Structured Multimedia Authoring

Lynda Hardman, Guido van Rossum, Dick C.A. Bulterman

CWI: Centrum voor Wiskunde en Informatica
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands
Phone: +31-20-592 4127 Fax: 4199

Email: Lynda.Hardman@cwi.nl

ABSTRACT

We present the user interface to the CMIF authoring environment for constructing and playing multimedia presentations. The CMIF authoring environment supports a rich hypermedia document model allowing structure-based composition of multimedia presentations and the specification of synchronization constraints between constituent media items. An author constructs a multimedia presentation in terms of its structure and additional synchronization constraints, from which the CMIF player derives the precise timing information for the presentation.

We discuss the advantages of a structured approach to authoring multimedia, and describe the facilities in the CMIF authoring environment for supporting this approach. The authoring environment presents three main views of a multimedia presentation: the *hierarchy view* is used for manipulating and viewing a presentation's hierarchical structure; the *channel view* is used for managing logical resources and specifying and viewing precise timing constraints; and the *player* for playing the presentation.

We present the authoring environment in terms of a short example: constructing a walking tour of Amsterdam.

Keywords: Multimedia authoring, Hypermedia authoring, Synchronization, User interface.

1. INTRODUCTION

The promise of multimedia computing is that it will redefine the way that people communicate with each other via computers. With the first wave of multimedia authoring environments now commercially available, the potential and power of adding sound and video to their documents is becoming apparent to users. At the same time, it is also becoming clear that the creation of even trivial multimedia presentations is a complex and time-consuming task. As a result, it is likely that the technology of multimedia will race ahead of the use of multimedia computing until better tools exist for users to create, and change, multimedia presentations.

We feel that a key to reducing this authoring burden is the use of structure-based authoring tools. These allow the explicit manipulation of the structure of a multimedia presentation rather than the implicit manipulation of this structure via, for example, a time-based view. Structure-based authoring of a multimedia presentation requires a well-developed model of a multimedia document.

ACM Multimedia 93 /6/93/CA, USA

Our work uses CMIF (CWI Multimedia Interchange Format) [2], a system-independent representation of a multimedia presentation. This represents two main aspects of a multimedia presentation: the structure of the presentation, and the assignment of logical resources. The explicit representation of the structure allows the support of modular authoring of the presentation. The structure is also used for deriving timing constraints in the presentation, thus relieving the author from having to explicitly state these constraints. The specification of logical, rather than physical, resources enables a presentation to be defined in a system-independent way.

The CMIF authoring environment was designed to support the composition of CMIF-based multimedia presentations from existing media items. This process can be divided into two separate stages—the construction of the structure of a presentation, from which basic timing information is derived, and the assignment of detailed timing constraints between constituent data items. These two tasks are supported in two views of the CMIF document—the hierarchy view, showing the structure, and the channel view, showing logical resource usage. A CMIF document player, that maps the logical document to a particular presentation environment, is also integrated into the authoring environment allowing the author to view the presentation as the reader will see it.

Editing of the media items included in a presentation, such as text fragments or sound clips, is carried out using existing editors appropriate to that medium. These editors are directly accessible from the CMIF authoring environment. A description of the architecture and implementation of the CMIF authoring environment is given in a companion paper [8].

The CMIF authoring environment also allows hyper-links to be created between items within a multimedia presentation. The emphasis of this paper, however, is on using a structure-based approach for authoring presentations composed from dynamic media, whether multimedia or hypermedia. Issues involving a data model combining hyper-links and dynamic media are more fully addressed in [4].

The following section discusses the advantages of using a structured approach to multimedia authoring and compares our work with related work. A small example, used throughout the rest of the paper, is introduced in section 3, after which the views of a CMIF document for supporting structure-based multimedia authoring are described in detail. For completeness, a small section discusses the hypermedia functionality of the system. Experiences learned from using and developing the CMIF editing environment are reported, followed by conclusions.

2. MULTIMEDIA AUTHORING: APPROACHES AND PROBLEMS

Authoring approaches to multimedia have developed from different areas of expertise, such as animation and interactive video. While existing approaches offer some insights into how we can handle multimedia, we need to step back and consider future requirements for authoring large, complex multimedia presentations. We first discuss the limits of two common approaches then go on to argue the necessity of a structure-based approach.

2.1. Time-lines and scripts

Authoring systems such as MacroMind Director [5] allow the creation of multimedia presentations using a timeline based approach. The authoring model presented is that of a line along which time flows, upon which objects, with accompanying location information, are placed. Another common approach to authoring multimedia is "script" or "flowchart" based, where the author explicitly programs timing and location information. Most of the commercially available multimedia editors listed in [9] are based on these two approaches.

