A Framework for Generating Adaptable Hypermedia Documents

Lloyd Rutledge, Jacco van Ossenbruggen, Lynda Hardman, and Dick C.A. Bulterman*

CWI (Centrum voor Wiskunde en Informatica)
P.O. Box 94079
1090 GB Amsterdam, The Netherlands
Tel: +31 20 592 41 27
E-mail: {lloyd,lynda,dcab}@cwi.nl

*Vrije Universiteit
de Boelelaan 1081a
1081 HV Amsterdam, The Netherlands
Tel: +31 20 444 77 81
E-mail: jrvosse@cs.vu.nl

ABSTRACT

Being able to author a hypermedia document once for presentation under a wide variety of potential circumstances requires that it be stored in a manner that is adaptable to these circumstances. Since the nature of these circumstances is not always known at authoring time, specifying how a document adapts to them must be a process that can be performed separately from its original authoring. These distinctions include the porting of the document to different platforms and formats and the adapting of the document's presentation to suit the needs of the user and of the current state of the presentation environment. In this paper we discuss extensions to our CMIF hypermedia authoring and presentation environment that provide adaptability through this distinction between authoring and presentation specification. This extension includes the use of HyTime for document representation and of DSSSL for presentation specification. We also discuss the Berlage architecture, our extension to HyTime that specifies the encoding of certain hypermedia concepts useful for presentation specification.

KEYWORDS

Hypermedia, Open Hypermedia, HyTime, DSSSL, SGML.

1 INTRODUCTION

Our research group has investigated the issues involved with hypermedia documentation and presentation with the development of the *CWI Multimedia Interchange Format (CMIF)*, and the *CMIFed* environment for the creation and playback of CMIF hypermedia documents [8]. CMIF encodes crucial hypermedia functionalities such as synchronization, hyperlinking, screen display layout, and presentation style. Our main goal for CMIF is that it be adaptable to individual presentation situations. The model for CMIF documents is applicable to a wide variety of hypermedia presentation circumstances, including different document formats, variations in platforms and software, available resources, user characteristics, and desired style of presentation. To facilitate this adaptability, we have developed an encoding for CMIF documents in HyTime, a standard for the interchange of hypermedia documents [4,10]. HyTime provides an encoding format, a general hypermedia model, and facilities for transformations to other formats and to variations in document presentation.

Copyright © 1997 by the Association for Computing Machinery, Inc. Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Publications Dept, ACM Inc., fax +1 (212) 869-0481, or permissions@acm.org.
ACM Multimedia 97, Seattle WA USA
©ACM

This paper presents the HyTime encoding for CMIF. It also describes the extension to the CMIF environment for processing this encoding and the presentation instructions that accompany it. This translation provides an empirical test of HyTime's ability to represent the hypermedia structure of existing environments. It also provides an opportunity to explore the issues behind processing this generic structure for actual presentation on interactive multimedia environments. The HyTime encoding of CMIF uses our Berlage architecture, an SGML-defined extension of HyTime that encodes how a document's HyTime-defined structure is mapped to certain aspects of presentation processing.

First some background material on the models and formats involved is given. Then we present the architecture resulting from the integration of these topics into the CMIF environment. Finally, our use of HyTime and the Berlage architecture to encode the hypermedia structure of CMIF is described.

2 MODELS AND FORMATS FOR ADAPTABLE HYPERMEDIA

In this section we describe existing models, standards, and formats that apply to adaptable hypermedia and have been used in the development of CMIF. Then we present an analysis of how these different models relate to each other in terms of their adaptability. This is followed by a general hypermedia model based on the models and formats described that is a basis for later discussion in this paper.

The CMIF document format, described in Section 1, has been developed in terms of the *Amsterdam Hypermedia Model (AHM)* [8]. The AHM extends the *Dexter Hypertext Reference Model* [7] into multimedia by adding to it concepts such as the synchronization of multimedia object display and more complex specifications of how link activation affects document presentation. The Dexter model specifies significant abstractions shared by typical hypertext systems to provide a reference for discussing and comparing these systems. It is designed to aid in the general discussion of different hypertext documents and applications. It defines a layered model onto which individual hypertext systems can be mapped. It also defines common terms to be uniformly applied to the hypertext systems when they are described by Dexter.

While the AHM provides a conceptual basis for adaptable hypermedia documents, it does not define a specific format or specify the means of processing documents for presentation. To provide an open process model for documents specified in CMIF we have adopted the use of ISO standards designed to encode hypermedia document models and their presentation in distinct circumstances [13,14]. *Hypermedia/Time-based Structuring Language (HyTime)* [4,10] is a standard for encoding hypermedia documents and the models they conform to in terms of widely-applicable hypermedia concepts. The hypermedia document structure HyTime represents includes complex hyperlinking, locating of document objects, and the scheduling of objects within measured coordinate systems such as space and time. These constructs constitute a general model for hypermedia documents and the sets to which they belong.

HyTime builds upon and extends into hypermedia the presentation-independent text structuring of *Standard Generalized Markup Language (SGML)* [6,11]. SGML is a widely used ISO standard for representing electronic text documents. It supports and facilitates document exchange, reuse and longevity by representing a document in its permanently stored form without defining its processing into any particular presentation.

SGML documents are divided up into *elements*, which can contain media for presentation or other elements. Each element has a collection of *attributes*. One type of attribute is the *unique identifier reference (IDREF)*, which references another element in the document. SGML provides for the definition of separate document sets, called *document type definitions (DTDs)*, which represent different types of documents used by separate communities. A DTD defines a general structure that a class of documents fit. This structure specifies the types of elements that can exist for a document set and what attributes each of those elements can have. An individual SGML document is an instance of the DTD associated with it. HyTime inherits SGML's ability to define distinct document sets. HyTime, like SGML, requires external mechanisms for specifying the presentation of its document.

The *Berlage architecture* is an extension to HyTime we developed. It preserves HyTime's ability to define distinct document sets. It extends HyTime by providing additional constructs that help guide the processing of HyTime-defined presentation-independent constructs into final presentations.

