Lloyd Rutledge, Lynda Hardman, Jacco van Ossenbruggen and Dick C. A. Bulterman "Implementing Adaptability in the Standard Reference Model for Intelligent Multimedia Presentation Systems", Proceedings of Multimedia Modeling 98*, October, 1998.*

Implementing Adaptability in the Standard Reference Model for Intelligent Multimedia Presentation Systems

Lloyd Rutledge, Lynda Hardman, Jacco van Ossenbruggen* and Dick C.A. Bulterman
CWI (Centrum voor Wiskunde en Informatica), Amsterdam, The Netherlands
*Vrije Universiteit, Amsterdam, The Netherlands

Abstract

This paper discusses the implementation of adaptability in environments that are based on the Standard Reference Model for Intelligent Multimedia Presentation Systems. This adaptability is explored in the context of style sheets, which are represented in such formats as DSSSL. The use of existing public standards and tools for this implementation of style sheet-based adaptability is described. The Berlage environment is presented, which integrates these standards and tools into a complete storage-to-presentation hypermedia environment. The integration of the SRM into the Berlage environment is introduced in this work. This integration illustrates the issues involved in implementing adaptability in the model.

1 Introduction

Separation of structural and style information has long been commonplace for text, and can also be found in many hypertext models. The Dexter hypertext reference model separates the structural information of the storage layer from the style and layout information encoded in the presentation specifications [14]. The AHM (Amsterdam Hypermedia Model) extends Dexter into hypermedia by accounting for timing, more complex aspects of user interaction and how these should be conveyed to the user [12]. In most hypermedia design models, including RMM [19] and HDM [8], the two types of information are designed during different phases of the design process.

Intelligent Multimedia Presentation Systems (IMPS) deal with the dynamic creation of multimedia presentations optimally geared towards the needs of a user. In broad terms, the created presentation should define what is presented to the user (the content), where it is presented (the spatial layout) and when it is presented (temporal layout). Given a collection of user goals and availability of resources these three aspects leave open an immense number of possible presentations. An IMPS is a reasoning system aimed at selecting the optimal one.

The Standard Reference Model for Intelligent Multimedia Presentation Systems (SRM) [1] specifies the decomposition of this process into well defined layers. It can be used as a basis for discussing and comparing different systems that adapt to the user and his or her presentation environment. The SRM can also be used in guiding the development of such systems. The Berlage environment was developed as prototype for exercising issues of adaptability in hypermedia [22]. For this paper the Berlage environment was modified and extended to follow the design of the SRM and to incorporate some of its functions. The Berlage environment is built of a collection of related public formats and tools for hypermedia and document processing.

Because of the diversity of multimedia user communities and of the functions and processing approaches involved, no one format has been found that applies to all multimedia environments. As a result, many different formats for hypermedia exist, each with a different scope. Some focus on particular document sets or user communities and thus are quite detailed and not widely applicable. Others focus on different sets of shared functions between communities of users, thus providing common frameworks but not complete environments. Some focus on details of the final hypermedia presentation, while others define more general structure of documents that may be presented in a variety of different manners.

The ISO standard HyTime (Hypermedia/Time-based Structuring Language) specifies the representation of hypermedia documents in a presentation-independent format [15][7]. HyTime is defined as a subset of SGML (Standard Generalized Markup Language) [18][9], which defines the structure of electronic documents in general. A related language that is also defined as an SGML subset is XML (Extensible Markup Language), which provides a common framework for documents from different applications on the Web [2].

The ISO standard DSSSL (Document Style Semantics and Specification Language) encodes the transformation from SGML documents to formats that present them [17]. Thus, DSSSL systems can accept XML and HyTime as input. The use of DSSSL with HyTime was recently made easier with the release of the second edition of HyTime, which contains new facilities for use with DSSSL.

SMIL (Synchronized Multimedia Markup Language, pronounced “smile”) is a new W3C proposed recommendation for final-form hypermedia presentations distributed on the Web [14]. With W3C’s promotion and at least two publicly available SMIL players, it is expected that SMIL will be a widely-used means of distributing hypermedia documents. SMIL is defined as a subset of XML. Thus, DSSSL can encode transformations that output SMIL.