Although both script and timeline based editing use intuitive paradigms, neither approach capitalizes on the inherent modularity in the structure of the presentation. This leads to fragmentation of the authoring process and unnecessary work when copying or updating parts of the presentation. For example, when an author wishes to re-use a part of the presentation (in order to repeat a similar sequence) it is unclear where copying should begin and end—statements at the beginning of a script may have an influence far into the presentation; in a timeline representation the end of one scene and the beginning of the next is not explicitly represented. Further, if a complex sequence has been completed and the author wishes to change some part of it, then all the previously defined relations have to be checked to see whether they are still valid, and then changed as necessary. This may involve reassigning the positions of all the objects on the timeline or rewriting a large portion of a script or flowchart.

Approaches have been developed which move away from the script and timeline approaches. For example, the Harmony system [3] supports the definition of timing relations directly between media objects, rather than defining them in relation to a time axis. This makes inserting or deleting objects in the presentation much easier. Another example is the Videobook system [7] which allows composition of multimedia objects into scene objects which can be grouped into scenes. The author can manipulate groups of objects directly, without having to work out where in a timeline or script the beginning and end of the grouping is. While both these models are useful they do not in themselves provide a sufficiently rich model for conveniently handling all aspects of multimedia authoring. In the following section we investigate the requirements for authoring support and discuss the implications of these.

2.2. Structured authoring—a better approach

The need for structure-based authoring

Creating a multimedia presentation requires the specification of which objects are played when and where on the screen, or audio channels. While this information needs to be specified somehow, an author tends to think in terms of the message being communicated, the ordering on the material, and the "flow" of the presentation—an author does not *want* to specify what happens every single second in the presentation. We want to provide an authoring system which supports the author as much as possible in thinking in high-level terms, and which automatically generates the corresponding low-level timing and placement information.

Our thesis is that most authors use structure when authoring, even in single media, but that this structure is normally implicit. Through providing computer support this structure can not only be made explicit, but can be manipulated. We conclude that we should make the implicit structure of multimedia presentations explicit and support its manipulation. We argue our case by considering the construction of a video news item, and editing paper documents using a word processor.

In a dynamic medium (such as video) any structure used during the authoring process is lost in the final representation. Consider a television news item composed from a number of video clips and accompanying soundtrack. During the construction the editor assembles the video clips in a particular order, perhaps cutting them as required. A soundtrack is chosen to accompany the video clips, synchronized at various points with the video. The news item has been created from a dozen or so different media sources, but once complete, on videotape, the news item is presented as a linear medium. The editor looking through the news item will recognize the different source materials, but no explicit record of the sources or the structure remains in the final video medium. This makes re-use, or improvement, of parts of the news item, for anything more than duplicating or cutting a time-slice, at best extremely inconvenient.

In the world of paper documentation we often use word processors which support the WYSIWYG paradigm. However, we should not be misled by this paradigm, since while authoring it is not strictly adhered to: the underlying structure of the document can be viewed and manipulated by the author while editing, although the printed output has no explicit reference to the structure. In the on-line editing environment the underlying structure is represented in many word processors by symbols which are displayed on the screen but are not printed out.

When creating a presentation from multiple dynamic media in an on-line environment, which is far more complex than editing either video or paper documents, we need some way of reducing the complexity of the author's task. Making the implicit structure explicit, and supporting editing via this structure, is an essential requirement in a multimedia authoring environment.

Implications of structure-based authoring

Having established the requirement for explicit structure based editing, we need to support the author in remaining aware of the "big picture" of the presentation while still being able to edit the finest details. Mills et al. [6] have constructed a prototype which illustrates this approach for video editing. The author can zoom in on parts of the video while still being aware of where that part belongs in the complete video.

As well as being able to navigate around the presentation's structure, an author should be given support for creating the structure "top down" or "bottom up". A multimedia presentation can be thought of in terms of, for example, scenes, collections of scenes, and clips composing a scene. An authoring environment should provide a means for creating an empty structure which can later be filled in: for example, a sequence structure is created and then filled in with individual data nodes comprising dynamic (video, audio, animation) or static data (text, graphics), or from collections of nodes. As well as this top down support, the author may wish to work bottom up, creating small clips and combining them into scenes. An authoring system needs to support both top down and bottom up approaches to allow both experimentation and easy construction by the author.

With a structure-based approach we are able to derive most of the timing relations required between media objects from the structure. In the cases where an author wishes to specify other timing constraints, these should be defined as such, and not via the intermediary of a timeline. Buchanan and Zellweger [1] have created a system which allows the author to define constraints between media items which are then used to calculate the exact timings for the presentation.