Document Style Semantics and Specification Language (DSSSL) [9] is a standard that encodes the translation of SGML, and thus HyTime, documents into other forms of output. DSSSL defines a system independent way to describe the processing of SGML documents of a given DTD into those of another DTD or into various forms of page-based output. DSSSL specifies this formatting process by defining a powerful document query language, a document transformation language, and a style language to apply a set of formatting characteristics to portions of a document. A DSSSL mapping is encoded in a *style sheet*, which is applied to a document or document set to specify its mapping into one possible style of presentation.

2.1 Adaptability of Document Models

The specification of adaptable document models has long been accomplished for text by the SGML community. This community has also specified how documents adapt to particular presentation circumstances using document conversion specifications such as DSSSL. Similar presentation-independent modeling with separate presentation mapping specifications has been explored for multimedia with the use of relational grammars [15]. This mapping of the authoring and presentation of multimedia documents has also been defined in terms of relative constraints placed on the measured aspects of a document's presentation [3]. HyTime and the property set it defines for use with DSSSL establishes formats for extending SGML and DSSSL from text into hypermedia for both document modeling and the adaptation of documents for presentation. This HyTime/DSSSL-based approach and the requirements for extending current SGML-only DSSSL into HyTime-defined structure for the creation of hypermedia style sheets have been explored in earlier work [12].

Many different models, formats and encodings for hypermedia documents exist, and they vary in how they can adapt to individual presentation circumstances. One analysis of how these models vary in their adaptability is shown in Figure 1. Markup and meta-markup languages, abstract models and document encodings are mixed in one figure, and, for brevity, are all regarded as document models in the following discussion.

The horizontal axis indicates hypermedia functionality. It runs from linear, text-only on the left to fully hyperlinked multimedia on the right. The vertical axis indicates presentation-independence. The models allowing descriptions of fully formatted presentations are towards the bottom. The more abstract models, describing the logical structure of the contents without presentation-specific details are towards the top.

Several important characteristics of document models can be derived from their position in the figure. First of all, we note that documents of the models close to the bottom are ready for presentation: they contain all the information needed to display the document on a specific medium. The distance from a particular model to the horizontal axis indicates the amount of information that needs to be added in order to present a document conforming to that model. The more abstract models have more involved processes for transformation of their documents to display or print.

Second, we observe that documents close to the bottom may contain more presentation information, but they usually contain presentation information for only one specific medium. As a consequence, these models are not suited in environments where documents are to be rendered on different output media. If the advantages of flexibility outweigh the drawbacks of the more complex formatting and presentation process, one should choose for a more abstract model.

In practice, choosing a single model for a particular application is often not sufficient. One needs both the flexibility of the more abstract models and the accurate control over the presentation offered by the models closer to the horizontal axis. As a result, one needs tools that can convert documents from one model into the other.

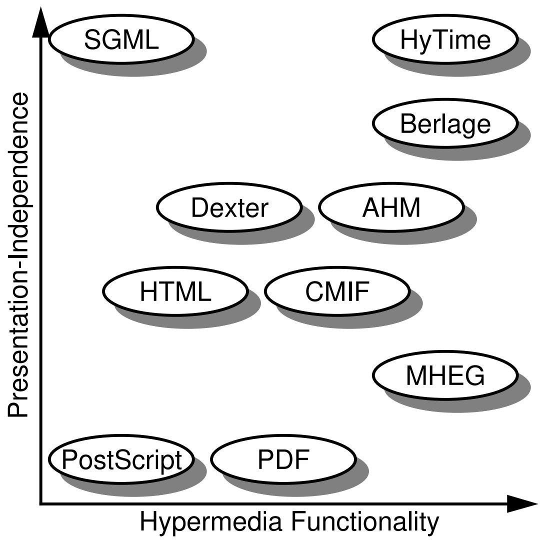

Figure 1: Chart of Document Models

Several types of translations may be needed, and usually involve combinations of the ones listed below. The most common, the translation from an abstract model into a presentation model, involves the association of specific presentation concepts of the target model with the abstract, logical concepts in the original model. This association is often wired into an application or defined in a style sheet.

The reverse process is usually more complicated since one needs to recognize the logical structure of the content on the basis of layout information. This type of conversion is very hard to do automatically, but is often needed to convert legacy data into the more abstract model of a new system. It is also difficult to convert from a text-based model to a model with more hypermedia support.

Conversion between two models that are positioned on more or less the same position is needed to allow document interchange between comparable programs with incompatible document formats. Usually, these conversions are relatively easy because they convert between similar models, and the conversion is mainly syntactic.

The CMIF document model holds a balance between presentation-independence and presentation-specificity. Its degree of presentation-specificity comes from its placement of media objects in positions on a windowed screen display and along a timeline. Because of the degree of presentation-independence it maintains due to its adaptability, CMIF's documents can be rendered to different hypermedia presentation formats, such as MHEG, in a variety of presentation styles. Documents of a strongly presentation-independent structure could be converted to CMIF to specify a general layout of their presentation and then further style sheets could define varying styles and formats for this presentation.

2.2 A General Hypermedia Model

In this section we describe a general model for hypermedia processing based primarily on the Dexter and AHM models but also taking into account the processing model used for SGML, HyTime, and some related formats. We use this model later in the paper for associating a layout with the document structure defined by our CMIF DTD. This model is also used to describe the scope of the Berlage architecture. The Dexter/AHM model and its relation to these SGML-based formats is illustrated in Figure 2. The Berlage document architecture, which is included in this diagram, is described in Section 5.

In SGML terminology, a document is separated into *content*, *structure*, and *presentation*. The *content* makes up the individual perceivable components of the document. The *presentation* is how the document appears to the user. SGML focuses on document *structure*, which specifies the relations between the content components that affect their potential final presentations without specifying any particular means of presenting them.