The issues discussed in this paper are illustrated with the Berlage hypermedia authoring and browsing environment [22]. This environment incorporates the use of HyTime to represent hypermedia documents in a format independent of their presentation. It also incorporates the use of DSSSL to specify the different mappings of these documents to their final presentations. Finally, Berlage generates and displays presentations encoded in SMIL. The Berlage environment consists of publicly available tools to demonstrate how such environments can be readily implemented on a wide scale.

These issues are also illustrated with an example application of the Berlage (named after H.P. Berlage, the architect of several buildings in Amsterdam) environment. This application is about the city of Amsterdam, The Netherlands, and is called Fiets (Foundation for Interactive Electronic Touring Systems, or *fiets* {pronounced “feets”}, the Dutch word for “bicycle” and generally the preferred means of personal transportation in Amsterdam) [22]. Fiets provides geographic and historic information about Amsterdam.

This paper first describes adaptability as implemented by style sheets in existing formats, tools and environments. Then the Berlage environment and its means of processing style sheets using these formats and tools is discussed. An overview of the SRM is provided. Finally, the use of the SRM in the architecture of the Berlage environment is presented. This use of the SRM illustrates the issues of adaptability that arise from its implementation.

2 Use of style sheets for adaptability in hypermedia

In this paper *adaptable hypermedia* is hypermedia that uses external intervention to be adapted for presentation. This contrasts with *adaptive hypermedia*, which adapts autonomously for presentation. Often adaptive hypermedia involves the author establishing what the varying circumstances of presentation are when the document is written and accounting for these possibilities at that time [3]. Adaptable hypermedia typically requires more processing than adaptive hypermedia, especially at presentation time. Adaptable hypermedia also requires maintaining a division between the document itself and how it is presented. However, adaptable hypermedia is usually more versatile than adaptive hypermedia, being able to account for presentation circumstances the author may not be able to predict. This paper concerns itself primarily with adaptable hypermedia. The distinction between an adaptable hypermedia document and its presentations is maintained with style sheets.

While the SRM defines a general processing model for dynamic presentation generation and adaptability, existing formats and tools based on style sheets currently provide an infrastructure for performing similar functions. The difference is that style sheet-based environments often do not have dynamic adaptation. Instead, static presentations are often generated that are tailored for circumstances that are known at generation time. This section discusses the DSSSL style sheet format and how it handles presentation generation and adaptability. Later this paper applies this discussion to implementing the SRM.

DSSSL is a Scheme-like language that describes how an SGML document is transformed into another SGML document or into a non-SGML format. Because HyTime documents are SGML documents, any HyTime document can be transformed by DSSSL. A DSSSL program is typically called a *style sheet*. The separation of style from structure and content enforced with the distinction between DSSSL and SGML/HyTime facilitates the creation of particular styles by the author that can be applied to documents of the same document set. Note that although the term style sheet is used, DSSSL can be used for more general, non-style transformations.

The design of typical DSSSL usage is shown in Figure 1. This diagram shows how an SGML document is processed with an accompanying style sheet by a DSSSL engine. The DSSSL engine processes the style sheet to generate the desired transformation of the source document into the presentation format. How this style sheet-based architecture was implemented in the Berlage environment is described later in this paper. Also described later in this paper are the extensions to this architecture, and their implementation in the Berlage environment, that are required to implement dynamic generation of presentations.

A hypermedia style sheet language needs to be able to define a mapping from the source hypermedia document model to the model of the target format. This can be the final-form model of the presentation (i.e. the model of the play-out environment) or another intermediate document model. In adaptive environments, style sheet conversion might also adapt to user characteristics (e.g. level of expertise) or to changing system resources (e.g. network bandwidth). A hypermedia style sheet could be used to indicate how to deal with limited resources (e.g. by specifying alternatives or quality of service negotiation protocols) on different platforms, thus making the document source independent of platform-specific details.

The DSSSL formatting process, for which publicly available tools exist [6], does such structural transformations. For documents with purely text content, DSSSL users can transform one SGML document type into another. The same process can also be applied to hypermedia document formats based on SGML. For instance, since SMIL documents are SGML-conforming documents, this extension can be used to generate SMIL documents from any other SGML-based hypermedia document format. One can use this to generate SMIL presentations from more abstract HyTime-encoded hypermedia documents. The Berlage environment uses this approach to translate SMIL to MHEG [24]. By employing reusable DSSSL libraries [16], the inner details of languages such as SMIL or MHEG can be hidden from individual style sheets. Such libraries would hide the inner details of the target format from the style sheet.