By defining a presentation structurally, we can include multiple parallel structures which allow, for example, different language sets to be defined. Extra languages can be added into the structure simply, and end readers can select which they wish to use.

In summary, an authoring environment for multimedia needs to provide support for viewing and manipulating presentations via their structure, supporting both top down and bottom up construction. Constraints between media items should be defined directly between those items and not via a timeline or separate script.

3. EXAMPLE PRESENTATION

In order to discuss the issues associated with authoring multimedia presentations we present an example of a "typical" multimedia document. Figure 1 shows a clip from a multimedia tour of Amsterdam. The walking route from this small tour of Amsterdam will be used as the example throughout the rest of the paper. The walking route contains three stops of interest, a view onto one of the old canals, a close up of some typical gables (shown in figure 1) and a picture of a group of street musicians.

The gables clip itself (figure 1) is formed from a collection of data nodes displayed on the screen, in this case an image and some text nodes. A spoken commentary is also given and the subtitles change in time with the spoken words. The text nodes at the bottom of the screen are linked to other parts of the multimedia presentation. There are two sets of subtitles (which can be seen in the figure): one in English and one in Dutch. There are also two separate languages which have been recorded as commentary. The reader can choose which of these languages to listen to or read.

In the following section we will describe how this presentation can be authored.

Figure 1. The gables clip from a walking route in Amsterdam.

4. THE CMIF AUTHORING ENVIRONMENT

The CMIF authoring environment has been designed to give authors a rich environment in which the multimedia presentation being composed can be viewed and manipulated in a number of ways, providing an overview of the presentation's structure and use of resources (such as screen area and audio output) while still allowing the manipulation of layout and timing details.

The information dealt with by the current CMIF authoring environment can be split into four basic media types—text, still images, audio and video (frame-based digital images). The media objects are created in their own editor(s)—available directly from within the CMIF authoring environment. The environment can be extended to support other media types, for example combined video and audio.

Rather than overload the author by presenting all the required information at once, the CMIF authoring environment separates out different types of information and presents them in three separate, but closely communicating, views. The separation of the three views is derived from the different tasks the author carries out.

The tasks the author is faced with when constructing a presentation are: to structure the available media items; to assign the media items to appropriate logical output devices (thus defining the document in system-independent terms); to add more detailed timing constraints; and to be satisfied that the presentation as the reader will see it is of sufficiently high quality.

The CMIF authoring environment has been designed to support the author with the current task, rather than requiring the author to manipulate all aspects of the presentation in one uniform interface. The three views provided give different information about the same underlying CMIF document, and each view allows the author to manipulate different aspects of the presentation.

For supporting the author with editing the presentation's structure we supply the *hierarchy view.* This view allows the author to define the structural relations among the media items making up the presentation.

Rather than forcing the author to define timing details for the complete presentation (as is required in the authoring systems mentioned in section 2.1) we use the already-specified structural information in order to derive basic timing information for the presentation. This, as part of the logical resource usage, is displayed in the *channel view.* The structure of the presentation indicates parallel and sequential timing relationships, and if the author wishes to add further synchronization constraints between any two data nodes this can be carried out in the channel view.

Finally, the author needs to be able to see the presentation as the readers will see it. The *player*, used to control the playing of a presentation, is closely integrated with the two other views, allowing the author to view the presentation while also having direct access to the same media items in the other two views. For example, the author can indicate an item as it is being played and have it highlighted in the other two views.

The hierarchy view, channel view and the player are described in detail in the following sections.

4.1. The Hierarchy View for Structure Manipulation

The hierarchy view is the primary authoring window, where the author can create multimedia presentations using a top-down or bottom-up approach. We first describe the interface to the hierarchy view, then we go through an authoring example to demonstrate how this view can be used for structure-based editing.

4.1.1 Description of the Hierarchy View

As argued in section 2.2, it is essential that the author is able to manipulate the structure of a presentation in an explicit manner, and that presentations can be built top down and bottom up. The purpose of the hierarchy view is to provide a means of displaying and manipulating the structure of a multimedia presentation.

The multimedia presentation has a hierarchical structure whose leaf nodes are the data nodes which are played in the presentation, and whose non-leaf nodes are composite nodes containing a collection of other composite nodes and/or data nodes. The hierarchial structure gives a choice of playing the child nodes (be they composite or data nodes) in parallel or sequentially. This then defines coarse timing information for the presentation. Each data node is assigned to a channel, a logical output device which is mapped by the player at runtime to a physical output device—an area on the screen or a loudspeaker channel (section 4.2 describes the channel view in detail). The hierarchical structure of the presentation is represented in the hierarchy view as an embedded block structure.