Dexter/AHM divides hypertext processing into three layers: the *run-time*, *storage*, and *within-component* layers. The run-time layer is concerned with the final presentation of documents, including how the user interacts with the interface and what activities occur during presentation. This layer makes up the user's perception of the document. The within-component layer contains the atomic containers of the media objects inside a document. This layer represents the individual items contained in the document. The storage layer is related to the within-component layer through *anchoring*, which enables media type-independent linking to portions of atomic media content. The run-time and within-component layers of the Dexter/AHM model are covered minimally, with the focus being on the remaining components of the model.

Because HyTime is strictly general-purpose and presentation-independent, it specifies nothing about what would fall in the run-time layer.

The Dexter/AHM storage layer provides the hypermedia structure of a document. This includes hyperlinking and the compositing of document objects from atomic media components. The HyTime hyperlinks module establishes non-hierarchical and non-linear relationships between document components. The structures defined by this module correspond very closely with links in the Dexter/AHM storage layer. SGML and the HyTime base module define several types of composites. We refer to the information of this layer as the *generic hypermedia structure* of the document.

AHM extends the Dexter model storage layer by introducing composites for defining synchronization that are based on temporal and atemporal composition [8]. *Channels* are introduced, which provide independently controlled presentation of document components. The HyTime scheduling module performs channel-like compositing by grouping objects for coordinated positioning in measured systems such as screen displays and timelines.

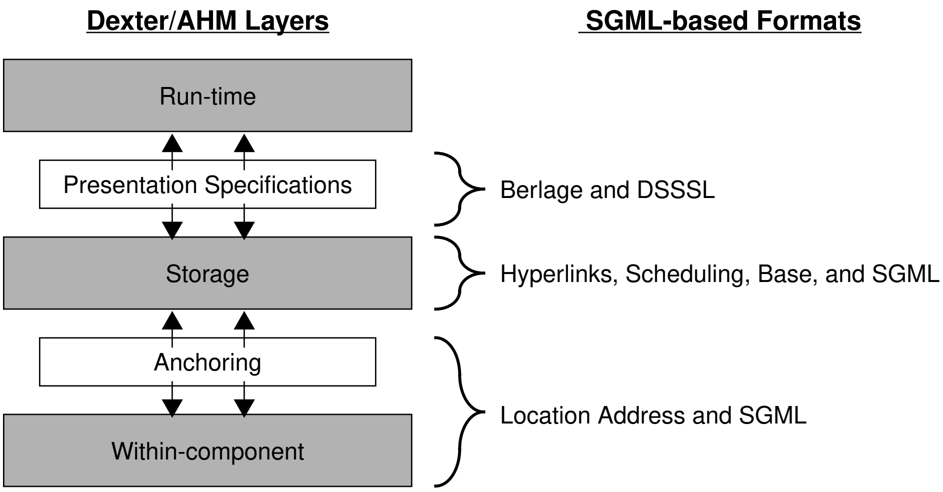

Figure 2: The Dexter/AHM Layers and Corresponding SGML-based Formats

A document's generic hypermedia structure cannot by itself be presented to a user. This structure must accompany a mapping that specifies how that structure is translated into a particular presentation. This mapping references aspects of the document's generic hypermedia structure to determine the document's appearance to the user. The mapping may also reference aspects of the document's presentation environment to determine the best means of displaying the document under the conditions given. This functionality corresponds with the Dexter/AHM *presentation specification* layer, which defines how the contents of the storage layer are to be processed when the user browses through them. DSSSL performs actions of this layer because it specifies how document structure is to be translated into presentation under particular circumstances. HyTime does not address presentation-specification because no HyTime constructs refer to presentation-time document characteristics. An extension to HyTime for representing presentation specification information appropriate for storage in the document itself is described in Section 5.

3 AUTHORING AND PRESENTATION WITH CMIF

This section discusses how CMIFed is used as an editor for HyTime-conforming documents and how DSSSL can be used to specify the adaptation of these documents to individual presentation situations. The design of the application of HyTime and DSSSL to CMIF processing is illustrated in Figure 3.

The contribution that HyTime-encoding makes beyond the use of SGML and DSSSL is the representation of common hypermedia properties in a standardized form. SGML provides a parsable syntax for representing the document's content and structure in a presentation- and processing-independent way. DSSSL provides a language for specifying formatting and other transformations of SGML documents. DSSSL code refers to SGML-defined syntax constructs to specify how the documents are to be processed as input. HyTime specifies certain composites of SGML constructs to be recognized as having hypermedia properties. DSSSL code can refer to these HyTime properties of a document and specify how the hypermedia structure they define is to be converted to a particular hypermedia format.

The first step in the process described in this section is the authoring of CMIF documents with the CMIFed authoring tool. The document is then processed by the CMIF to HyTime converter, generating a HyTime-conforming SGML document as output. This document conforms to the DTD that defines the HyTime-conforming CMIF document set, which is shown in Figure 5 and described in Section 4. The DTD and the resulting document can be processed by other SGML and HyTime systems.

The second step for the user is generating presentations of HyTime-encoded CMIF documents. If a style sheet exists, this step can be taken with no interaction from the user. Otherwise, one must be created. A DSSSL style sheet is processed with the document into its equivalent in other formats. Style sheets can be written by and the conversion can be performed by users other than the author of the original document.

3.1 Porting CMIF to Other Applications

Portability of documents is important not just between platforms but also between applications. CMIF documents are presentable on a wide variety of platforms because CMIF presenters have been developed for each of these platforms. However, on each platform a CMIF presenter is still required for viewing the document. It is a problem when a CMIF player is not available on that computer or if the user has more familiarity with another hypermedia presenter. This section discusses how the application of HyTime and DSSSL to this task can specify the translation of CMIF documents to the formats of other browsers.

The most straightforward way to present a CMIF document on a non-CMIF application is to convert the document to that application's format. All CMIF documents can be made portable to a given application by developing a converter from CMIF to its equivalent in the other format. Developing this tool would be a cumbersome task for those not directly familiar with CMIF.