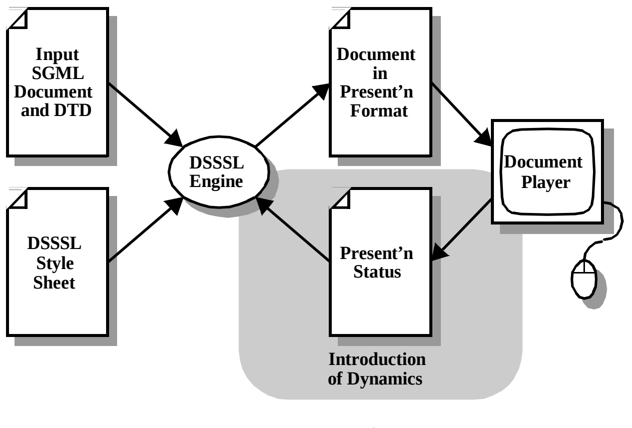

Figure 1: Typical Usage of DSSSL and the Introduction of Dynamics

3 The Berlage environment

This section describes how existing publicly available standards and tools are used in building the Berlage environment. First, the standards HyTime and SMIL are described, as is how they apply, respectively, to the storage and presentation of hypermedia. This is followed by a description of the Berlage environment and how it uses these standards and publicly available tools that process them to make a complete hypermedia authoring and presentation system. How the Berlage environment was extended to implement the SRM is described in the next section.

3.1 HyTime

HyTime is an ISO standard for representing presentation-independent hypermedia data. It is built upon SGML, which provides the basic structuring information that applies to document data in general. HyTime adds more complex structuring constructs and attaches hypermedia semantics to certain patterns of composites of this structure. The basic hypermedia semantics that HyTime represents include *hyperlinking*, which establishes descriptive relationships between document objects, and *scheduling*, which puts document objects in coordinate systems that can represent spatial and temporal structure.

HyTime and SGML are generally intended for encoding documents that are presentation-independent. They can apply to a wide variety of presentation situations but do not themselves represent particular presentations. HyTime and SGML documents typically must be processed into a different format appropriate for final presentation.

HyTime and SGML are meta-languages. They encode not only individual documents but also the document sets to which they belong. A document set is defined by an SGML *DTD (document type definition)*. An individual document conforms to a particular DTD. A DTD defines a specific syntax, in terms of SGML constructs, that its documents must follow. HyTime inherits from SGML the use of DTDs to define individual document sets.

With the document content and general structure being established by HyTime, the author of DSSSL style sheets can focus on the desired style of presentation, the mapping of the navigational interface, and the adapting of the presentation to particular environments and individual users.

3.2 SMIL

SMIL is a format representing hypermedia presentations on the Web. It incorporates basic hypermedia principles such as spatial layout, temporal composition, synchronization and navigational hyperlinking. It also has constructs that adapt presentations to the characteristics of individual environments and users. SMIL specifies the display of multiple media items in a coordinated fashion: displaying visual items and related locations on the screen, and synchronizing the timing of the presentation of these media items. SMIL specifies the display of a navigational interface that allows the user to select what portions of the presentation to currently display.

SMIL has an easy-to-author format whose syntax resembles HTML. Since SMIL documents are XML documents, they are SGML documents, and thus are readily processed as output, and as input, of DSSSL transformations.

3.3 The Berlage environment

Berlage is a hypermedia environment that uses publicly available standards and tools. The Berlage environment was introduced in previous work [22]. A diagram of the environment design is shown in Figure 2.

HyTime is used by Berlage to represent stored hypermedia data. SMIL is used to encode the hypermedia presentation of Berlage documents. DSSSL is used in the Berlage environment by authors to encode presentations of storage documents. A large document can be written once for long-term storage, and it can be presented in many different ways without re-editing the original document.

Standard text editors are used to create Berlage documents, DTDs and style sheets. A Berlage document and corresponding style sheet are input to a DSSSL engine and a SMIL document is generated. The DSSSL engine used in the Berlage environment is Jade, which is publicly available [6]. This SMIL document is then presentable across the Web with GR*i*NS or other SMIL players.

