The Dortmund Family of Hypermedia Models - Concepts and their Application

Klaus Tochtermann
(University of Dortmund - Germany, Department of Computer Science, LS1
tochterm@ls1.informatik.uni-dortmund.de)

Gisbert Dittrich
(University of Dortmund - Germany, Department of Computer Science, LS1
dittrich@ls1.informatik.uni-dortmund.de)

Abstract: This paper presents the Dortmund Family of Hypermedia Models (DFHM). Existing formal models for hypermedia mostly lack the flexibility and adaptability and, often not more than one existing system conforms to such a model. The DFHM overcomes this drawback by means of optional and alternative data types. The conformance of a hypermedia system to the DFHM can be conditionalised upon one member of the family.

The DFHM has been formalised in VDM, but the aim of this paper is to give an informal overview of the main concepts. Therefore, any formalisms are omitted here. The first part of the paper deals with hypermedia fundamentals from a conceptual perspective. Apart from basic concepts, e.g. nodes and links, also structuring concepts, e.g. views, folders and others, are discussed in detail. Some examples are given to convey how models for existing hypermedia systems can be derived from the DFHM. The second part demonstrates the power of these concepts by introducing main features of a hypermedia system that has been developed for the use in educational settings. This hypermedia system bases upon a member of the DFHM.

Key Words: hypermedia, hypermedia model, hypermedia system, hypermedia structuring

Category. H.5.1, I.7.2

1 Introduction

In recent years, several hypermedia models have been developed of which the Dexter Reference Model is the most influential model in the literature today. As we see it, existing models for hypermedia fall into two main categories: 1) abstract but yet informal models, and 2) abstract and precise mathematical models. Models belonging to the first category are for example the HAM [Campbell, Goodman 88], HDM [Garzotto, Paolini 92], the HM-Data Model [Maurer et al. 93], [Maurer et al. 94]. or the Amsterdam Hypermedia Model as it is described in [Hardman, Bulterman 94]. The second category includes the Dexter Reference Model as it is described in [Halasz, Schwartz 90], the VDM-Model of [Lange 90], the hypergraph model of [Tompa 89], or our family of formal models published in [Tochtermann, Dittrich 95].

In contrast to others, the model of [Tochtermann, Dittrich 95] is flexible and can easily be adapted to different requirements. Since this model supports differing variations of the data types it can be regarded as a family of interrelated models rather than one model with fixed data type specifications. Conformance to the model can than be conditionalised upon one member of the family of models. While in [Tochtermann, Dittrich 95] the emphasis is placed on the technical side of the model, this paper intends to afford an insight into the various concepts without the use of

formalisms. Apart from that, this paper also illustrates the application of these concepts in an existing hypermedia system. In this way, this paper may help to reduce the existing gap between theory and application in hypermedia modelling, c.f. [Tochtermann, Dittrich 95].

Similar to the Dexter Reference Model and the Amsterdam Hypermedia Model which have been developed in Dexter and Amsterdam, respectively, we call our model the "Dortmund Family of Hypermedia Models" (DFHM).

1.1 Structure of the Paper

In section 2, our paper begins with an informal description of essential parts of the DFHM that has been developed at the University of Dortmund, Germany. Not only does this section pay attention to hypermedia fundamentals, like nodes, anchors, links etc., but focuses also on sophisticated structuring concepts. Several structuring concepts are illuminated: composite nodes, link structures, views, view nodes, and folders. The composite node facilities allows nodes to contain not only basic data like text, graphic etc., but also other nodes. Link structures are used to organise links. Views serve the purpose of bringing together often used data and hiding less used data. In order to allow views to filter not only entire nodes of an underlying hyperdocument but also portions of underlying nodes, we introduce the concept of view nodes. Folders allow a collection of nodes, links, or even folders themselves to be treated as a single unit. Section 3 mainly focuses on the application of the proposed concepts and illustrates how these concepts have been integrated into a hypermedia system that has been developed in Dortmund. Related work is discussed in section 4 and section 5 summarises the benefits of our model. Finally, section 6 closes the paper with an overview of our current and future activities.

2 Hypermedia Concepts in the DFHM

This section informally introduces essential parts of the DFHM. The formal parts of the model are written in VDM (Vienna Development Method, [Jones 90]) but are omitted here. However, the formal specifications of the hypermedia data types without any operations can be found in [Tochtermann, Dittrich 95]. The overall model including operations acting upon the data types is published in [Tochtermann 95a]. The basic model has been extended in types so that typed hypermedia systems can formally be described. This extension is published in [Tochtermann 95b] and [Kröcker 95]. Finally, to illustrate the power of our model VDM specifications of the following hypermedia systems exist: Multimedia

ToolBook [ToolBook 92] and HyperDoLL are formalised in [Tochtermann 95a], parts of the formal specification of Intermedia exist in [Tochtermann, Dittrich 95], and parts of gIBIS, a typed hypermedia system developed by Conklin and Begemann [Conklin, Begemann 88], are specified in [Kröcker 95].

2.1 Hypermedia Fundamentals

Hypertext is a concept of information management in which data is stored in a network of nodes and links. Nodes represent some information and the links create

associations between disparate nodes. In a link, the information, e.g. a word in a text, that provides the link should be related to the information it will access when the link is selected. Non-linear information structures consisting of nodes and links are referred to as hyperdocuments. The information manipulated by hypertext systems has undergone a remarkable evolution in recent years. The first systems linked nodes containing mainly discrete data like text and graphic. Current systems, however, can also manipulate continuous data like video, animation or sound. This has led to the term hypermedia system which synergestically combines the powerful concepts of hypertext and multimedia.

The subsequent sections informally describe common hypermedia concepts, e.g. nodes, links, hyperdocuments, in greater detail.

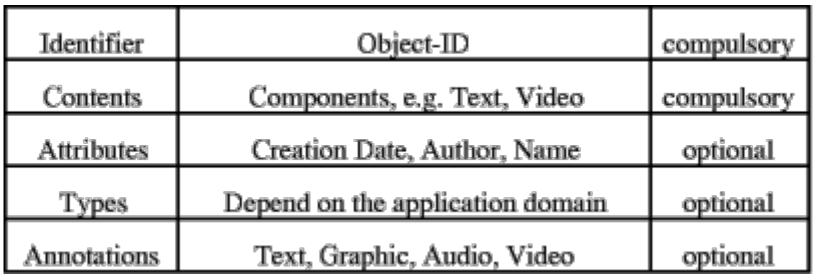
2.1.1 Nodes

Nodes are self-contained information units, i.e. users can understand the node's content without knowledge about any context information. Unlike in linear documents, this is important in hyperdocuments because users can reach a node from more that only one context.

A node has five constituents of which two are compulsory and three are optional, c.f. table 1. Compulsory constituents are a node's content and a node's identifier. In addition, a node may have attributes, types, and annotations.

In hypermedia systems the node's content can either be discrete media, i.e. text and graphics, or continuous media, i.e. audios and videos. Attributes are used to store meta information about a node, like creation date, author, name of the node etc. Types assist in reinforcing hypertext structure and can be used to incorporate knowledge in hypermedia systems [Nanard, Nanard 91]. To avoid confusions between attributes and types we want to clarify our notion of the two terms. Our notion of types and attributes is derived from the notion of [Hammwöhner, Kuhlen 94]. Basically, they define types as restrictions on hypermedia objects and their combination, whereas attributes - note, that attributes are referred to as labels in [Hammwöhner, Kuhlen 94] - only indicate a discursive function of a hypermedia object. Due to this, types require functions or operations to control the restrictions on the hypermedia objects; these functions are formalised by data type invariants in the formal specification of the DFHM [Kröcker 95]. However, in contrast to [Hammwöhner, Kuhlen 94] our concept of types does not support inheritance. Instead we allow each hypermedia object to have more than only one type.