Encoding CMIF documents as HyTime makes it easier for developers to create converters from CMIF to formats of other players. Developers could use DSSSL style sheets to encode the format conversion instructions for processing the HyTime version of the document with DSSSL tools into the desired output format.

3.2 Adapting CMIF Documents for Presentation Environments

There are variations that exist between environments that must be adapted to in presented documents. Section 3.1 discusses the adaptation to one such variable: the hypermedia document presenters that are available. Other variations between environments exist as well. These include the available peripherals for media object presentation and the available bandwidth for the transmission of distributed document data. This section presents

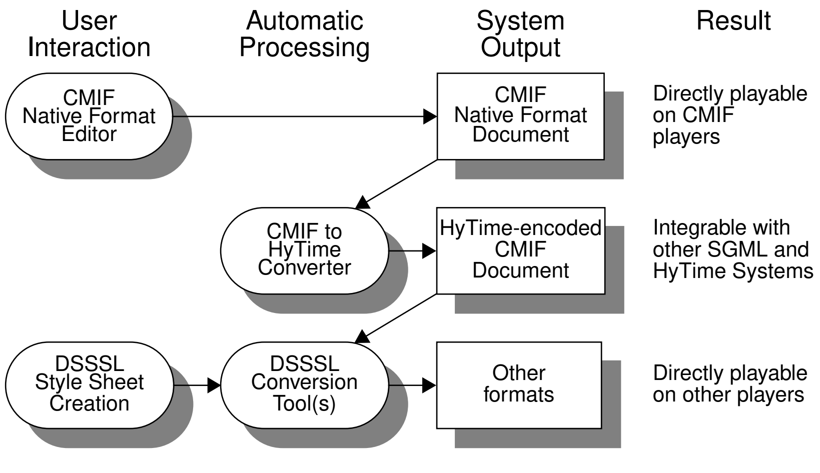

Figure 3: A Framework for CMIF Document Processing

how HyTime and DSSSL are able to specify the way document presentations are adapted to these variations.

Our CMIF DTD allows alternative formats to be specified for the display of a presentable document object. A single document object may have assigned to it files of different formats. Upon presentation of the object, the file of the format for which the appropriate software or peripheral is available in the environment can be selected to be processed for presenting that object. Files of formats for different media types could also be assigned to one media object for situations where no tools for presentation of a particular media type are available.

Similarly, a CMIF document can be authored as adaptable to variations in the available network bandwidth, addressing issues such as quality of service and graceful degradation. If the transmission of a particular media file for presentation can be anticipated as taking too long then an alternative smaller file conveying the same information, though with less quality, can be transmitted instead in an acceptable time frame. Another alterative could be to send a file of a different media type that uses less bandwidth. For example, an image could be replaced with a text description of the image, taking up much less bandwidth but still conveying information represented by the document object.

In our design a DSSSL style sheet could be written that refers to this presentation environment information in determining which of alternative files to select for presenting a particular media object. The peripherals available on the current environment could be queried to determine which of the alternative file formats of a media object is used for its presentation. Similarly, the available bandwidth and anticipated transmission speed could be queried to determine what size file should be transmitted for presenting a particular media object. In the latter case, the author of the style sheet could specify the importance of transmission speed versus the quality of presentation of each media object.

DSSSL does not provide constructs for querying these aspects of a presentation environment. It does, however, provide mechanisms for the creation of extensions of what can be queried for in converting documents. For CMIF, such an extension would be created to allow the presentation environment to be queried against.

4 THE HYTIME ENCODING OF CMIF

In this section we discuss the use of HyTime in defining a document model and format for CMIF. This discussion is illustrated with an example of a hypermedia document authored and presented with CMIFed, a screen dump of which is shown in Figure 4. The DTD for the HyTime-encoding of CMIF documents is illustrated in Figure 5. By representing CMIF hypermedia document structure with HyTime constructs, CMIF is mapped to a common hypermedia model whose properties can be queried against in DSSSL style sheets. An in-depth knowledge of how CMIF encodes hypermedia concepts would not be needed to access its hypermedia structure. This HyTime encoding also provides an empirical test of HyTime and its ability to represent hypermedia functionality present in an implemented hypermedia environment.

Similar empirical studies have been performed on HyTime's ability to represent the hypermedia semantics of an existing document format with the HyOctane project [1]. Here we describe HyTime's ability to encode the hypermedia semantics particular to CMIF.

First the example CMIF document is introduced. Then, we divide hypermedia into separate areas and discuss the constructs from each area in turn. These areas are *content*, *hyperlinking*, *composition*, and *measured positioning*. For each area, we describe the HyTime and CMIF constructs involved. We also discuss how