4 Implementing the SRM in the Berlage environment

The SRM components are illustrated in Figure 3. The SRM divides dynamic presentation generation into two areas: *generation process* and *knowledge server*. The generation process performs the run-time generation of the presentation based on the user's interaction, the history of the presentation and information provided by the knowledge server. The knowledge server stores and provides long-term instructions and information that apply to multiple presentations at any point in their run. As such, the generation process is the active component of the SRM, and the knowledge server provides the data that gets processed.

Before the incorporation of the SRM, the Berlage environment created only static presentations. With the SRM incorporation performed for this paper, the Berlage environment can adapt the presentation during the presentation itself. The dynamic generation of DSSSL code required by the SRM implementation necessitated additions and modifications to the Berlage environment architecture. These additions are shown in Figure 2. The primary new component added is the http server. The server enables the Berlage environment to keep track of the

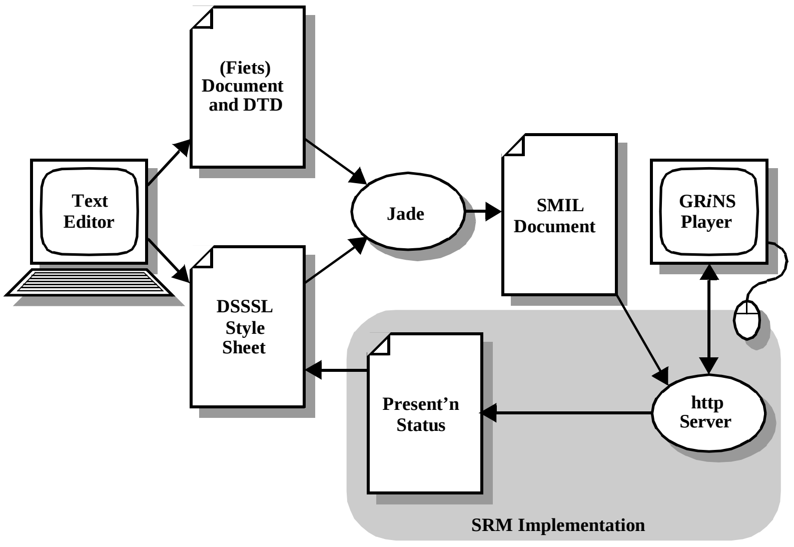

Figure 2: Berlage Environment Design with SRM Implementation

user's interaction with the presentation and with each interaction generate a new presentation state in the form of DSSSL code which can be processed with the style sheet. The environment then calls the Jade DSSSL engine to again process the document with the DSSSL style sheet, which includes by reference the newly generated presentation state code. This results in the creation of new SMIL code, which is then passed back to the SMIL player.

To enable Berlage to keep track of the user interaction, the destination of all navigational hyperlinks in the SMIL code is specified with URL code that the http server recognizes. These URL addresses are in the format used by HTML forms. Fields and values for those fields are set in the URL to convey to the server what the status of the presentation was and what choice the user selected from among those that were available. The server can then update its internal data store to properly represent the presentation's state. Next it generates the DSSSL presentation state code for inclusion in the next Jade processing of the document and its presentation. This generated DSSSL code includes data that enables the style sheet to define SMIL hyperlink destination URLs so that the server will recognize them when they are activated and respond appropriately.