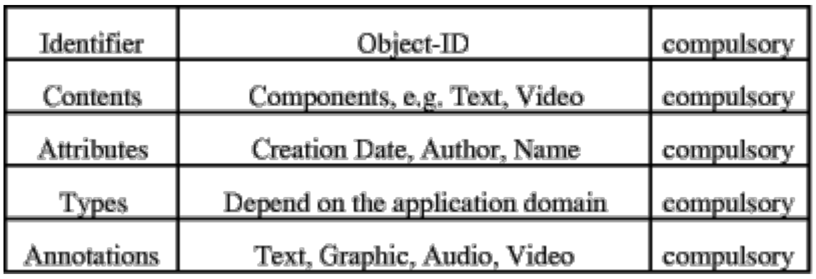
Typed nodes are indispensable for semi-formal representation of knowledge and are often used in hypermedia systems in which concepts of artificial intelligence are integrated. Systems combining concepts of expert systems and hypermedia systems are referred to as expertmedia systems [Rada, Tochtermann 95]; an overview of artificial intelligence in connection with hypermedia is given in [Tochtermann, Zink 95]. Basing upon typed nodes, link constraints can be employed. To give an example, a link constraint can guarantee that nodes of certain types can only be connected by links with a compatible link type. Often types depend on a particular application domain even though some systems are known, e.g. [Neubert 93], that support types that do not depend on a particular domain. Annotations provide further information about the content of the node they are attached to. Normally annotations can not be dissociated from nodes and can therefore only be accessed from the node they belong to.



Identifier	Object-ID	compulsory
Contents	Components, e.g. Text, Video	compulsory
Attributes	Creation Date, Author, Name	optional
Types	Depend on the application domain	optional
Annotations	Text, Graphic, Audio, Video	optional

Table 1: Constituents of nodes

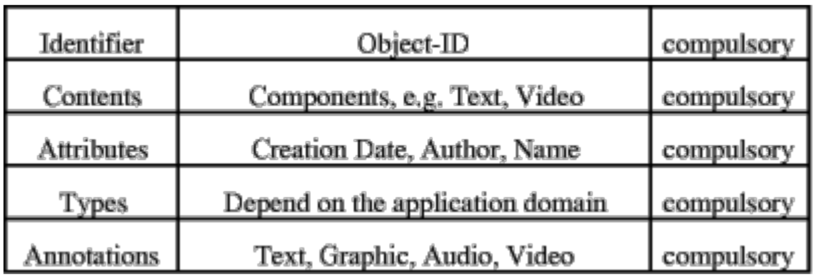
On the basis of this specification we can derive different specifications for different notions of nodes. For example, in the strongest model of the DFHM all constituents of nodes are compulsory:



Identifier	Object-ID	compulsory
Contents	Components, e.g. Text, Video	compulsory
Attributes	Creation Date, Author, Name	compulsory
Types	Depend on the application domain	compulsory
Annotations	Text, Graphic, Audio, Video	compulsory

Table 2: Constituents of nodes in the strongest model of the DFHM

Further, Multimedia ToolBook supports neither annotations nor types but attributes for nodes. Thus, the corresponding model of the DFHM provides the following constituents for nodes:



Identifier	Object-ID	compulsory
Contents	Components, e.g. Text, Video	compulsory
Attributes	Creation Date, Author, Name	compulsory
Types	Depend on the application domain	compulsory
Annotations	Text, Graphic, Audio, Video	compulsory

Table 3: Constituents of nodes in MultiMedia ToolBook

In contrast to this, gIBIS requires types for nodes. This results in the following constituents of nodes in the corresponding model:

Identifier	Object-ID	compulsory
Contents	Components, e.g. Text, Video	compulsory
Attributes	Creation Date, Author, Name	compulsory
Types	Issue, Position, Argument, Other	compulsory

Table 4: Constituents of nodes in gIBIS

2.1.2 Components

The spread of hypermedia systems requires appropriate concepts for managing the content of nodes, like text, graphic, video etc. In order to meet this requirement we introduce a concept of so-called components. (Note, that our notion of "component" corresponds to the notion of the Dexter's "within-component" but not to the Dexter's notion of "component!) Figure 1 depicts the idea:

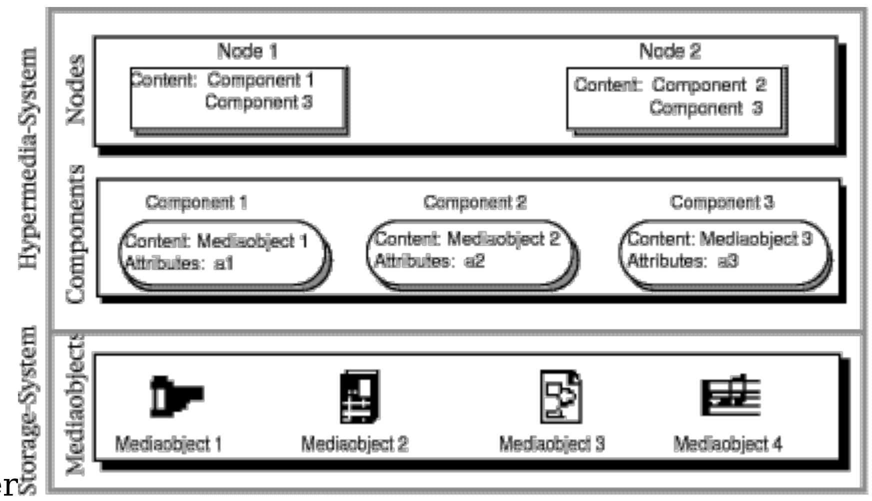
center

Figure 1: Concept of components

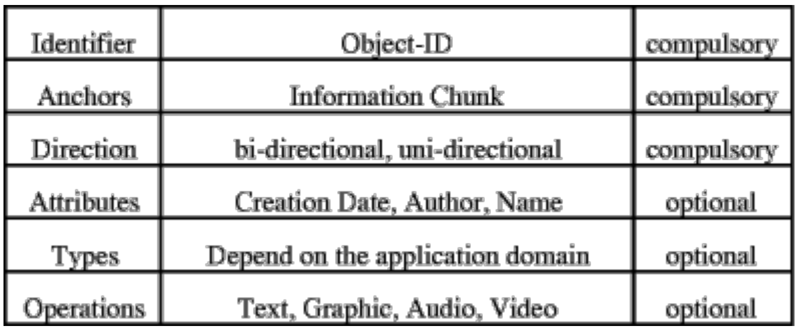
An underlying storage system contains so-called media objects representing raw multimedia data, e.g. text, graphic, audio, or video. For storage management a multimedia database, a relational database or a file system can be used. One further important advantage of this concept is, that similar to Intermedia [Yankelowich, Haan 88], [Haan, Kahn et al. 92] or Hyper-G [Maurer 95], [Dalitz, Heyer 95], [Flohr 95] media objects can be created by special applications, e.g. text with a word processor. Thus, corresponding hypermedia systems provide an environment of "small" applications rather than integrating the entire functionality within one application. At the component level, a media object is associated with a component. By means of attributes a component may provide meta information about a media object. This additional but optional information can support retrieval purposes of components. Finally, each node can refer to an arbitrary but finite number of components. In contrast to media objects, components and nodes can be managed by the hypermedia system.

Identifier	Object-ID	compulsory
Content	Media Object	compulsory
Attributes	Creation Date, Author, Name	optional

Table 5: Constituents of components

2.1.3 Links

Links connect several nodes to another. A link's anchor describes the attachment of links to nodes. Selecting an anchor in a source node enables the user to follow the link to one of the existing target nodes. Basically a description of links must take its identification, anchors, the link's direction, optional attributes, optional link types and optional operations into consideration, c.f. table 6. Links with one or more origin and one or more destination are called n-to-m-links. In addition, a link can either be static or dynamic. In contrast to static links, where the anchors are fixed, dynamic links have destinations that are function denotations. That is, instead of having links pointing at a fixed point, like a sequence of characters, they contain a function. When the link is followed, this function must be interpreted and the result yielded is the final destination. The direction of a link specifies whether a link is uni- or bi-directional. In contrast to uni-directional links, bi-directional links cannot only be followed from the origin to the destination, but also from the destination to the origin. Operations consist of a series of commands which are executed when the user follows a link. For example, assuming that a user establishes a link to a video. Then, the link could be supplied with an appropriate operation, changing the status of the sound track, e.g. switch off the sound of the video. Similar to nodes links can provide attributes and types. With typed links users can differentiate between various kinds of relationships between connected nodes.