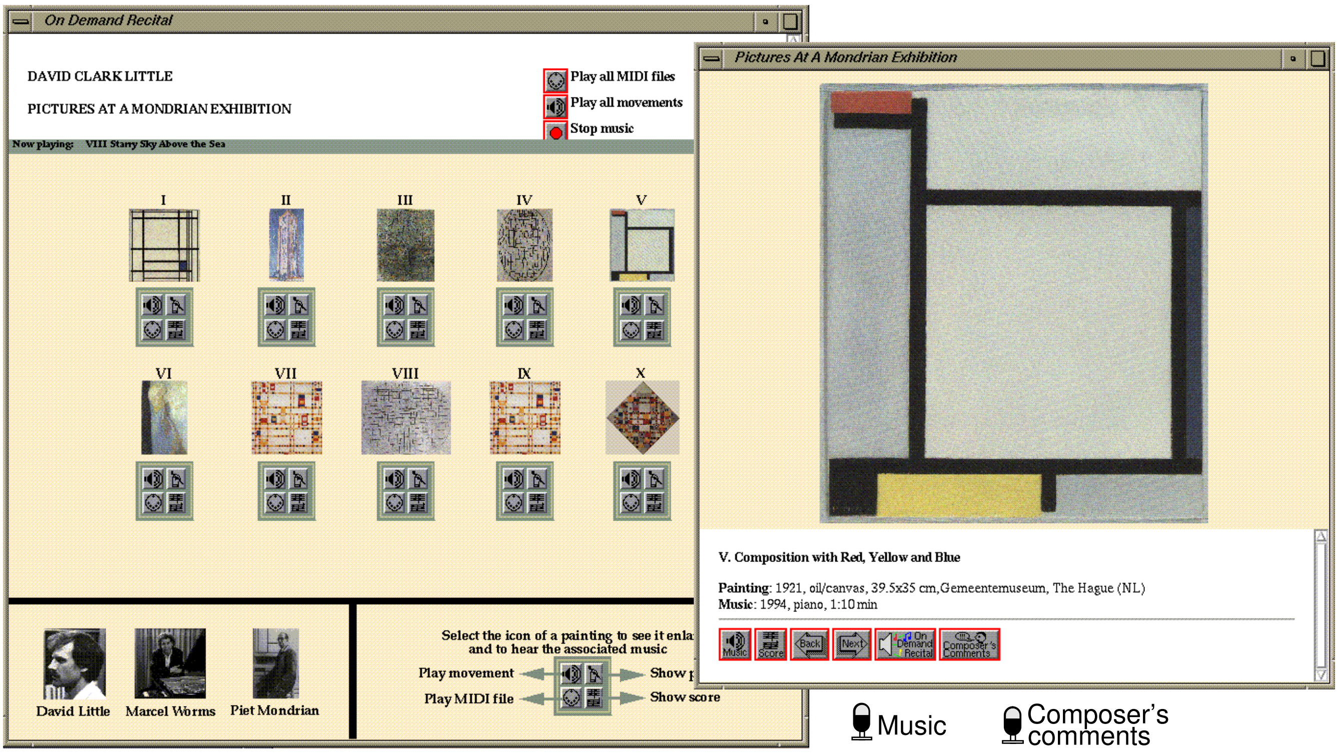

Figure 4: Display of Example CMIF Document Presentation

these constructs address either the presentation-independent or presentation-specific issues of their use within the format.

4.1 CMIF Document Example

To demonstrate the authoring and presentation of CMIF documents we have developed an application entitled *Pictures at a Mondrian Exhibition*. The application has a visual, musical, and informative aspect. The visual aspect consists of images of 10 paintings from the exhibition *Mondrian from Figuration to Abstraction*, shown in the Gemeentemuseum, the Hague, from February 20th to May 29th, 1988. The musical aspect consists of 10 piano compositions written by David Little and played by Marcel Worms, with each composition written for one of the paintings. The informative aspect consists of text and oral descriptions of the paintings, the compositions and the artists involved. The document, and the user's interaction with its display, coordinates the presentation of objects from these three aspects.

A screen dump from a presentation of this document is shown in Figure 4. The larger window, present when the presentation is in "On Demand Recital" mode, gives the user a high-level navigational interface, showing thumbnails of the 10 paintings and