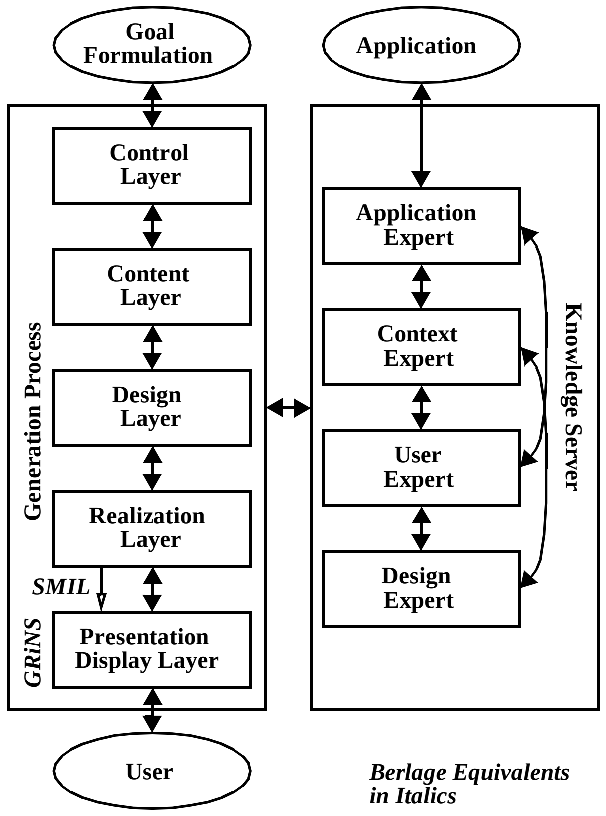

Figure 3: The SRM and its Components

The SRM generation process is divided up into layers. The *control layer* determines how the goals of the presentation should be met and makes sure that they are. In the *content layer*, a rough outline of the content and structure of the presentation is made. In the *design layer*, every part of the presentation is designed. In the *realization layer*, the design is realized into a full specification of the presentation in a platform independent way. It is the *presentation display layer* that processes the contributions of these other layers into the final presentation to the user.

A number of expert modules are defined which assists the generation process. The *application expert* provides information about the media used. It can, for example, say the dimensions of an image file to be used. It can also provide semantic information about media objects. The *context expert* keeps track of the presentation's history: what was shown, and how the user interacted. The *user expert* conveys information about a particular user that can be used in tailoring the presentation for him or her. Remaining information processing that affects how a presentation is adapted is put in the *design expert*.

In the Berlage environment, the functions of the generation process and knowledge server can be embodied at least in part by DSSSL code. The knowledge processing encoded in the Fiets application's DSSSL code is of a relatively simple nature, making basic decisions. DSSSL as a language has been adequate for this. More complex decision processes may require software outside of DSSSL to complement what is provided in the Berlage environment.

In the Berlage environment the control of the generation of the presentation is handled primarily by DSSSL processing. To achieve this goal, the amount of processing done by the http server must be minimized. As a result, the server is simply used to store old fields and values from previous URL requests, add new fields and values from incoming requests, and modify existing fields with new values from incoming requests. The presentation state simply conveys these fields and values, and the DSSSL style sheet has instructions on how to handle this information.

4.1 Control layer

The control layer determines at each point in a presentation what goals there are to meet and how best to meet them. It communicates frequently with the context expert, which keeps track of what has happened in the presentation so far. This layer influences the message to be expressed by a presentation, but does not make any decisions directly affecting the instantiation of parts of a particular presentation.

For example, Figure 4 shows one of the Fiets interfaces. Here, the user is shown a collection of Amsterdam buildings about which more detailed information can be accessed. A thumbnail image of the front of each building is shown, and each image is the starting point for a hyperlink leading to more information about that building. It is a goal of the application that the user visit each of the buildings. The user is helped in this task by the changing appearance of this portion of the Fiets presentation. After the user has visited a building and returns to this screen, the visited building, and all previously visited buildings, are shown as faded so that the user can distinguish them from the buildings that remain to be presented in detail.

When the user selects a building, Berlage generates presentation state DSSSL code that when included in the style sheet for processing causes SMIL code to be generated that displays detailed information on that building. In this and all future presentation state DSSSL code generated, code is introduced for that building indicating that it has been visited. The control layer style sheet code for Fiets has instructions that access this code in determining whether or not to fade the image displayed for each building in generating the SMIL code representing the next step in the presentation. The information conveyed by the presentation state code corresponds with the context expert of the SRM.