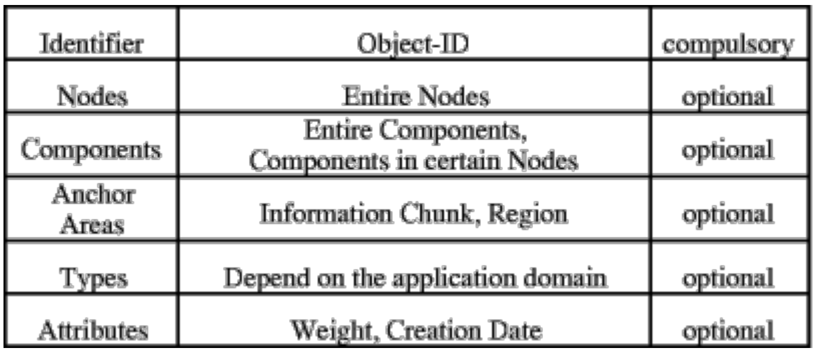


Identifier	Object-ID	compulsory
Anchors	Information Chunk	compulsory
Direction	bi-directional, uni-directional	compulsory
Attributes	Creation Date, Author, Name	optional
Types	Depend on the application domain	optional
Operations	Text, Graphic, Audio, Video	optional

Table 6: Constituents of links

2.1.4 Anchors

The attachment of links to nodes is referred to as anchor. The origin anchor is the starting point of a link whereas the destination anchor is the endpoint of a link. An anchor comprises five optional parts: nodes, components, anchor areas, types and attributes. The first three fields describe the specific origin or destination of an anchor so that in each anchor at least one of the first three fields must be specified. The more these fields are specified, the more detailed is the description of an anchor. For example, an origin can either be an entire node, a component independent of any nodes referring to this component, a component within a certain node, or an anchor area within a component. To take another example, an anchor area can be a region within a graphic, a sequence of characters, or a sequence of frames within a movie. The attributes might serve the purpose of expressing weights for link anchors as used in [Tochtermann, Dittrich 92]. In addition [Tochtermann 95a] also allows hyperdocuments to be an anchor. Due to this it is possible to follow a link that interconnect nodes of different hyperdocuments. Finally, some hypermedia systems also support typed anchors, e.g. MacWeb [Nanard, Nanard 93]. Table 7 lists the constituents of anchors.

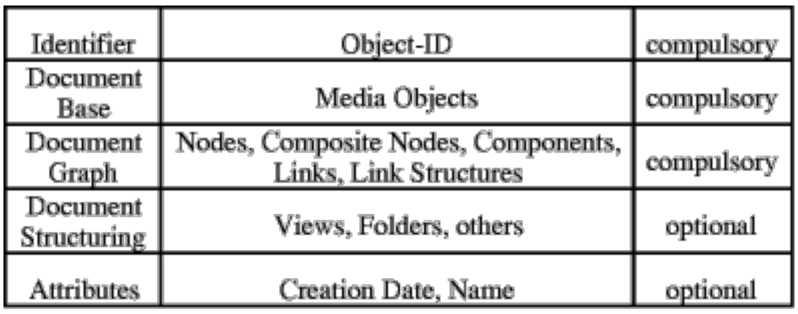


Identifier	Object-ID	compulsory
Nodes	Entire Nodes	optional
Components	Entire Components, Components in certain Nodes	optional
Anchor Areas	Information Chunk, Region	optional
Types	Depend on the application domain	optional
Attributes	Weight, Creation Date	optional

Table 7: Constituents of anchors

2.1.5 Hyperdocuments

Basically we can refer to a set of nodes which are connected by links as a hyperdocument. However from the modelling perspective different kinds of information are stored in different layers of a hyperdocument, c.f. table 8. The document base manages all media objects, like texts or videos etc. For example a multimedia database can be employed for this purpose. The document graph represents the non-linear information structure of a hyperdocument and it is made up of nodes, composite nodes (c.f. section 2.2.1), links, link structures (c.f. section 2.2.2) and components. Some modern hypermedia systems support the user with structuring concepts, so that the last layer of a hyperdocument provide optional document structurings (c.f. section 2.2). Finally, a hyperdocument is always identifiable and may have attributes. Our model does not supply types for hyperdocuments, although one might imagine that types can be employed to classify different kinds of hyperdocuments, e.g. learning documents created by CBT systems, information documents created by point-of-information systems etc. In our opinion types for hyperdocuments depend on the hypermedia system which has been used to create the document. However, typically hypermedia systems are adapted to a certain application domain, e.g. learning and teaching, and the consequence is that only hyperdocuments of a corresponding kind can be created with such a system. Hence, explicit types for hyperdocuments are not strictly necessary.

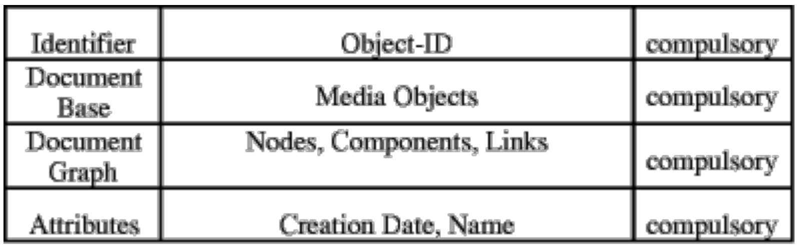


Identifier	Object-ID	compulsory
Document Base	Media Objects	compulsory
Document Graph	Nodes, Composite Nodes, Components, Links, Link Structures	compulsory
Document Structuring	Views, Folders, others	optional
Attributes	Creation Date, Name	optional

Table 8: Constituents of hyperdocuments

Some hypermedia systems do not explicitly support structuring mechanisms, e.g.

MultiMedia ToolBook. Thus, the corresponding member of the DFHM provides the following constituents for hyperdocuments.



Identifier	Object-ID	compulsory
Document Base	Media Objects	compulsory
Document Graph	Nodes, Components, Links	compulsory
Attributes	Creation Date, Name	compulsory

Table 9: Hyperdocuments in MultiMedia ToolBook

Further examples of members of the DFHM are given in [Tochtermann, Dittrich 95].

2.2 Hypermedia Structuring Concepts

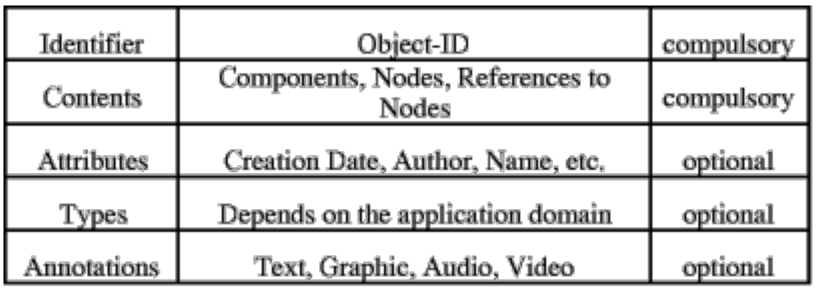
Generally, structuring concepts abstract a group of information objects into higher-order objects. For example, structuring concepts allow a collection of nodes and/or links to be treated as a single unit. They may help to organise and to categorise hyperdocuments and thus, to enhance their quality and readability.

This section introduces five structuring concepts integrated in the DFHM.

2.2.1 Composite Nodes

In his famous paper "Seven Issues for the next Generation", Halasz [Halasz 88] proposed to expand the basic node-link concept of hypertext. His paper spawned many suggestions for structuring concepts, e.g. [Grønbaek 94], [Wang, Rada 94]. One idea was to find a way of dealing with node hierarchies by means of composite nodes, i.e. nodes containing other nodes. One can differentiate between hierarchy by reference and hierarchy by inclusion. A reference relation between a composite node and other nodes means that the composite node references the other nodes that are

located outside the composite node. An inclusion relation means that the composite node contains the other nodes. As a consequence, the included nodes are not visible from outside the composite node. The DFHM does not differentiate between "simple" nodes, i.e. nodes containing only components, and composite nodes but proposes to allow nodes containing components together with other nodes or references to other nodes. The following table summarises the characteristics of composite nodes.