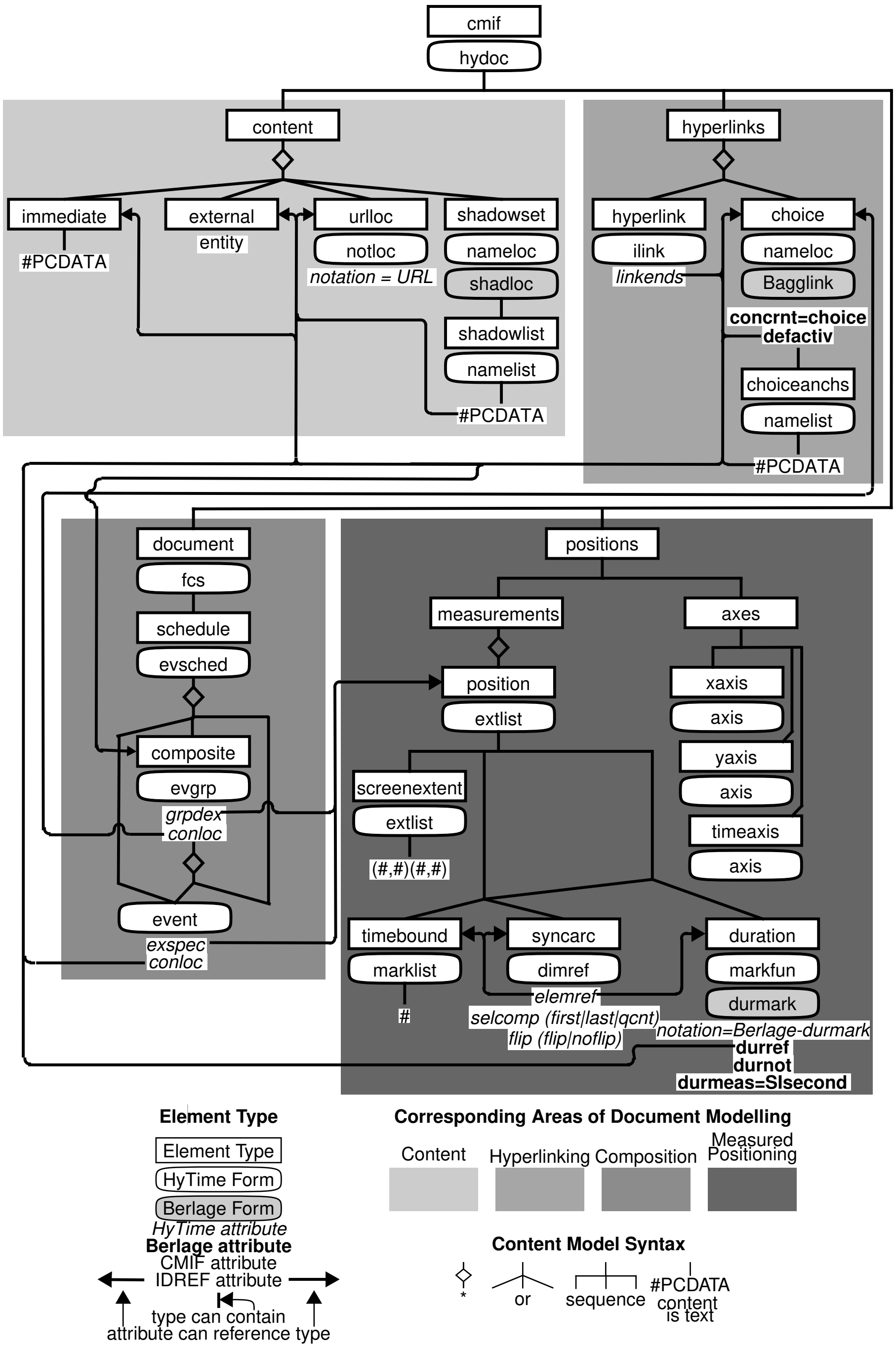

Figure 5: The HyTime DTD for CMIF

allowing each to be selected for presentation. The smaller window presents one painting at a time. Typically the composition for a painting is played in conjunction with its display in this window, though user interaction can override this. This second window has buttons for playing the composition, showing the score, and accessing the composer's comments on each painting and composition. During "On Demand Recital" mode, the user can select any painting at any time for display from the larger window. The user can also request a "concert", in which the paintings and their compositions are presented sequentially without further interaction.

4.2 Content

In a file making up a single hypermedia document, much of what specifies the media content shown in any presentation of the document is referenced as external files. However, some of what gets shown in a presentation can be stored directly in the document file itself. CMIF provides constructs for including both of these types of content.

In the Mondrian document, the content consists of painting images, composition sound files, text commentary, oral commentary sound files, postscript scores, images for the artists involved, and images for the interface buttons. All of these are stored as external files on the local file system.

An integral part of the SGML format is this direct containing of text data in the document itself. Every element in an SGML document can contain other elements or straight text. HyTime inherits this characteristic. The most direct way to incorporate text into an SGML or HyTime document is in this fashion. In CMIF, the *immediate* construct is used to include text directly in the document file. Its HyTime encoding includes the text as direct content. As media content itself, this construct fits in the Dexter/AHM within-component layer

An SGML *entity* is a string that can be processed by a local file server to return a file's contents. SGML entities handle the locating of external files in their entirety. The *external* construct of the CMIF HyTime encoding uses an SGML entity to specify a file on the local system as a media object.

The location capabilities of the SGML entity are extended by the HyTime *notation location (notloc)*. The contents of a notloc are processed to return some data and correspond again with the address field. A notloc also specifies a *notation* for its address text string to determine how it is to be processed to return the data. The CMIF HyTime encoding uses notloc to specify files located using a URL in its *urlloc* construct. The contents of a urlloc are a string in URL notation specifying the located file.

4.3 Composition

The CMIF document model defines compositions to coordinate the timing of the display of document components. A *parallel* composite specifies that its components are to be displayed simultaneously. A *sequential* composite specifies that its components are to be displayed in order, one after the other. A *choice* composite specifies that at most one of its components is to be displayed at any point in the presentation. That component is the one most recently activated as the endpoint of a hyperlink.

In the Mondrian document, the simultaneous display of the 10 thumbnails in the larger window is represented as the node for each thumbnail being a child of a parallel composite. The presentation during "concert" mode is represented with a sequential composite, with each of its children being a node for the presentation of a painting and its composition in the smaller window. A choice composite is used to represent the contents of the second window during "On Demand Recital" mode, which displays the painting that was most recently selected by the user.

The HyTime scheduling module groups objects for coordinated positioning within numeric systems. While in presentation-oriented hypermedia formats such as CMIF coordinate systems are typically measured in terms of time and screen display space, HyTime can use any set of dimension types and any number of axes for its coordinate systems. Upon presentation, these generic schedules can be mapped to screen display coordinates and timelines of typical computer interfaces. They can also be mapped to other aspects of the presentation.

The HyTime *finite coordinate space (fcs)* and *event schedule (evsched)* forms are used to define the timeline and two-dimensional screen display used by CMIF. The HyTime *event group (evgrp)* form collects media objects and other evgrps to be collectively positioned in one area of an event schedule. The positioning of its contents are in terms of the bounds established for an evgrp. The HyTime evgrp is used to represent the parallel, sequential, and choice composites used by the CMIF authoring interface. In the HyTime DTD, all three types of composites are represented as a *composite* element type. A choice composite is distinguishable from the others by its HyTime *content location (conloc)* attribute, which references a choice element located elsewhere in the document to specify its contents rather than having them directly placed within it.

The HyTime *event* form defines a position in the schedule for an individual media object. It is used by the CMIF DTD *event* element type. The HyTime *extent specification (exspec)* attribute references the element defining the values representing this event's position in the schedule. The conloc attribute is used here to reference the content element specifying the media object displayed by this event.

4.4 Hyperlinking

Here we discuss hyperlinking in the navigational sense. CMIF navigation is defined by the *hyperlink* construct. With these, a mouse click on a word or image changes the presentation and the information displayed in the manner common in hypermedia interfaces.

The primary navigational hyperlinks in the Mondrian document relate the selection of each thumbnail with the presentation of its painting. When a thumbnail is selected, the larger window stays the same while the smaller window displays the selected painting and the recording of the that painting's composition is played. The presentations of the image and composition active at the time of the thumbnail selection are stopped and replaced with those of the selected painting.

HyTime considers hyperlinking not as navigation but more generally as the establishment of relationships or associations between document objects. A HyTime hyperlink joins together a collection of *anchors*. It specifies that each anchor plays some role in an association or relationship with the other anchors. HyTime hyperlinking constructs define allowable *directions of traversal* between anchors in a hyperlink. Here traversal is defined not just as navigation but as any processing of the link's representation of the link endpoints under consideration. For each anchor in a link, it can be specified whether or not traversal to or from the anchor through that link it allowed. With this specification, it can be said for each pair of anchors in a hyperlink whether or not traversal can start at the first anchor and go to the second through that hyperlink. In such a case, the first anchor would be the *starting anchor* and the second would be the *ending anchor* for the pair.