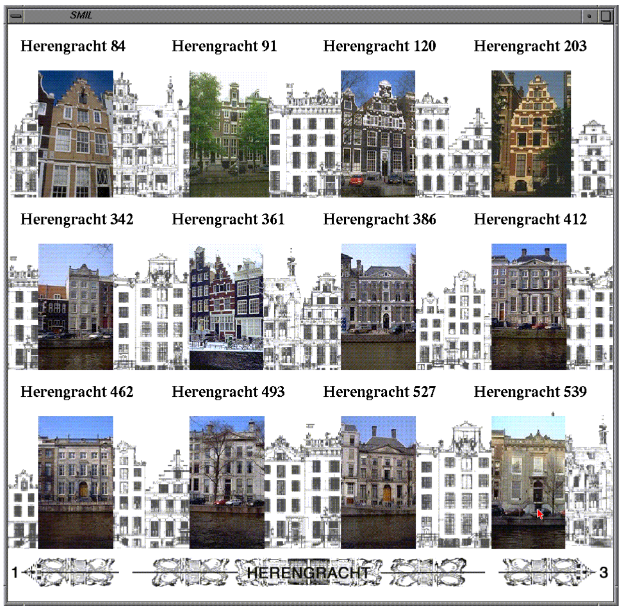

Figure 4: GR*i*NS Showing a Fiets Presentation

4.2 Content layer

In the SRM, the content layer takes a single goal passed on by the control layer and refines it to a list of subgoals. These are in turn transformed into a set of communicative acts and relations among these acts. This is carried out by four separate, but communicating processes—*goal refinement*, *content selection*, *media allocation* and *ordering*.

Goal refinement is the determination of the high level goal of the presentation. In the Berlage environment, this is considered the task of the author writing a style sheet. It is not considered here as an automatically processed task. Other systems, however, may implement knowledge processing that makes goal decisions at this level. Since it is not part of Berlage processing, we do not discuss goal refinement further here.

The *content selection* process communicates with the application expert and decides, given some semantic content, what media objects are appropriate for conveying it. In the Fiets presentation, when the user has selected a building the content of the resulting SMIL display must be determined. This display shows detailed information about a building, such as close-up images of different parts of it. This display also conveys more detailed information on how the building appears on the outside and on the inside. This associate of semantic descriptions with media contents in done in Berlage with HyTime. The processing of this HyTime code corresponds with the SRM's application expert.

When the content layer receives a request for detailed information on a particular building, it determines what the appropriate semantic content is for displaying this information. Precisely which buildings to display would be determined, along with what aspects of each building to display with it. What media items to use to represent these is determined elsewhere in the SRM, as is how to display them.

The *media allocation* component assigns a particular medium to a communicative act, producing a media communicative act. In the Berlage environment, this corresponds to the selection of particular media objects to display for given semantic content. In the Fiets example, the semantic content of detailed information on a building's exterior, determined during content selection, could correspond with particular images of its gable ornamentation and entranceway. This correspondence is determined during media allocation.

The instructional code for determining this media allocation is contained in the content layer portion of the style sheet. The content layer portion of style sheet the instructions for determining the media allocation for displaying given content. In Fiets, such instructions often refer to the HyTime-defined structure of the stored document to determining this media allocation. For example, HyTime structures in Fiets associate images and descriptive audio and text of gable ornamentation and entranceways with individual buildings. This HyTime code would be accessed by Jade as instructed by the content layer portion of the DSSSL style sheet to determine the media files to use for a particular building.

The result of the *ordering* process is a not necessarily linear ordering of the media communicative acts. Navigational hyperlinks, timing relations and spatial layout are all different ways of expressing ordering relations among media items in the final SMIL presentation code. When a communicative act is to be expressed using a number of media items, these methods can be traded-off against one another. For example, rather than display many items together, links can be made among smaller groups of the items. Also, the sequential playing of items may be a suitable alternative for laying out items next to each other at the same time. The choice among these alternatives in the Fiets application is described in other work [21]. These different ways of “ordering” the material need to be captured within the content layer, but may not be finalized until the design or realization layer [13]. We restrict the content layer to specifying a structure of links, and leave the decisions on temporal and spatial layout to other layers. In the Berlage environment, specifying this link structure for a given state in the presentation means determining what to show on the display being generated and what information should instead be made accessible by links from this generated display.

4.3 Design Layer

The design layer is split into two communicating processes: *media design* and *layout design*. The processes of media design and layout design carry on in parallel, imposing no ordering requirements for which of these should be finalized first.

The *media design* component makes decisions on the “look and feel” of the presentation. It can communicate with the design expert and make decisions on styles. Style information can include color, font, background wallpaper images, and other aspects of the presentation that do not affect the actual media content or its positioning in space and time in the presentation.

