model, Interoperability and Beyond*. Addison-Wesley, 1995.

[17] W. Klas and K. Aberer. Multimedia Applications and Their Implications on Database Architectures. In *Advanced Course on Multimedia Databases in Perspective. University of Twente, The Netherlands*, June 1995.

[18] W. Klas, K. Aberer, and E. Neuhold. Object-Oriented Modeling for Hypermedia Systems using the `VODAK` Modeling Language (`VML`). In *[11]*. Springer Verlag Berlin, 1994.

[19] T.D.C. Little and A. Ghafoor. Synchronization and Storage Models for Multimedia Objects. *IEEE JSAC*, 8(3):413–427, April 1990.

[20] M. Löhr and T. C. Rakow. Audio Support for an Object-Oriented Database Management System. *Multimedia Systems Journal*, 3, 1995.

[21] K. Meyer-Wegener. *Multimedia Datenbanken*. Leitfäden der angewandten Informatik. Teubner Stuttgart, 1991.

[22] F. Moser, A. Kraiss, and W. Klas. L/MRP: A Buffer Management Strategy for Interactive Continuous Data Flows in a MMDBMS. In *Proceedings of the 21st International Conference on Very Large Databases 1995 (VLDB'95)*, Zürich, Switzerland, September 1995. Morgan Kaufmann.

[23] S.R. Newcomb, N.A. Kipp, and V.T. Newcomb. The HyTime Hypermedia/Time-based Document Structuring Language. *CACM*, 34(11), November 1991.

[24] Kingsley C. Nwozu, P. Bruce Berra, and B. Thuraisingham, editors. *Design and Implementation of Multimedia Database Management Systems.* Kluwer Academic Publishers, 1996.

[25] T. Rakow and P. Muth. The V^3 Video Server - Managing Analog and Digital Video Clips. In *Proc. SIGMOD '93*, pages 556–557, May 1993.

[26] T. Rakow, E. J. Neuhold, and M. Löhr. Multimedia Database Systems - The Notions and the Issues. In G. Lausen, editor, Tagungsband GI-Fachtagung Datenbanksysteme in Büro, Technik und Wissenschaft (BTW), Dresden März 1995, pages 1–29. Springer Verlag, Informatik Aktuell, 1995.

[27] R. Steinmetz and Klara Nahrstedt. *Multimedia: Computing, Communications and Applications.* Prentice Hall, 1995.

[28] N. Streitz, J. M. Haake, J. Hannemann, A. Lemke, W. Schuler, H. Schütt, and M. Thüring. SEPIA: A cooperative hypermedia authoring environment. In *D. Lucarella, J. Nanard, M. Nanard, P. Paolini (Eds.): Proceedings of the 4 th ACM Conference on Hypertext (ECHT '92), pages 11-22, Milano, Italy, November 30 - December 4, 1992*, pages 11–22. ACM Press, New York, 1992.

[29] H. Thimm and W. Klas. Playout Management – An Integrated Service of a Multimedia Database Management System. In *FirstInternational Workshop on Multi–Media Database Management Systems*, pages 28–30. Blue Mountain Lake, NY, USA, IEEE Computer Society Press, August 1995.

[30] H. Thimm and W. Klas. *Reactive Playout Management - Adapting Multimedia Presentations to Contradictory Constraints.* Technical Report (Arbeitspapiere der GMD) 916. GMD Sankt Augustin, 1995.

[31] H. Thimm and W. Klas. Delta-Sets for Optimized Reactive Adaptive Playout Management in Distributed Multimedia Database Systems. In *Proceedings of the 12th IEEE International Conference on Data Engineering, Orleans, Lousiana, USA*, February 1996.

[32] H. Thimm and W. Klas. Playout Management in Multimedia Database Systems. In *[24]*, 1996.

[33] H. Thimm and T.C. Rakow. A DBMS-Based Multimedia Archiving Teleservice Incorporating Mail. In W.Litwin and T.Risch, editors, *Proceedings of the First International Conference on Applications of Databases (ADB)*, pages 281–298, Vadstena, Sweden, 1994. Lecture Notes in Computer Science 819, Springer Verlag.

[34] H. Thimm, K. Röhr, and T.C. Rakow. A Mail-based Teleservice Architecture for Archiving and Retrieving Dynamically Composable Multimedia Documents. In *Proceedings of the Conference on Multimedia Transport an Teleservices, MMTT94*, 1994.

[35] H. M. Vin. Multimedia Conferencing in the Etherphone Environment. *IEEE Computer*, 24(10):69–79, October 1991.

[36] T. Wahl and K. Rothermel. Representing Time in Multimedia Systems. In *Proceedings of the International Conference on Multimedia Computing and Systems*, pages 538–543, Boston, MA, May 1994. IEEE Computer Society Press.

[37] J. Wäsch and K. Aberer. Flexible Design and Efficient Implementation of a Hypermedia Document Database System by Tailoring Semantic Relationships. In *Proceedings of the Sixth IFIP Conference on Data Semantics (DS-6), Atlanta, Georgia, USA, May 30 - June 2, 1995*, To appear in Meersman, Mark [Ed.]: Database Application Semantics. Chapman and Hall, 1996.