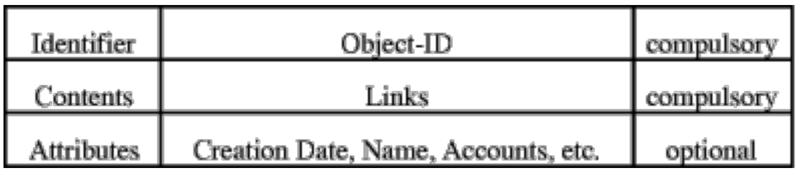


Identifier	Object-ID	compulsory
Contents	Components, Nodes, References to Nodes	compulsory
Attributes	Creation Date, Author, Name, etc.	optional
Types	Depends on the application domain	optional
Annotations	Text, Graphic, Audio, Video	optional

Table 10: Constituents of composite nodes

2.2.2 Link Structures

In many hypermedia systems links are global, this means that each link is available at all time to all users. However, faced with the fact that users often want to work in their own context a more sophisticated concept for dealing with links is required. The idea of link structures has its origins in the webs of Intermedia [Yankelowich, Haan 88], [Haan, Kahn et al. 92]. A link structure is a set of links that interconnects defined parts of a hyperdocument. Link structures also allow to impose different links on the same document. Using this structuring concept different contexts for different users can be provided by different link structures. Conceptually, a link structure consists of an identification, a set of links, and optional attributes, c.f. table 11.



Identifier	Object-ID	compulsory
Contents	Links	compulsory
Attributes	Creation Date, Name, Accounts, etc.	optional

Table 11: Constituents of link structures

2.2.3 Views

Views are widely accepted structuring concepts in various areas of computer science. For instance, in database systems different groups of users are provided with their own view of the database. A benefit is, that users do not have to be aware of the overall complexity of the underlying database. Views can also simplify user interfaces by allowing a user to ignore information that are of no interest to him. Besides, views provide a level of security and independence by shielding private data from unauthorised access and modification.

Firstly, we give some examples demonstrating the intuitive appeal of views in hypermedia: In the near future, digital libraries will play an important role in education, c.f. [Marchionini, Maurer 95]. The hyperlinked electronic documents offered by digital libraries allow teachers to prepare and to provide courses that have been conceptually inaccessible in traditional libraries. Normally, courses base on different chapters of different books and if a teacher wants to provide this material to the students he will have to gather the material from the different information resources. However, in digital libraries teachers might define a view containing only the information being pertinent to that course. Such a view can be regarded as a virtual book comprising different interconnected parts of the g345 different underlying electronic documents. Another example in an educational setting is illustrated in figure 2. The figure depicts a hyperdocument representing the teacher's material. However, if the teacher wants students to work on a particular lesson only parts of his material should be available. Therefore, it is reasonable to define a view containing only the information being pertinent to that lesson. In figure 2, two views for two different lessons are displayed. Since views may reflect other coherences than those between the underlying hyperdocument, it should be allowed to superimpose new links in a view. These new links are displayed in grey.

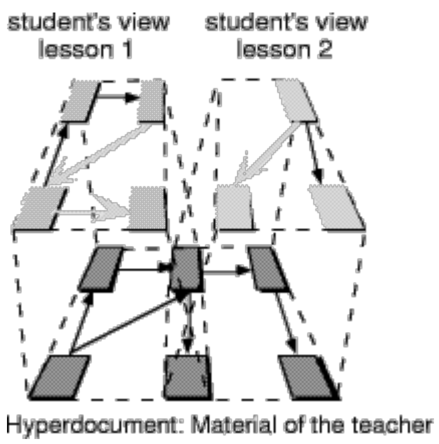
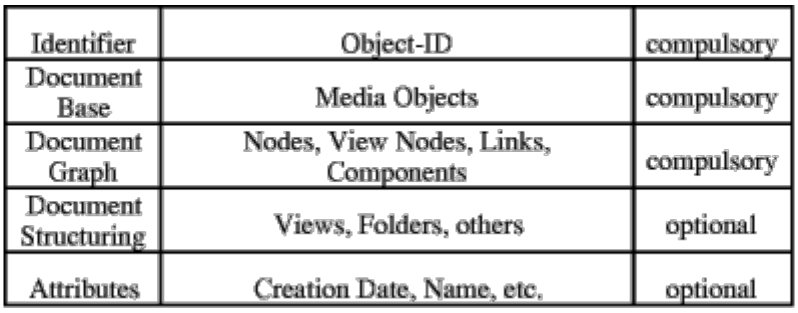


Figure 2: Views on a hyperdocument

From the conceptual perspective, views are made up of nodes, links, components and media objects. These constituents can be organised in a document graph. Unlike hyperdocuments, a special kind of nodes, so-called view-nodes, is allowed in the document graph of views (c.f. section 2.2.4). Similar to hyperdocuments, views may provide structuring mechanisms so that cascaded views can be allowed. Finally, personalisation of views can be achieved by augmenting views with nodes that do not exist in the underlying documents. This would allow students to add personal comments to a view on the teacher's material.



Identifier	Object-ID	compulsory
Document Base	Media Objects	compulsory
Document Graph	Nodes, View Nodes, Links, Components	compulsory
Document Structuring	Views, Folders, others	optional
Attributes	Creation Date, Name, etc.	optional

Table 12: Constituents of views

2.2.4 View Nodes

Employing our concept of components, we can enhance the concept of views by introducing view nodes. A view node is a node containing only chosen components or nodes of underlying nodes or view nodes. The following figures outline this idea. On the left, figure 3 sketches an underlying node and two appertaining view nodes. The view node on the left contains three of the six components of the original node. The view node on the right contains only two of these. In addition, our concepts allows view nodes based upon several underlying nodes as shown on the right in figure 3. To take an example, assume a node in the teacher's view deals with a mathematical exercise including its solution. The solution, however, should be concealed from students so that a corresponding view node should only contain the exercise but not its solution.

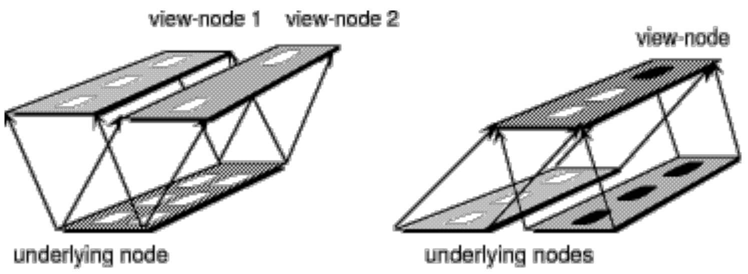


Figure 3: The concept of view nodes

2.2.5 Folders

It is a well-known fact that composition schemes are necessary for enhancing the organisation and thus the readability of hyperdocuments. Composition schemes allow a group of links, nodes or even other composition schemes concerning one topic to be treated as a single unit. They can be employed to organise or categorise large collections of nodes, links or composition schemes and users are encouraged to use them as an additional organisation structure. Both, authors and readers will benefit from composition schemes: the logical structure of the information can explicitly be modelled by the author and thus, it is much easier for readers to get a sound grasp of the mental model of the hyperdocument.

Our model is extended by introducing so-called folders for these purposes. (Note, that folders in this paper correspond to the templates in [Tochtermann, Dittrich 95].) A folder is a container object containing nodes, links or even other folders. To better understand the underlying idea of folders it may be helpful to think of directories in a file system, which contain files or other directories. The files and directories correspond to nodes and folders, respectively.

Folders support links between folder documents and documents that exist outside of the folder. Cascaded folders, i.e. folder hierarchies, can be used for navigation purposes. To take an example, in a digital library the logical structure of subject catalogues can be organised by folders. If users want to access to documents dealing with office automation, navigation could start at the top of the hierarchy by opening a folder representing the subject "Information Systems". In this folder other subjects, e.g. "Information Systems Application", are also represented by folders. Opening the folder "Information Systems Application" can lead to a further folder "Office Automation" containing divers documents to this subject (c.f. section 3, figure 7). By the way, documents or even folders themselves can be contained in several different folders. For example this is useful for modelling subject catalogues in digital libraries, since normally documents can be assigned to several different catalogues.