The *layout design* component is responsible for the spatial and temporal arrangement of the media items in the presentation. In this layer the constraints used for determining the spatial and temporal layout need to be generated. These specifications should be internally consistent before being handed on to the realization layer, which may, nonetheless, be unable to satisfy them requiring the generation of a new set of constraints. The generation of the spatial layout design could occur along those lines described in the Reference Model for Intelligent Multimedia Layout [10].

In the Fiets example, one screen display has thumbnail images for multiple buildings. The arrangement of thumbnail images on a single screen is a design layer spatial layout decision. The design layer arranges the display of the images as from left to right along rows going from the top to the bottom of the screen. The design layer states that the images that overflow off the right of one row wrap around to the left side of the row below it. Another decision made by the design layer is that if there are too many buildings to select from, the thumbnails are distributed among multiple screen displays. Access to each of these displays is provided with hyperlinks. How these guidelines map to the exact placement of these images is determined not here but by the realization layer

4.4 Realization Layer

In this layer the final decision is taken on the media items to be played in the presentation and their corresponding spatial and temporal layout. The output of this layer is the final SMIL code that gets presented to the user through the presentation display layer, as shown in Figure 3.

Again, the layer is split into two communicating processes, paralleling those in the design layer—media realization and layout realization. The media realization component ensures that appropriate media items are chosen. These may already exist, or may be generated specifically for the task.

The layout realization component calculates the final temporal and spatial layout of the presentation based on the constraints specified in the design layer. The duration in time of an atomic media object’s presentation may be explicitly encoded in the storage structure or derived from the content for continuous media. The duration of a portion of the presentation consisting of multiple media objects can be calculated on the basis of these objects along with any other temporal constraints specified in the design layer. Where links define possible paths among separate multimedia presentations, there is also a duration perceived by the user when the link is followed.

In the Fiets example, the exact position of each building thumbnail and the number of thumbnails on each screen display is determined. This positioning gives the building images an even distribution in the screen space. The exact font size and positioning of the associated text and hyperlink hotspots are also determined here. The design layer determines the manner in which thumbnails are arranged, while the realization layer determines the details of this arrangement given the circumstances of a given presentation.

4.5 Presentation Display Layer

As shown in Figure 3, the presentation display layer is embodied in the Berlage Environment by GR*i*NS. which processes the generated SMIL code for final presentation to the user.

5 Summary

This paper explored the implementation of adaptability in environments that are based on the Standard Reference Model for hypermedia. Adaptability was discussed in terms of style sheets, particularly as represented by DSSSL. The Berlage hypermedia environment and its use of style sheets and existing public standards, such as HyTime, DSSSL and SMIL, and tools such as Jade and GR*i*NS, was described. The integration of the SRM into the Berlage environment was introduced in this work. This integration was used to illustrate the issues presented in this paper regarding the processing of adaptability and its relationship with the Standard Reference Model.

6 Acknowledgments

The CMIFed and GR*i*NS environment was implemented by Sjoerd Mullender, Jack Jansen and Guido van Rossum. The artwork and graphic design of the Fiets application was done by Maja Kuzmanovic. The development of CMIFed and GR*i*NS was funded in part by the European Union ESPRIT Chameleon project.