The author composes a presentation by defining the structure of the presentation, and assigning the appropriate data nodes to the structure. The author does not need to specify any timing information at this point, since this is deduced from the hierarchical structure and the durations of the nodes within the structure. (Timing constraints *can* be added later—see section 4.2).

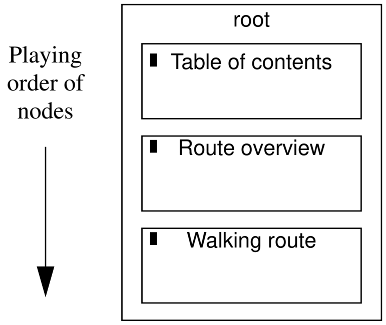


Figure 2. Top level structure of the Amsterdam tour. A small dark box indicates that the node containing it has embedded structure, not currently being shown.

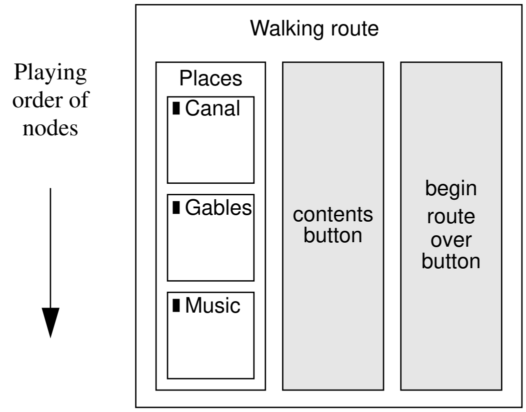


Figure 3. Structure of the *Walking route* sequence. The *Places* node contains three children each with nested structure. The shaded middle and right-hand boxes represent text data nodes (leaf nodes of the hierarchical structure).

The box labelled *Gables* plus the two text nodes make up the clip shown in figure 1.

The structure of a small tour of Amsterdam is shown in figure 2. The *Walking route* section contains the gables clip shown in figure 1.

The author can navigate around the hierarchical structure and zoom in on nested structures. This not only reduces the screen space required for representing the structure, but gives a focussed view of the part of the structure the author is currently interested in. For example, if we zoom in on the *Walking route* in figure 2, we see the structure shown in figure 3.

The two text nodes in figure 3 remain on the screen for the duration of the *Walking route* sequence. This use of the hierarchical structure enables screen layouts to be designed where persistent objects, such as titles and logos, need be placed only once and are retained on the screen throughout the scene (in fact, for the duration of the structure in which they are defined). This has the result that authoring work is reduced through the use of the hierarchy—items that remain for a greater part of the presentation can be defined in the higher levels of the structure.

A composite node can be of two types—parallel or sequential. In figure 3 the three boxes *Places*, *contents button* and *begin route over button* are played in parallel; the three smaller boxes nested inside the *Places* box are played one after the other. The duration of a composite node is derived from its children: the duration of a serial composite node is the sum of the durations of its children; that of a parallel composite node is the duration of the longest child. When a node has no explicit duration, for example a textual title, it is presented for the duration of its parent.

Within the hierarchy view the author can select any object (data node or composite node) and cut, copy or paste it, or interrogate it for more information, such as the object's attributes. Examples of attributes are: node name, data file referred to, channel used, explicit duration, comment, highlight colour.

4.1.2 Authoring in the Hierarchy View

A multimedia presentation can be authored by first creating the structure and then assigning data nodes at the leaves of the structure. The data node needs to be assigned to a channel (a logical output device) and to have a media object associated with it, usually by a reference to a file containing the data. When associating a file with a data node, the author does not need to be aware of the details of the data format of the element being inserted, since the player will convert it (if it is one of the recognized formats) at runtime.

In order to give a feel for creating a multimedia presentation using the hierarchy view, we return to our walking tour through Amsterdam, of which the gables clip shown in figure 1, is a part. The structure we will create is shown in figure 3. The stages in the creation process are shown in figure 4. (While this presentation could be created either in a top-down or bottom-up manner, we will describe the top-down approach here.)