In our DTD we assume an additional navigational semantics beyond what HyTime specifies for hyperlinks. The CMIF hyperlink is represented by the HyTime form *independent link (ilink)*. The HyTime definition for a hyperlinks specifies that each has a source and a destination for traversal.

The CMIF hyperlink determines how the user initiates a navigational traversal. How the display changes, and how it remains the same, after a traversal is determined by the CMIF choice composite, described in Section 4.3, and its position in the document hierarchy. When a document object is traversed to, that object will not necessarily occupy the entire screen display. There may be other document objects displayed with it that are maintained on the screen from before the user's activation of that hyperlink traversal. Similarly, a sound being played when the user activates a hyperlink may or may not continue after the ending anchor is traversed to.

At most one child of a choice node appears at any time during a presentation. One child can be established as the initial object displayed. When the user activates one child as the ending anchor of a hyperlink traversal, all the other children and their contents are no longer shown and the child traversed to is displayed. The aspects of the presentation determined outside that choice node remain the same after this traversal. Thus, what is newly displayed after a hyperlink traversal is the endpoint of that traversal. What is removed from the presentation are the siblings of that endpoint if they are children of a choice composite. All other aspects of the presentation remain the same.

The choice construct is used in conjunction with the *composite* construct, described in Section 4.3. A composite can specify that one of a collection of document objects can appear at a screen location and duration of time of a presentation. The choice construct says what that collection of objects is for a composite. It is defined as a composite with HyTime location addressing constructs.

4.5 Measured Positioning

In CMIF, spatial and temporal positioning are determined by attributes of the media nodes. Temporal positioning is also determined by composite nodes, as described in Section 4.3.

There are many instances of spatial positioning in the Mondrian document, the most notable being the positioning of the 10 painting thumbnails in the larger window. The document's most notable instance of position along a timeline is the "concert" mode, where the presentation of a painting must be simultaneous with that of its composition, and both must start just after the presentation of the previous painting and composition end.

The HyTime *marker list (marklist)* and *extent list (extlist)* forms specify the measured position for a document object in an event schedule. The CMIF element type *position* specifies all the individual values for an object's scheduling. It contains the *screenextent* element, which states the four integers defining an object area on the screen. It can also contains *timebound* elements, which each specify either the starting time or duration of an object's presentation.

Synchronization is an important hypermedia concept handled by the CMIF *synchronization arc (syncarc)*. It is represented here with the HyTime *dimension reference (dimref)* form, which specifies the current coordinate in terms of another coordinate for another scheduled object. HyTime's dimref has been shown to be able to represent all of the synchronization relationships that occur in multimedia documents [5].

With a HyTime dimref, the referenced measurement must be an explicitly defined measurement or a dimension reference, allowing multiple levels of indirection. However, in HyTime, a dimension reference must eventually refer to either an explicitly stated value or a *marker function (markfun)*, which leaves the determination of the measurement to an unspecified external process. Marker function references refer to values that cannot be determined at authoring time. Thus, their exact specification falls outside the scope of HyTime and fits in the presentation specification layer.

5 The Berlage Document Architecture for Presentation Specification

HyTime is defined not as a DTD but as a *meta-DTD*. This allows multiple DTDs to be defined for documents that use HyTime. A DTD specifies explicit element types, but the HyTime meta-DTD specifies collections of attributes, called *architectural forms*, that can exist in the element types of a DTD. If an element type is said to conform to an architectural form, then it uses the attributes defined for that form. Such an element type can use other attributes as well that fall outside of HyTime. This enables new document sets to be created that use HyTime forms and incorporate their representational hypermedia semantics.

The HyTime standard document has formalized this meta-DTD approach with *SGML architectures*. An SGML architecture is defined by a meta-DTD, like HyTime's. Multiple DTDs can be made to define document sets for documents conforming to a given architecture. Architectures can also be layered on top of other architectures, causing all documents conforming to the first to also conform to the latter. HyTime is not built on top of any other architectures, but other architectures can be built on top of HyTime's.

HyTime contains no constructs for specifying structure that fall in the presentation specification layer of document modeling. In writing a HyTime DTD for CMIF, we encountered some presentation specification concepts that were broad enough to apply to many hypermedia document sets. We have defined an architecture to represent these presentation specification concepts. We call this the *Berlage architecture*, named after the Dutch architect of several important buildings in Amsterdam. The architectural forms for presentation specification can be used in DTDs for other HyTime document sets. These new forms are defined with HyTime form attributes, and thus elements conforming to these presentation specification forms also conform to their HyTime forms and will be recognized as such by HyTime engines, even if the engine does not recognize Berlage forms. The Berlage architecture extends HyTime, rather than overriding it. Berlage documents are also HyTime-conforming documents. Further, semantics deemed appropriate for HyTime representation are encoded with HyTime constructs in the Berlage architecture. What Berlage provides are some presentation-specification layer constructs that style sheets can refer to in addition to HyTime generic structure constructs in determining a presentation.

5.1 The Berlage Shadow Location Form

The CMIF *shadowset* construct specifies a single media object as a collection of alternative stored representations of that object. The presentation environment would select the most appropriate one to process for presenting the object, perhaps using criteria specified in a style sheet as described in Section 3.2.

In the Mondrian document, there is a low fidelity monophonic and high fidelity stereophonic file for each sound clip. There is also a low resolution and high resolution file for each image. The choice of which version of each media type to play is made by the user at the beginning of the document's presentation.

HyTime has no constructs for representing this concept because HyTime does not represent any hypermedia properties that aren't known at the time the document is being authored. This concept is represented instead by the Berlage form *shadow location (shadloc)*, which specifies a collection of located media objects of which only one will be presented.