References

1. Bordegoni, M., Faconti, G., Feiner S., Maybury, M.T., Rist, T., Ruggieri, S., Trahanias, P. and Wilson, M. "A Standard Reference Model for Intelligent Multimedia Presentation Systems", *Computer Standards and Interfaces* 18(6,7) December 1997, North Holland, pp. 477-496.
2. Bray, T., Paoli, J. and Sperberg-McQueen, C.M., *Extensible Markup Language (XML)*. W3C Recommendation, January 1998. http://www.w3.org/TR/REC-xml.html.
3. Bulterman, D.C.A. "User-Centered Abstractions for Adaptive Hypermedia Presentations", in Proc. *ACM Multimedia 98*, September 1998.
4. Bulterman, D.C.A. "Models, Media and Motion: Using the Web to Support Multimedia Documents", in Proc. *Multimedia Modeling 97*, November 1997.
5. Bulterman, D.C.A., Hardman, L., Jansen, J. Mullender, K.S. and Rutledge, L. "GRiNS: A GRaphical INterface for Creating and Playing SMIL Documents", in Proc. *Seventh International World Wide Web Conference (WWW7)*, April 1998.
6. Clark, J. Jade — James' DSSSL Engine. http://www.jclark.com/jade/.
7. DeRose, S. and Durand, D. *Making Hypermedia Work: A User's Guide to HyTime*. Kluwer Press, Boston. 1994.
8. Franca Garzotto, Luca Mainetti, and Paolo Paolini. "Hypermedia Design, Analysis, and Evaluation Issues", *Communications of the ACM*, 38(8):74 86, August 1995.
9. Goldfarb, C. *The SGML Handbook*. Oxford University Press. 1991.
10. Graf, W., "Intelligent Multimedia Layout: A Reference Architecture", *Computer Standards and Interfaces*, 18 (1997).
11. Halasz, F., and Schwarz, M., "The Dexter Hypertext Reference Model", *Communications of the ACM*, vol. 37, no. 2, February 1994.
12. Hardman, L., Bulterman, D.C.A. and Rossum, G. van, "The Amsterdam Hypermedia Model: Applying Time and Context to the Dexter Model", *Communications of the ACM*, vol. 37, no. 2, February 1994.
13. Hardman, L., Worring, M., and Bulterman, D.C.A., "Integrating the Amsterdam Hypermedia Model with the Standard Reference Model for Intelligent Multimedia Systems", *Computer Standards and Interfaces*, 18 (1997).
14. Hoschka, P. (ed.). *Synchronized Multimedia Integration Language*. W3C Recommendation, June 1998. http://www.w3.org/TR/REC-smil/.
15. International Standards Organization. *Hypermedia/Time-based Structuring Language (HyTime)*. Second Edition. ISO/IEC IS 10744:1997, 1997.
16. International Standards Organization. *Coding of multimedia and hypermedia information — Part 5: Support for base-level interactive applications (MHEG-5)*. ISO/IEC IS 13522-5:1997, 1997.
17. International Standards Organization. *Document Style Semantics and Specification Language (DSSSL)*. ISO/IEC IS 10179:1996, 1996.
18. International Standards Organization. *Standard Generalized Markup Language (SGML)*. ISO/IEC IS 8879:1985, 1985.
19. Tomás Isakowitz, Edward A. Stohr, and P. Balasubramanian. "RMM: A Methodology for Structured Hypermedia Design". *Communications of the ACM*, 38(8):34 44, August 1995.
20. Rutledge, L., Buford, J.F. and Rutledge, J.L. "Modeling Techniques for HyTime", in Proc. *Multimedia Modeling 95*, November 1995.
21. Rutledge, L., Hardman, L., van Ossenbruggen, J. and Bulterman, D. "Structural Distinctions Between Hypermedia Storage and Presentation", in Proc. *ACM Multimedia 98*, September 1998.
22. Rutledge, L., van Ossenbruggen, J., Hardman, L. and Bulterman, D. "Practical Application of Existing Hypermedia Standards and Tools", in Proc. *Digital Libraries 98*, June 1998.

23. Rutledge, L., van Ossenbruggen, J., Hardman, L. and Bulterman, D. “A Framework for Generating Adaptable Hypermedia Documents”, in Proc. *ACM Multimedia 97*, November 1997.

24. W.R.T ten Kate, P.J. Deunhouwer, D.C.A. Bulterman, L. Hardman, and L. Rutledge. “Presenting Multimedia on the Web and in TV broadcast”. In *3rd European Conference on Multimedia Applications, Services and Techniques*, May 1998.

25. Van Rossum, G., Jansen, J., Mullender, K.S. and Bulterman, D.C.A. “CMIFed: A Presentation Environment for Portable Hypermedia Documents”, in Proc. *ACM Multimedia 93*, August 1993.

26. Van Ossenbruggen, J., Hardman, L., Rutledge, L. and Eliëns, A. “Style Sheet Support for Hypermedia Documents”. in proc. *Hypertext 97*, pages 216 217, April 1997.

27. Weitzman, L. and Wittenburg, K., “Grammar-Based Articulation for Multimedia Document Design”, *Multimedia Systems*, 4: 99-111, 1996