We start by discussing the steps needed to create the structure shown in figure 4 (a). (Note that while this first description is very detailed we will combine a number of steps in subsequent paragraphs in order to build a realistic document.) We begin by taking the (empty) root node and inserting a child, which we call *Walking route*. Since we know that the node will contain several parallel components, we define it to be a parallel node. We "fill" the new node with three parallel children, calling the first one *Places*, the second one *contents button* and the third one *begin route over button*. As part of the creation process, we associate the *contents button* with the text file "contents.txt", a file that already exists in our file system. We also assign it to the *Content*s *button* channel, which has been pre-defined by the user to display text information at the bottom left of the presentation's main window. The *contents button* will be linked to the *Table of contents* composite node (shown in figure 2). The *begin route over button* is created in a similar manner to the *contents button* and will be linked to the start of the

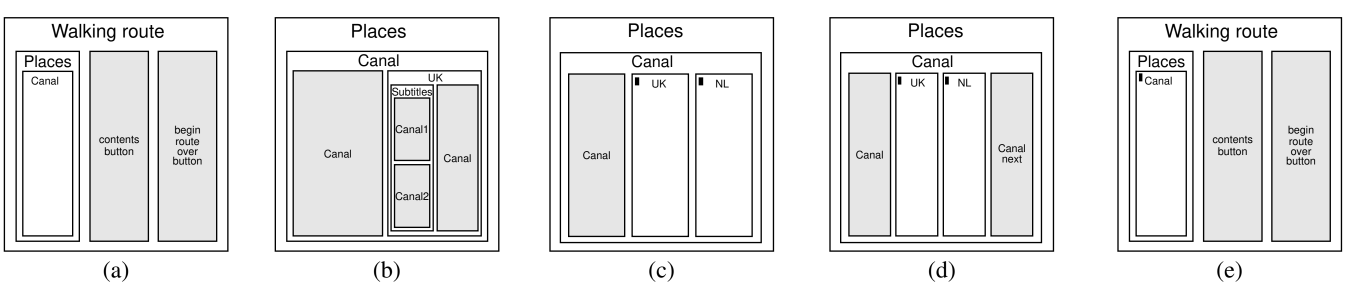


Figure 4. Authoring in the hierarchy view—stages in the creation of figure 3.

walking route—in fact the *Walking route* composite node. Next we define the *Places* node.

The *Walking route* node has been defined as a parallel node, indicating its three children should be played at the same time. We reselect the *Places* node and indicate that it should be interpreted as a serial node, meaning its children will be played one after the other. We then create a child of this node and name it *Canal*, making it a parallel node (it will contain the rest of the data nodes which are presented in the clip).

We now move to figure 4 (b) where we define the internal structure of the *Canal* node. We zoom in on the *Canal* node and create a child node, also naming it *Canal*. (Note that the author can choose to reuse node names since they are not used by the system.) We assign the large picture channel and select a picture file. For the soundtracks and subtitles it gets a little more complicated, since we have two spoken languages and two sets of subtitles which are synchronized with the speech. (We will tackle the synchronization in the channel view; for now we need only create the structure.) First we create a parallel node *UK*, then create two children: *Subtitles* and *Canal*. *Subtitles* is a serial node with two children *Canal1* and *Canal2* both placeed in the “Subtitles UK” channel and assigned text files; *Canal* is placed in the “Speech UK” channel and assigned a sound file. The structure now corresponds with that shown in figure 4 (b).

Having created the complex substructure for the UK text and speech we copy the node *UK* then paste the copy after the *UK* node. We then change its name to *NL*. We now go through the already-created structure and reassign the files to the corresponding Dutch files. (Note that we do not need to change the names of the nodes, although we could if we chose to do so.) The structure is now as shown in figure 4 (c), where the underlying structure of the *UK* and *NL* nodes is identical.

The final part of the Canal clip is the *Canal next* button. This is created after the *NL* node, assigned to the “Next button” channel, and associated with a text file. This structure is shown in figure 4 (d). The zoomed out view is shown in figure 4 (e).

We can now copy the completed *Canal* structure, paste the copy after itself and rename it to *Gables*. Instead of having to repeat the structure creation process we have only to navigate through the existing structure, change a few names as appropriate and assign the different data files (all the channels remain the same). This is repeated to create the *Music* clip and we arrive at the structure shown in figure 3.

4.2. The Channel View for Logical Resource Allocation

While the hierarchy view provides a means of organising the structure of a presentation, it provides only an indirect way of specifying logical resource use. To provide the author with control of the available resources, the channel and synchronization view, or *channel view*, provides a view of the media objects mapped onto the available logical resources (called *channels*). By supplying an extra layer above the physical resources the author is able to describe the presentation in a system-independent way. It is up to the player, optimized for a particular hardware configuration, to interpret the logical channels and assign the media objects to the available physical output devices.

4.2.1 Description of Channel View

The channel view shows the timing relations derived from the structure defined in the hierarchy view. The data nodes making up the presentation (the leaves of the hierarchical structure) are shown with their precise durations and timing relationships. If the author changes the timing in any part of the presentation, via either the hierarchy or channel views, the channel view is immediately updated to reflect this.