Conceptually, folders have an identification and consist of nodes, links, structurings, e.g. views and folder, etc., so that basically folders can be regarded as special hyperdocuments. (For more information about the differences between the two the reader is referred to [Tochtermann, Dittrich 95].)

Identifier	Object-ID	compulsory
Document Base	Media Objects	compulsory
Document Graph	Nodes, Composite Nodes, Components, Links, Link Structures	compulsory
Document Structuring	Views, Folders, others	optional
Attributes	Creation Date, Name	optional

Table 13: Constituents of folders

2.3 Relationships between the Constituents of Hyperdocuments

The following figure gives an overall overview of the relationships between the different constituents of hyperdocuments in the strongest model of the DFHM, i.e. the model where all constituents of all hypermedia objects are compulsory. An arrow labelled with "n, m" represent a n-to-m relationship between the connected entities, e.g. a node can contain m components and a component can be contained in n different nodes.

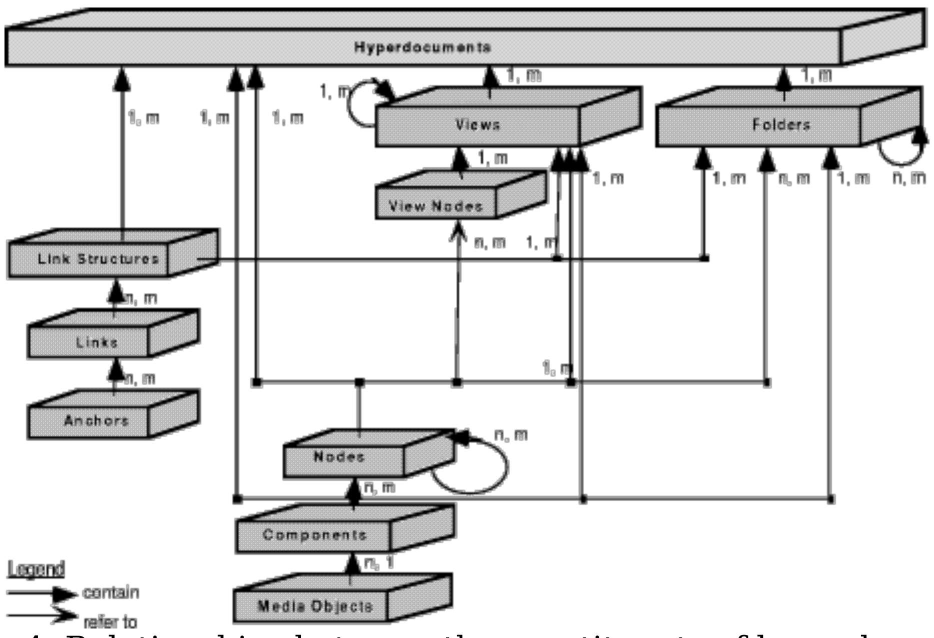


Figure 4: Relationships between the constituents of hyperdocuments

2.4 Operations

Our data type specifications need to be extended in order to cope with the tasks faced by most programmers who want to develop hypermedia systems on the basis of a member of the DFHM. The major extension is to cope with the fact, that hypermedia systems are interactive systems and therefore manipulation of complex data structures is very important. Thus, the concern in this section is with an overview of the operations provided by the DFHM. In the formal VDM specifications, operations consist of pre- and post-conditions. The pre-condition states what must be true in order for an operation to be defined. The post-condition defines the result of the processing the operation carried out.

The DFHM supports three different classes of operations: read-only operations, write operations and retrieval operations. Retrieval operations are formalised in [Kröcker 95] whereas all other operations can be found in [Tochtermann 95a]. All operations are specified for the data types of the strongest model of the DFHM so that operations for weaker models of the family can be obtained by generalisation.

2.4.1 Read-only Operations

Operations appertaining to this class do not change the state of the objects they are acting upon. For example, the follow-link operation requires a link's origin as input parameter and yields a link's destination as output. However, the operation does not change the link. Typically, the following operations belong to this class: open- hyperdocument, open-node, navigational operations such as follow-link etc.

2.4.2 Write Operations

The operations of this class change the state of objects. For example, creation of a new node changes the state of a hyperdocument when the new node is stored. In addition all modify operations belong to this class, e.g. modifying the type of a link changes the state of the link .

2.4.3 Retrieval Operations

Read- and write-operations, although not always formally specified, can be found in almost all other hypermedia models. However, unlike other models, the DFHM also provides retrieval operations. Generally, one can differentiate between content-oriented and structure-oriented retrieval operations. Content-oriented retrieval operations may apply techniques of information retrieval, e.g. using author indexes and subject indexes serving as descriptors of the document they represent. However, the creation of indexes is time consuming and typical hypermedia systems are interactive systems so that the content of nodes will change frequently. As a consequence, indexes have to be updated frequently in order to maintain consistency between indexes and documents. On the other hand, structure-oriented retrieval operations capture the structure of the data.

In the DFHM the structure of the data corresponds to the (typed) document graph of an hyperdocument where (typed) nodes are connected to each other by (typed) links. The advantage of structure-oriented retrieval operations over content-oriented retrieval operations is that they do not rely on indexes. Hence, at the moment our model only provides structured-oriented retrieval operations. Typical structure- oriented retrieval operations yield substructures of a hyperdocument's document graph, e.g. a path consisting of nodes and links of particular types or a document graph with a link structure containing only links of a certain type.

2.5 Some Remarks about the Formalisation of the DFHM

The previous sections presented the DFHM in a rather informal way. The idea was to give a broad overview of essential parts of the model rather than providing technically detailed descriptions of the different concepts. However, in this section we briefly want to summarise key features of the formal part. Firstly there are two main drawbacks of existing formal models:

1) In [Halasz, Schwartz 90] Halasz and Schwartz sum up that no existing system conforms to the Dexter Model and that the model is more powerful than existing hypermedia systems (this is also the case for other formal models). This drawback results from the fact, that formal models are specified without using optional or alternative mechanisms and, thus, the models are not flexible and adaptable to different requirements.

2) A second drawback is that formal models provide a limited set of concepts and pay little attention to further extensions of the model. For example, precise mathematical models mostly deal with fundamentals, e.g. nodes, anchors, or links, and rather less attention is paid to sophisticated structuring concepts. However, there has been a spate of interest in structuring concepts for hypermedia. In recent years, a lot of exciting concepts have been presented and often examples were given to demonstrate

their intuitive appeal. Unfortunately, precise models often don't take sophisticated structuring concepts into account and normally structuring concepts are not formally described. In our opinion both will benefit from a formal specification of such concepts: The value of a model will increase because it is not only limited to basic concepts, like nodes and links. The value of structuring concepts will increase because often precise specifications afford remarkable insight into the concept.

The DFHM uses optional and alternative data type specifications to eliminate the first problem mentioned above. The result is a flexible model that can easily be adapted to different requirements. It is worth stating at this point that different hypermedia systems have been formalised with different members of the DFHM. To overcome the second problem is more difficult than the last one, as an enormous amount of various structuring concepts exists. Therefore the DFHM provides formal specifications of well-known structuring concepts. In this way, we provide a basis for further integrations of structuring concepts. [Tochtermann, Dittrich 95] and in particular [Tochtermann 95a] give several examples illustrating how the DFHM can be extended in further structuring concepts and adapted to additional requirements.

3 Application of the DFHM

In this section, we illustrate the intuitive appeal of most of the concepts presented in the previous section. Different hypermedia systems base upon a member of the DFHM. The most succesful systems are HyperMed and HyperDoLL. HyperMed is a hypermedia system for anatomical education; a description can be found in [Tochtermann et al. 96]. In this section we place the emphasis on HyperDoLL. HyperDoLL is the German acronym for "Hypermediales Dortmunder Lehr- und Lernsystem". This means that HyperDoLL has been developed in Dortmund and that it is a hypermedia system for educational settings. It is worth mentioning at this point that it would be helpful to give an illustrative example of a complete hyperdocument structured in accordance with the model. However, such a description would go beyond the scope of this paper.