5.2 The Berlage Aggregate Link Form

The CMIF choice composite defines how the display of certain document components ends when others are selected for display through the user's navigational interface, as described in Section 4.4. Its use is demonstrated by the Mondrian document's second window when the presentation is playing an "On Demand Recital".

The concept of document objects being displayed as a result of user navigation corresponds well with HyTime's notion of hyperlink anchor activation. However, HyTime does not specify an opposing component to this, which would specify which activated document objects would become deactivated. Such a specification involves too much of a document's presentation activities to be appropriate for HyTime representation.

Because this aspect of presentation specification has the potential to be used by a wide variety of document sets, we have defined a form to represent it called the *Berlage aggregate link (Bagglink)*. The Bagglink extends the HyTime *multiple location (multloc)*, by specifying that at most one of the group of objects located by the multloc is to be considered the located object at any point in the document's processing. This object is the one most recently traversed to through HyTime-defined hyperlinks during the document's processing. The Bagglink attribute *default active anchor (defactiv)* assigns one of the anchors as active if none have yet been activated. We use Bagglink in the HyTime/Berlage definition of the CMIF *choice* in our DTD, as shown in Figure 5.

5.3 The Berlage Duration Marker Form

Sometimes the ending time of a media object's presentation affects other aspects of the presentation timing. Since most CMIF media objects are stored externally to the CMIF document itself, they are subject to modifications that are more recent that the final modification time of the document, and thus the duration of these media objects during any presentation is not always certain at authoring time. The CMIFed environment, when processing a CMIF document, also processes these media object files to check their duration so that all timing information can be calculated for the document's presentation. Such processing is render-time. Synchronization arcs to the ending times of certain externally stored media objects are thus author-time generated constructs that refer to render-time characteristics. Also, render-time determined durations are needed to schedule the presentation of children of sequential composites when the duration of one or more of the children is determined by the implicit duration of external media files.

The duration of the presentation of each painting of the Mondrian document when in "concert" mode is determined by the time it takes to play each painting's composition. This in turn determines the starting time of the next painting.

Our Berlage architectural extension has a *duration marker (durmark)* form, which extends a HyTime marker function to specify that it returns a value representing the duration or length of a media object. A duration marker has attributes that specify the media object itself, what its format is, and what unit of measure applies to the value returned. This form is used with the duration element of the CMIF DTD, as shown in Figure 5. The duration element represents a synchronization arc that refers to the duration or length of a media object stored in an external file.

6 CONCLUSION

Our goal is to facilitate the generation of different target presentations from a single source document. We developed a framework to support the automatic processing of structured hypermedia documents into final form presentations. We first discussed a unified perspective of generic hypermedia structuring and associated presentation specifications as defined by the Dexter/AHM model and by the inter-related ISO standards SGML, DSSSL, and HyTime. The framework for CMIF document processing was presented in terms of this perspective. This framework builds upon text-based standards for the transformation of structured documents into final form presentations and extends it to include the processing of hypermedia documents. The processing of CMIF documents in such a framework requires their conversion into HyTime to obtain a presentation independent representation. While the presentation specifications contained in the original CMIF document can be recorded in style sheets, we identified a number of presentation specification concepts that are broad enough to apply to many hypermedia document sets. For these we developed the Berlage architectural forms which can be reused by other HyTime document sets and associated style sheets.

REFERENCES

1. Buford, J., Rutledge, L., Rutledge, J.L., and Keskin, C. "HyOctane: A HyTime Engine for an MMIS". *Multimedia Systems*, vol. 1, no. 4, February 1994.

2. Buford, J. "HyTime: Evaluation and Implementation Experience". *Proceedings of ACM Hypertext 96*, March 1996.

3. Candan, K.S., Prabhakran, B., and Subrahmanian, V.S. "A Framework for Supporting Multimedia Document Authoring and Presentation". *Proceedings of ACM Multimedia 96*, November 1996.

4. DeRose, S., and Durand, D. *Making Hypermedia Work: A User's Guide to HyTime*. Kluwer Press, Boston. 1994.

5. Erfle, R. "Specification of Temporal Constraints in Multimedia Documents Using HyTime". *Electronic Publishing*, vol. 6, no. 4, December 1993.

6. Goldfarb, C. *The SGML Handbook*. Oxford University Press. 1991.

7. Halasz, F., and Schwartz, M. "The Dexter Hypertext Reference Model". *Communications of the ACM*. Vol. 37, No. 2, February 1994.

8. Hardman, L., Bulterman, D.C.A., and van Rossum, G. "The Amsterdam Hypermedia Model: Adding Time and Context to the Dexter Model". *Communications of the ACM*. Vol. 37, No. 2, February 1994.

9. International Standards Organization. *Document Style Semantics and Specification Language (DSSSL)*. ISO/IEC IS 10179:1996. 1996.

10. International Standards Organization. *Hypermedia/Time-based Structuring Language (HyTime)*. ISO/IEC IS 10744. 1992.

11. International Standards Organization. *Standard Generalized Markup Language (SGML)*. ISO/IEC IS 8879. 1985.

12. Van Ossenbruggen, J., Hardman, L. Rutledge, L., and Eliëns, A. "Style Sheet Support for Hypermedia Documents". *Proceedings of ACM Hypertext 97*. April 1997.

13. Rutledge, L., van Ossenbruggen, J., Hardman, L., Bulterman, D., and Eliëns, A. "Generic Hypermedia Structure and Presentation Specifications". *Proceedings of ICCC/IFIP 97*. April 1997.

14. Rutledge, L., van Ossenbruggen, J., Hardman, L., and Bulterman, D. "Cooperative Use of MHEG and HyTime in Hypermedia Environments". *Proceedings of Hypertexts and Hypermedia: Products, Tools, Methods 97*. September 1997.

15. Weitzman, L., and Wittenburg, K. "Grammar-based Articulation for Multimedia Document Design", *Multimedia Systems*, vol. 4, 1996.