The data nodes are each assigned to a channel. The channels enable the author to define high-level presentation characteristics for each media type, so that presentations can be composed without having to specify details for each individual node: for example, a sound channel defines a volume; a text channel defines a rectangular area on the screen and a font. Attribute values can be overridden by an individual data node, for example, a short text string can be displayed in a larger font.

As well as providing a device-independent description of a data node’s display characteristics, channels allow the author to include, for example, multiple languages (spoken or written) within one presentation rather than having to recreate the complete presentation for each language. The player allows the reader to dynamically select which language to listen to, by selectively turning channels on or off.

Figure 5 shows the channel view for the Walking route scene in figure 1. The structure of the same scene is shown in figure 3. All the data nodes contained within the hierarchical structure of the

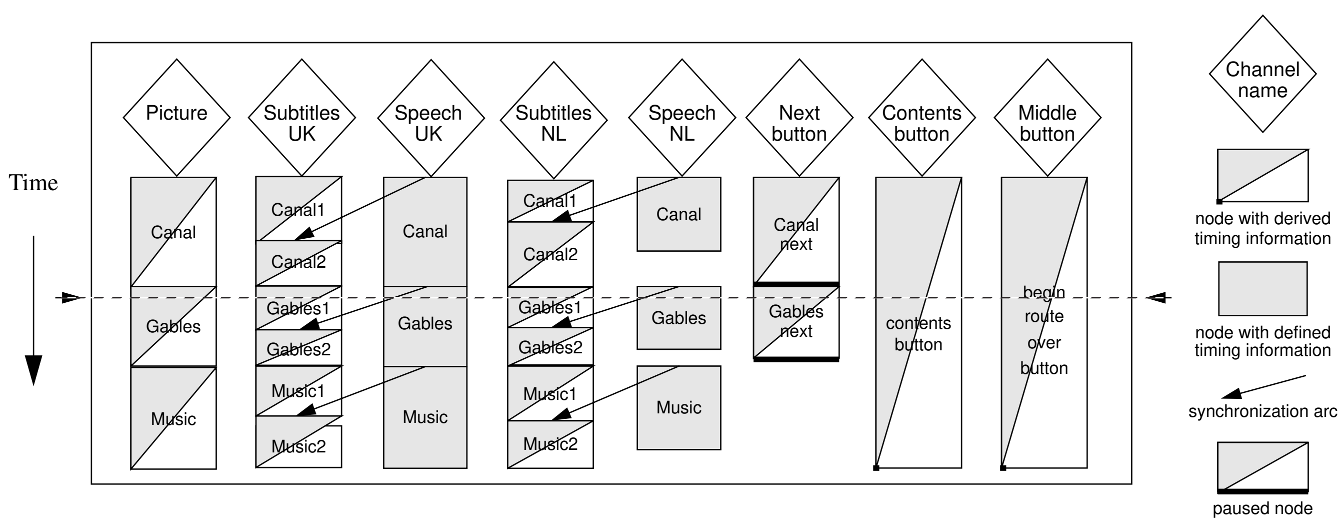


Figure 5. Channel View for the *Walking route* sequence.
The diamonds at the top of the figure show the channel names. The data nodes assigned to the channels are represented as boxes beneath the diamonds. The height of a box represents its duration. A fully-shaded box has its duration explicitly defined, either through its data type, for sound and video, or through the author assigning a specific duration. A box with a shaded triangle has inherited its duration from its parent in the presentation’s structure.

scene are shown in the channel view. The correspondence between the structure shown in the hierarchy view, figure 3, and the data nodes in the channel view, figure 5, is shown in figure 6.

When the multimedia presentation is played the ordering of the data nodes is taken directly from the channel view—time runs from top to bottom and nodes on all the active channels are played in parallel. For example, the data nodes intersecting with the dotted, horizontal line in figure 5 are those presented during the gables clip shown in figure 1.

While the majority of the timing relations between data nodes are derived satisfactorily from the hierarchy view, the author may wish to explicitly state some timing constraints. Synchronization arcs, represented by the lines with arrowheads in figure 5, allow the author to impose explicit timing constraints between media objects. For example, the display of the text line of the *Gables2* subtitles in figure 5 coincides with the utterance of the associated phrase, somewhere within the *Gables* audio object. (For further details of the information contained in synchronization arcs see [2].)

Within the channel view the author can select any object (channel, data node or synchronization arc) to interrogate it for more information, such as the object's attributes. Synchronization arcs and channels can be created and deleted. Data nodes can only be interrogated, since they are defined in relation to the presentation's structure, and so can only be created and deleted in the hierarchy view.

As well as the authoring facilities available from the channel view, it also dynamically highlights the data nodes as they are being played. When the system has sufficient time it looks ahead in the presentation and fetches data that will be needed. The stages of this pre-scheduling are also shown by highlighting the data nodes in the channel view. Further details on the scheduling are given in [8].