3.1 HyperDoLL at a Glance

Using a member of the DFHM, i.e. a formal model, we implemented a prototype hypermedia system called HyperDoLL. The formalisation can be found in [Tochtermann 95a]. HyperDoLL has been developed in C++ runs on Apple Macintosh. The tool is adapted to the needs of lecturers in educational settings and a sophisticated view concept is provided to support lecturers in creating and presenting hypermedia-based talks or lectures.

This section briefly introduces important features of HyperDoLL. Some of the future developments concerning HyperDoLL are described in section 6. Some of these activities are well advanced and others have been specified but no completion date can be announced at the time of writing. HyperDoLL supports the basic concepts described in section 2, e.g. nodes, components, links, anchors, and hyperdocuments. In addition, views, view nodes, and folders serve for structuring purposes.

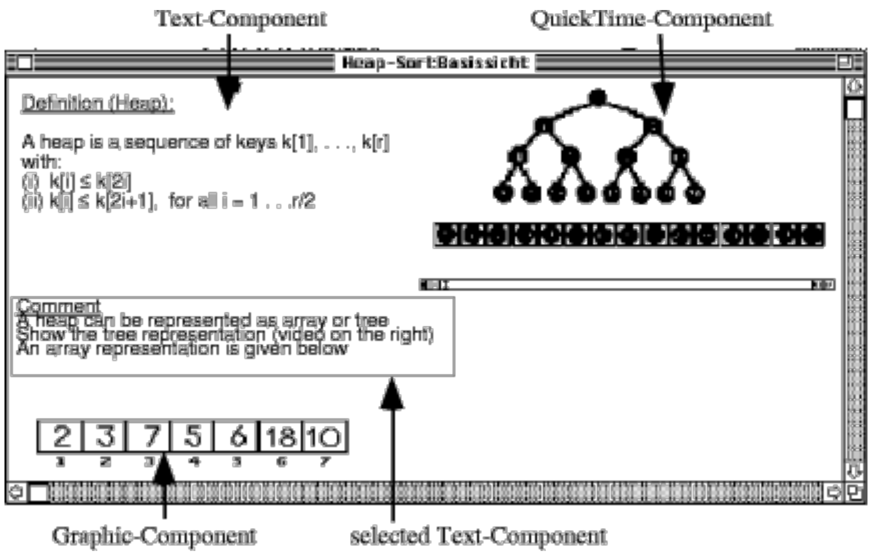
At the moment, our prototype stores all hypermedia objects with the exception of media objects, in a HyperDoLL document. The Macintosh file system is used for separate storage management of media objects. Thus, a media object representing a

video is associated with a QuickTime movie file, a graphic with a file in PICT format, a text with a file in the Macintosh standard text format and sound with a file in AIFF format. In addition, HyperDoLL also offers a comfortable text editor which supports users in creating new components. This concept enables authors to use standard editing programs for creating and modifying media objects.

HyperDoLL offers a graphical finder displaying all existing components. Components can be inserted into nodes by using the drag-and-drop paradigm. Within a node a component can be moved by dragging it with the mouse to the desired place. Furthermore, authors can create attributes for components, e.g. name, notes and keywords representing information about the component's content.

Figure 5 displays a node of a hyperdocument created with HyperDoLL. The appertaining hyperdocument explains the heapsort algorithm using appropriate animations. The node contains four components: a textual definition of a heap, a QuickTime movie illustrating the heapsort algorithm, some comments of the lecturer and a graphic showing an array representation of a heap. The comment is selected and can be dragged to any place within the node.

HyperDoLL also offers a view concept to users. At the moment, however, HyperDoLL only allows a 1-to-m relationship between nodes and view-nodes, i.e. it is possible to define several view nodes for each node, but each view node can only base on one underlying node (c.f. figure 3). Thus, different views on an underlying document graph can be defined. In order to enhance the quality of talks and lectures, lecturers can be provided with a lecturer view and a presentation view. The idea is to display the lecturer's view on a computer's screen. This view supports the lecturer during his talk. The presentation view is projected with an LCD panel or videobeam and enables the audience to follow the talk. Figure 6 shows a view-node for the node displayed in figure 5. The view-node belongs to the presentation view and thus it contains only the components being important for the audience of a lecture. Hence, the component representing the lecturer's private comment and the graphic don't appertain to that view-node.

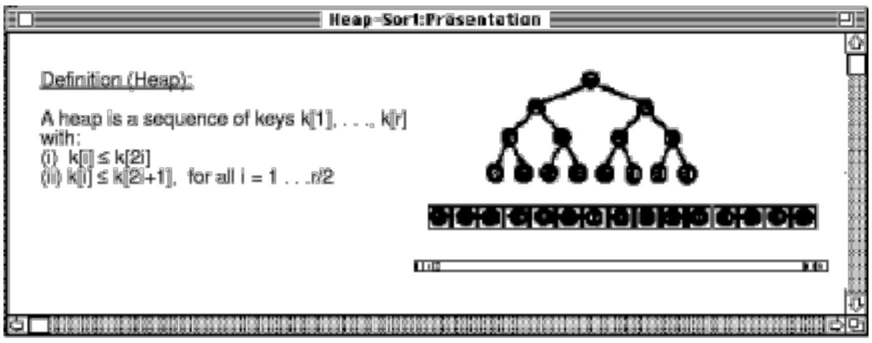


Definition (Heap):

A heap is a sequence of keys k[1], . . ., k[r]
with:
(i) k[i] ≤ k[2i]
(ii) k[i] ≤ k[2i+1], for all i = 1 . . . r/2

Comment
A heap can be represented as array or tree
Show the tree representation (video on the right)
An array representation is given below

Figure 5: Node consisting of four media-objects



Definition (Heap):

A heap is a sequence of keys k[1], . . ., k[r]
with:
(i) k[i] ≤ k[2i]
(ii) k[i] ≤ k[2i+1], for all i = 1 . . . r/2

Figure 6: View-node for the node displayed in figure 5

The links in HyperDoLL are uni-directional, operational and provide attributes. The operations attached to links serve the purpose of synchronising different views. Synchronising of views is important for presentations. For example, it must be guaranteed that the projected node in the presentation view corresponds to the current node in the lecturer's view. In order to keep both views consistent, a link operation is executed when the lecturer follows a link in his view. Such an operation ensures that a node of the lecturer's view is displayed on the screen and that the corresponding view node is projected for the audience.

By means of attributes links can be concealed from students. For example, the visibility of links in a learning view, i.e. a view providing questions and answers, can depend on the answers the students give to the questions. This allows us to adapt the material "on the fly" to the student's knowledge. This idea resembles the idea of "Authoring on the fly", a term coined by H. Maurer. In contrast to our approach, the idea of Maurer is to generate new material during the process of presenting a lecture, e.g. recording and digitising actions of the lecturer such as highlighting certain material.

Finally, currently the concept of folders is being integrated in HyperDoLL. Each folder can contain an arbitrary number of nodes or even other folders and each node can belong to more than only one folder. The following figure shows a folder hierarchy in HyperDoLL. Its visualisation corresponds to the Apple User Interface Guidelines. The folder hierarchy sketches a subject classification system for Computer Science Information in a digital library. The classification corresponds to the ACM Computing Classification System. At the lowest level, e.g. "Office Automation", articles dealing with this subject are displayed.

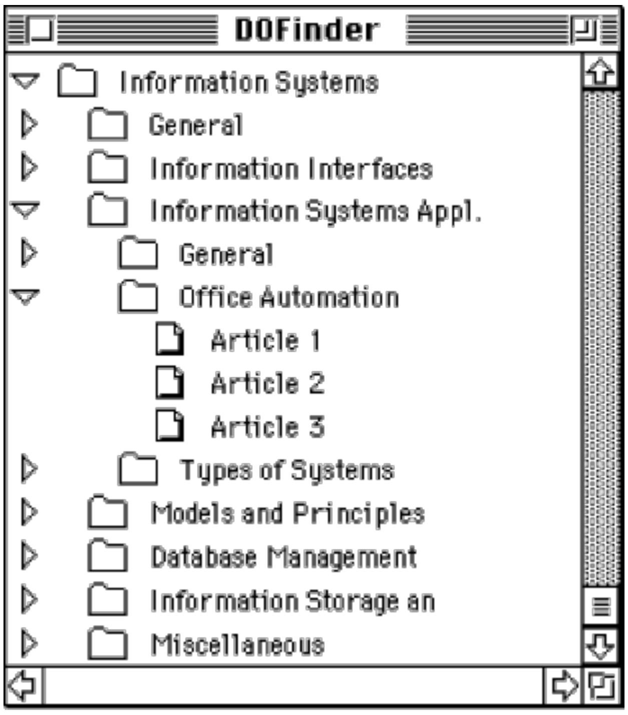


Figure 7: Folder hierarchy in HyperDoLL

Similar to components and links, attributes for nodes and view nodes can be defined by the author. A retrieval engine supports attribute search mechanisms and can be used as an additional means of navigation.

Up to now, HyperDoLL do not support the concepts of link structures and composite nodes but for instance theses concepts can be found in Intermedia and NoteCards. For further information about these systems the reader is referred to [Haan, Kahn et al. 92], [Yankelowich, Haan 88] and [Halasz 88], respectively.

4 Related Work

Our layered architecture described in section 2.1 (see also figure 1) mainly corresponds to the layer concept of the HOME system [Duval, Olivie et al. 95]. The Home development has been inspired by HM-Data Model [Maurer et al. 93], [Maurer et al. 94]. The HOME system differentiates between a raw data layer, a multimedia layer and a hypermedia layer. The raw data layer contains the raw multimedia data objects, i.e. a video, a bitmap etc. The multimedia layer manages meta data of isolated objects stored at the raw data layer. Finally, meta data about relationships between different objects are stored at the hypermedia layer. The raw data layer corresponds to the storage system of our architecture and the multimedia layer can be associated with the component level of our model. The hypermedia layer in the HOME system bases on a set-oriented approach. Even though multisets are allowed, we believe that our approach for modelling hypermedia objects is more general than the approach used in the HOME system.

The concept of views was caused by our work in related research domains. In particular, the second author of this paper also researches on structuring concepts in Petri nets, e.g. hierarchies and layers [Fehling 92], [Brodda, Buttler 90]. The structural parallels between Petri nets and hyperdocuments - basically both base on directed graphs - brought us to the question of whether these results could be adapted to hypermedia. In addition, other hypermedia approaches were taken into account: For example, there are ``structural'' views [Tochtermann, Dittrich 92] or [Bernstein, Bolter et al. 91] reflecting a hyperdocument's structure or parts of it. These views mainly serve to prevent users from experiencing disorientation. There are also approaches based solely on typed nodes or typed links [Creech, Freeze et al. 91], context nodes [Casanova, Tucherman 91], or on queries [Beeri, Kornatzky 90]. Finally, Tompa defines views by appropriate mappings in hypergraphs [Tompa 89]. However, one flaw shared by these approaches is that only entire nodes can be filtered from an underlying hyperdocument. Hence, we introduced the concept of view nodes allowing us to filter also portions of nodes. To our knowledge such a powerful view concept does not exist in another hypermedia system and model.

The concept of folders has its origins in the work of [Catlin, Garret et al. 91] although the efforts of the authors have not been for a theoretical model. We incorporated not only their ideas into our approach but also investigates new ideas for creating templates, c.f. [Tochtermann, Dittrich 95]. In addition, the implementation and visualisation of folders is quiet similar to the collections in Hyper-G. However, in contrast to the collection browser in Hyper-G, HyperDoLL supports the drag-and- drop paradigm. This allows users to easily re-organise a folder hierarchy.

There are other approaches dealing with retrieval operations in hypermedia, e.g. a logical query language presented in [Beeri, Kornatzky 90] permits queries resulting in a user-tailored view of the underlying hyperdocument, [Amann, Scholl 92] employ an algebraic query language basing upon a graph data model. Finally, a remarkable survey on retrieval in hypermedia can be found in [Arents 95].

5 Summary

In this section we list the new and borrowed concepts used in the DFHM. Finally we briefly summarise the advantages of our approach.

5.1 New and Borrowed Concepts

One aim of our research was not only to develop new hypermedia concepts but also to integrate existing and already well-accepted concepts. As we see it, the DFHM provides a new notion of

- the content of nodes and components,

- anchors,
- hyperdocuments,
- views and view nodes,
- folders.

We borrowed the following concepts from former approaches:

- nodes and composite nodes,
- links and link structures.

5.2 Advantages of the DFHM

- The DFHM is a family of formal models which is adaptable to different needs. Thus, different hypermedia systems can be described with different members of the DFHM. This is not possible with other formal models for hypermedia.
- Our approach is flexible and easy to extend. For instance, a model for hypermedia-based design of micro electronic systems is under current development. This model bases upon the DFHM and extends our concepts for the given application domain.
- The DFHM integrates structuring concepts, e.g. views, view nodes, folders, link structures, into the basic node-link model of hypertext.
- The DFHM provides typed hypermedia objects. Constraints between typed hypermedia objects are formalised by invariants [Kröcker 95].
- The DFHM is formally described using VDM and thus each concept is unambiguous [Tochtermann, Dittrich 95].
- The DFHM does not consist of a set of several independent concepts. Instead an integrated formalisation of hypermedia concepts is given.
- The DFHM provides a set of formally described operations acting upon the given concepts [Tochtermann 95a].
- Basing upon different members of the DFHM we developped two different hypermedia systems, HyperDoLL c.f. section 3.1, HyperMed [Tochtermann et al. 96], and one expertmedia system, HyperGIW. This indicates that the DFHM is powerful enough to cover a varity of different aspects of hypermedia systems.

6 Outlook

The modelling activities have been finished in August of this year. We believe that we have reached a point beyond which it does not seem profitable to further refine or extend the DFHM. Still, several concepts are not yet implemented and we direct our attention to the integration of these concepts in HyperDoLL:

We are currently working on the integration of an object oriented database (POET) for storage purposes of all hypermedia objects. The implementation will be finished by the end of February 1996. In addition, a visual programming language will be added by mid of March 1996. With this language we intend at providing further support for lecturers to organise the presentations for a lecture. For the future, we are planning to add concepts for cooperative hypermedia as they are known from SEPIA [Streitz et al. 92]. Finally, aspects of distributed hypermedia will be taken into consideration, c.f. [Duval, Olivie et al. 95].

References

[Amann, Scholl 92] Amann, B., Scholl, M.; "Gram: A Graph Data Model and Query Language"; Proc. of the 4th ACM Conference on Hypertext, Milano (1992), 201-211.

[Arents 95] Arents, H.; "Knowledge-based Indexing and Retrieval of Hypermedia Information"; in: [Rada, Tochtermann 95], 137-170.

[Beeri, Kornatzky 90] Beeri, C., Kornatzky, Y; "A Logical Query Language for Hypertext Systems"; Proc. of the European Conference on Hypertext, INRIA, The Cambridge Series on Electronic Publishing, (eds. A. Rizk, N. Streitz, J. André), Paris (1990), 67-80.

[Bernstein, Bolter et al. 91] Bernstein, M., Bolter, J.D., Joyce, M.J., Mylonas, E.; "Architectures for Volatile Hypertext"; Proc. of the 3rd ACM Conference on Hypertext, San Antonio (1991), 243-256.

[Brodda, Buttler 90] Brodda, A., Buttler, P.; "PetriLab, a Tool for Modeling and Simulation of Petri net based system descriptions" (in german); Diploma Thesis, Dept. of Computer Science, University of Dortmund, Germany, (1990).

[Campbell, Goodman 88] Campbell, B., Goodman, J.M.; "HAM: A General Purpose Hypertext Abstract Machine"; Communications of the ACM, 31, 7 (1988), 856-861.

[Catlin, Garret et al. 91] Smith Catlin, K., Garret, L.N., Launhardt, J.A.; "Hypermedia Templates: An Author's Tool"; Proc. of the 3rd ACM Conference on Hypertext, San Antonio (1991), 147-160.

[Casanova, Tucherman 91] Casanova, M., Tucherman,L.; "The Nested Context Model for Hyperdocuments"; Proc. of the 3rd ACM Conference on Hypertext, San Antonio (1991), 193- 201.