4.2.2 Using the channel view

We have already shown in section 4.1.2 how to create the structure for a multimedia presentation. We now demonstrate how synchronization arcs are created. We will create an arc between the voiceover and the subtitles shown in figure 7. First a node is selected and the selection is locked, highlighting the node. A second node is then selected and the create synchronization arc command is given. This brings up a dialog box which allows the timing of the synchronization arc to be specified, and whether the arc goes from or to the beginning or end of the data node.

Once the synchronization arc has been created its attributes can be changed by selecting it directly and requesting the attribute dialog.

4.3. Player

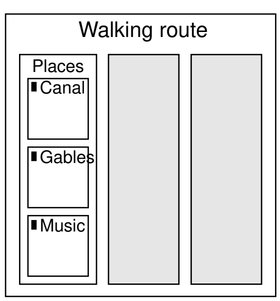
Figure 1 shows the gables clip from the walking route scene as it appears in the player. The player provides facilities, such as start, pause and stop, for the author or the end user to control the playing of the multimedia presentation.

An important role for the player is to interpret the system-independent specification of the multimedia presentation (in terms of the presentation's structure, logical resource allocation and timing constraints) and actually play the presentation on the available hardware. This process is described in detail in [8].

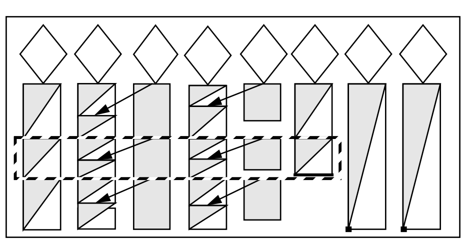
From the authoring perspective, the player is closely integrated with the hierarchy and channel views which allows the author to play any part of the presentation to check how it will appear to the end user. The selection in the hierarchy view (composite node or data node) or the channel view (data node) can be played immediately in the player. This facility allows the author to preview a small section of the presentation without having to play from the beginning of a long sequence.

The author has direct access to data nodes via the player and can interrogate them and change information which relates to a particular node. The author can select a node then call up a menu with a choice of options, including editing the node's attributes, highlighting the node in the hierarchy view or channel view, or choosing an appropriate media editor for that node. The placement of the windows for screen based channels is also carried out via the player.

The player also allows the user to select which channels should be played, for example to select one of a number of voice-overs in different languages. In figure 1 the subtitles in either or both languages can be turned off, and the user can choose to listen to a Dutch or an English soundtrack.



(a) Figure 3



(b) Figure 5

Figure 6. Correspondence between hierarchy and channel views
Data nodes contained in the nested structure of the *Gables* section in (a) correspond to the data nodes enclosed by the stippled box in (b).

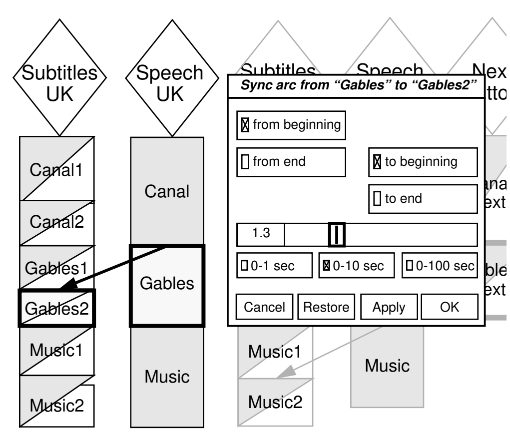


Figure 7. Creating a synchronization arc.
The *Gables* and *Gables2* nodes have been selected and the synchronization arc attribute dialog has been brought up. The synchronization arc shown specifies a delay of 1.3 seconds from the beginning of the *Gables* node to the beginning of the *Gables2* node.

5. HYPERMEDIA

While this paper has emphasized a structured authoring approach for multimedia, the CMIF authoring environment has been built to support hypermedia, that is a collection of multimedia presentations through which the reader can navigate using hyperlinks. We have already implemented single source, single destination links between anchors defined in data nodes. We also allow the destination of a link to be a composite node. In the former case the user is taken to the new presentation as if the complete presentation had been "fast-forwarded" to the destination anchor. This does not support exactly the hypermedia model as described in [4]. For this, we need to implement issues such as context for anchors, where an anchor is not only defined in relation to a data node, but is also associated with a composite node that can be retained or discarded when a link from the anchor is followed.