[Conklin, Begemann 88] Conklin, J., Begemann, M.L.; "gIBIS: A Hypertext Tool for Exploratory Policy Discussion"; ACM Transactions on Information Systems, 6, 4 (1988), 303- 331.

[Creech, Freeze et al. 91] Creech, M., Freeze, D., Griss, M.L.; "Using Hypertext In Selecting Reusable Software Components"; Proc. of the 3rd ACM Conference on Hypertext, San Antonio (1991), 25-38.

[Dalitz, Heyer 95] Dalitz, W., Heyer, G.; "Hyper-G - Das Internet-informationssystem der 2. Generation"; dpunkt Verlag für digitale Technologie GmbH, Heidelberg, (1995).

[Duval, Olivie et al. 95] Duval, E., Olivie, H., Hanlon, P., Jameson, D.; "HOME: An Environment for Hypermedia Objects"; J.UCS (Journal for Universal Computer Science), 1, 5 (1995), 265-287.

[Fehling 92] Fehling, R.; "Hierarchical Petri nets" (in german); PhD Thesis; Verlag Dr. Kovac, Hamburg, Germany (1992).

[Flohr 95] Flohr, U.; "Hyper-G Organizes the Web"; Byte, November 1995, 59-64 .

[Garzotto, Paolini 92] Garzotto, F., Paolini, P., Schwabe, D.; "HDM - A Model for the Design of Hypertext Applications"; Proc. of the 3rd ACM Conference on Hypertext, San Antonio (1991), 313-328.

[Grønbaek 94] Grønbaek, K.; "Composites in a Dexter-based Hypermedia Framework"; Proc. of the ACM Conference on Hypermedia Technology, Edinburgh (1994), 59-69.

[Haan, Kahn et al. 92] Haan, B., Kahn, P., Riley, V., Coombs, J., Meyrowitz, N.; "IRIS Hypermedia Services"; Communications of the ACM, 35, 1 (1992), 36-51 .

[Halasz 88] Halasz, F.; "Reflections on NoteCards: Seven Issues for the next Generation"; Communications of the ACM, 31, 7 (1988), 836-852.

[Halasz, Schwartz 90] Halasz, F., Schwartz, M.; "The Dexter Reference Model"; Proc. of the Hypertext Standardization Workshop, Nat. Institute of Standards and Technology Publication 500-178, Gaithersburg, (1990), 95-133.

[Halasz, Schwartz 94] Halasz, F., Schwartz, M.; "The Dexter Reference Model"; Communications of the ACM, 37, 2 (1994), 30-39.

[Hammwöhner, Kuhlen 94] Hammwöhner, R., Kuhlen, R.; "Semantic control of open hypertext systems by typed objects"; Journal of Information Science, 20, 3, (1994), 175-184.

[Hardman, Bulterman 94] Hardman, L., Bulterman, D., van Rossum, G.; "The Amsterdam Hypermedia Model: Adding Time and Context to the Dexter Model"; Communications of the ACM, 37, 2 (1994), 30-39.

[Jones 90] Jones, C.B.; "Systematic Software Development using VDM"; Prentice Hall (1990).

[Kröcker 95] Kröcker, C.; "Formal Specification of Types in a VDM-Model for Hypermedia" (in german); Diploma Thesis, Dept. of Computer Science, University of Dortmund, Germany, (1995).

[Lange 90] Lange, D.; "A Formal Approach to Hypertext using Post-Prototype Formal Specification"; D. Bjørner, C.A.R. Hoare, H. Langmaack (eds.) VDM '90 VDM and Z - Formal Methods in Software Development, LNCS 428, Springer, Heidelberg/New York (1990), 99- 134.

[Marchionini, Maurer 95] Marchionini, G., Maurer, H.: "The Roles of Digital Libraries In Teaching and Learning"; Communications of the ACM, 38, 4, (1995), 67-75.

[Maurer et al. 93] Maurer, H., Scherbakov, P., Srinivasan, P.: "A new Hypermedia Data Model"; Proc. of the Fourth International Conference on Database and Expert System Applications, Prague (1993), LNCS No.720, 685-696.

[Maurer et al. 94] Maurer, H., Philpott, A., Scherbakov, P.: "Hypermedia systems without links"; Journal of MCA 17,4 (1994), 321-332.

[Maurer 95] Maurer, H.; Prelimary Copy of "Hyper-G - The Second Generation Web Solution", (to appear) Addison-Wesley (1996).

[Nanard, Nanard 91] Nanard, J., Nanard, M.; "Using Structured Types to incorporate Knowledge in Hypertext"; Proc. of the 3rd ACM Conference on Hypertext, San Antonio (1991), 329-343.

[Nanard, Nanard 93] Nanard, J., Nanard, M.; "Should Anchors be typed too? An Experiment with MacWeb"; Proc. of the 5th ACM Conference on Hypertext, Seattle, Washington (1993), 51-62.

[Neubert 93] Neubert, S.; "Model Construction in MIKE (Model Based and Incremental Knowledge Engineering)"; Proceedings of the 7th European Workshop on Knowledge Acquisition for Knowledge-Based Systems, (eds. N. Aussenac, G. Boy et al.), Springer, Lecture Notes in Artifical Intelligence No. 723 (1993), 200-219.

[Rada, Tochtermann 95] Rada, R., Tochtermann, K. (eds.); "ExpertMedia - Expert Systems and Hypermedia"; World Scientific Publishing, London, England (1995).

[Streitz et al. 92] Streitz, N., Haake, J., Hannemann, J., Lemke, A., Schuler, W., Schtt, H., Thüring, M.; "SEPIA: A Cooperative Hypermedia Environment"; Proc. of the 4th ACM Conference on Hypertext, Milano (1992), 11-22.

[Tochtermann 95a] Tochtermann, K.; "A Formal Model for Hypermedia: Description and Integrated Formalisation of Essential Hypermedia Concepts" (in german); Dissertation (PhD Thesis); Dept. of Computer Science, University of Dortmund; Verlag Shaker, Aachen, Germany, (1995).

[Tochtermann 95b] Tochtermann, K.; "ExpertMedia and Information Systems"; in: [Rada, Tochtermann 95], 29-60.

[Tochtermann, Dittrich 92] Tochtermann, K., Dittrich, G.; "Fishing for Clarity in Hyperdocuments with Enhanced Fisheye-Views"; Proc. of the 4th ACM Conference on Hypertext, Milano (1992), 212-221.

[Tochtermann, Dittrich 95] Tochtermann, K., Dittrich, G.; "Towards a Family of Formal Models for Hypermedia"; Proceedings Hypertext - Information Retrieval - Multimedia Conference (HIM '95), (eds.) R. Kuhlen, M. Rittberger, Constance (1995), 77-91.

[Tochtermann et al. 96] Tochtermann, K., Tresp, C., Hiltner, J., Reusch, B., Freund, A., Weidemann, J., Hohn, H.-P., Denker, H.-W.; "HyperMed: A hypermedia system for anatomical education"; to be published in Proceedings of AACE World Conference on Educational Multimedia/Hypermedia, Boston, USA, Juni 1996.

[Tochtermann, Zink 95] Tochtermann, K., Zink, V.;"Artificial Intelligence and Hypermedia"; in [Rada, Tochtermann 95], 61-90.

[Tompa 89] Tompa, F.; "A Data Model for Flexible Hypertext Database Systems"; ACM Transactions on Information Systems, 7, 1 (1989), 85-100.

[ToolBook 92] Multimedia ToolBook - The Graphical Development Kit For Multimedia PC Applications, Version 1.53 for Windows 1992; Asymetrix Corporation Washington.

[Wang, Rada 94] Wang, W., Rada, R.; "A Composition Schema for Reusable Hypertext"; CD- ROM Edition - Proceedings of the 2nd World Congress on Expert Systems, (Ed. Jay Liebowitz), Lisbon (1994).

[Yankelowich, Haan 88] Yankelowich, N., Haan, B.: "Intermedia: The Concept and the Construction of a Seamless Information Environment"; IEEE Computer, (1988), 81-96.