Authoring hypermedia presentations requires even more support than multimedia, since we require more than just the creation of multimedia presentations: the reader can choose one of a number of paths through the complete presentation and is unlikely to see the complete collection of presentations. We believe the current authoring environment is powerful enough for the specification of structured hypermedia presentations, although the tools we currently have need to be extended to give the author a structured view of the links.

6. EXPERIENCES

Through the development of the CMIF author, and the construction of various demonstration applications, we find the structured approach to authoring natural and easy to use, although it perhaps requires a greater learning curve in the beginning. Although we are generally satisfied with the environment there are a number of improvements we would like to make to the interface which we list here.

It is a matter of debate how much information should be displayed in the hierarchy and channel views. In some cases we would like to show the structure in the channel view, although this could prove very confusing. We would also like to show in the hierarchy view whether a data node has an inherited time or its own defined duration. All this information is available, but the author first has to change views, and set up the focus, to see it.

Another possible enhancement would be to display the data from the object itself in either the channel or hierarchy views. The benefits are questionable. The advantage would be that the author can immediately see which picture or video is being used. The problems would be how much extra time it would cost to access and display the information, and how would the aspect ratio be handled in the (automatically drawn) views. The current system doesn't show the data directly, but allows the author to play it on request.

One drawback which is encountered fairly often is the desire to be at two foci at once in the hierarchy view. This can be because some small structure is being copied from one place to somewhere else, or because a hyperlink is being authored between two parts of the presentation. Our current opinion is that two separate hierarchy views on the same structure would be a solution.

When creating a presentation with different language options it would be much more convenient to be able to turn sets of channels on and off from the presentation itself, for the example in figure 1 we would allow the reader to choose to listen to English or Dutch. The system would make sure that the one language was spoken and the other turned off.

In building a larger demonstration we had the need for different collections of channels which formed a layout on the screen (as in figure 1). It would be very convenient to have the layout as an object within the system, so when choosing channels to assign data nodes to it is clear which layout they belong to.

7. CONCLUSION

The task of creating a multimedia presentation is multi-levelled and time-consuming. In order to support the author in this task we have created the CMIF authoring environment for creating and editing multimedia presentations. This environment supports a structured approach to authoring multimedia, allowing the author to view and manipulate the presentation in the way best suited to the authoring task. The hierarchy view allows the editing and viewing of the presentation's hierarchical structure, from which the timing constraints are derived. The channel view allows the viewing of logical resource use by the presentation and the specification of precise timing constraints. The player allows the user to play parts, or all, of the multimedia presentation, and to activate and deactivate channels.

The current system has been used to author different types of presentations, for example hypertext-style documents with many links and few long sequences, and longer linear multimedia presentations. This variety in demonstration examples allows the authoring environment to be improved without biasing it to one particular style of editing.

Acknowledgements

Other members of the multimedia group at CWI who have contributed to the work on the CMIF editor are Jack Jansen, Robert van Liere, Sjoerd Mullender and Dik Winter.

References

[1] *M Cecelia Buchanan and Polle T Zellweger*, "Specifying Temporal Behavior in Hypermedia Documents", ECHT '92 (Proceedings of the ACM Conference on Hypertext), Milano, Italy Nov 30 - Dec 4 1992, 262 - 271.

[2] *Dick C A Bulterman, Guido van Rossum and Robert van Liere*, "A Structure for Transportable, Dynamic Multimedia Documents", USENIX conference June 1991 Nashville TN, 137 - 155

[3] *Kazutoshi Fujikawa, Shinji Shimojo, Toshio Matsuura, Shojiro Nishio and Hideo Miyahara*, "Multimedia Presentation System 'Harmony' with Temporal and Active Media", USENIX conference June '91 Nashville TN, 75 - 93

[4] *Lynda Hardman, Dick C A Bulterman, Guido van Rossum*, "The Amsterdam Hypermedia Model: Extending Hypertext to Support *Real* Multimedia", Hypermedia, May 1993, 5(1).

[5] "Director version 2.0", MacroMind 1990 (dynamic media authoring tool for the Apple Macintosh)

[6] *Michael Mills, Jonathan Cohen and Yin Yin Wong*, "A Magnifier Tool for Video Data", CHI '92, May 1992 Monterey CA, 93 - 98.

[7] *Ryuichi Ogawa, Hiroaki Harada and Asao Kaneko*, "Scenario-based Hypermedia: A Model and a System", ECHT '90 (First European Conference on Hypertext), November 1990, INRIA France, 38 - 51

[8] *Guido van Rossum, Jack Jansen, K Sjoerd Mullender and Dick C A Bulterman*, "CMIFed: a Presentation Environment for Portable Hypermedia Documents", Multimedia '93.

[9] *J William Semich*, "Multimedia Tools for Development Pros", Datamation, August 15 1992, 90 - 95.