A Visual Authoring Environment for Prototyping Multimedia Presentations

Ombretta Gaggi and Augusto Celentano
Dipartimento di Informatica
Università Ca' Foscari di Venezia
Via Torino 155, 30172 Mestre (VE), Italia
{ogaggi,auce}@dsi.unive.it

Abstract

In this paper we describe an authoring environment which allows the author to set up and test a complex multimedia presentation by defining the synchronization relationships among media. The main component of the authoring environment is a visual editor based on a graph notation, in which the nodes are media objects involved in a multimedia presentation, and the edges are the synchronization relations between them. Several external representations can be generated: a timeline-based representation highlighting media sequencing, and an XML-based description suitable for further processing. An execution simulator helps the author to check the presentation behavior before delivery.

Keywords: Hypermedia authoring, media synchronization, multimedia presentation, execution simulation, World Wide Web.

1. Introduction

The growth of network bandwidth encourages the presence of rich media types, such as animations, audio and video in WWW documents. The integration of multimedia material into hypermedia documents is widely used in many applications like distance learning (e.g., ilearning.oracle.com), web advertising and e-business (e.g., www.sony.com), virtual tourism (e.g., www.360thecity.com), cultural heritage (e.g., www.palazzograssi.it), news delivery (e.g., www.cnn.com), entertainment (e.g., www.warner.com), and so on.

Authoring such documents is a complex task, since the author must deal both with the structure and layout of the document, and with its temporal behavior. The task is more difficult when dealing with interactive documents, since unanticipated user interaction can alter the correct timing relationships between media. In this situation the test of a presentation cannot rely on the execution under all possible circumstances. Rather, simulation of media related events and user interaction should be possible in order to test the synchronization and coordination among media besides their unattended execution.

In this paper we illustrate an authoring environment based on a synchronization model we have presented in [3, 5] and described in detail in [7], which allows the author to set up and test a complex multimedia presentation by defining the synchronization relationships among media. The main component of the authoring environment is a visual editor based on a graph notation, in which the nodes are media objects involved in a multimedia presentation, and the edges are the synchronization relations between them, defined upon the occurrence of events like begin and end of the media components.

An execution simulator helps the author to check the presentation behavior not only during a normal continuous play, but also in presence of user interaction like pausing and resuming a presentation, stopping a media playback, or following a link in a hypermedia document. Several external representations can be generated: a timeline-based representation highlights media sequencing, while an XML-based description can support several purposes, like translation into SMIL or other languages for delivery, querying for multimedia retrieval [5] and automatic generation of presentations from templates [4].

This paper is organized as follows. After briefly reviewing the relevant literature in Section 2, Section 3 introduces the media synchronization model on which the authoring system is built. The visual authoring system itself is illustrated in Section 4. Section 5 presents the XML language produced by the authoring system. Section 6 discusses the features and the use of the execution simulator, and Section 7 draws some concluding remarks.

2. Related work

Many metaphors are proposed by existing authoring tools. The simplest is the timed-based metaphor, used by Adobe Premiere [1]: multimedia elements are presented and organized in tracks along a time line. Macromedia Director [12] implements a theatrical metaphor in which text, audio and other media objects are cast members on a stage, and the score is a sequencer which animates the actors. These metaphors are simple and intuitive, but are not easy to manage and maintain, since a modification to the time of an event can require to adjust the time relationships between several objects.

Macromedia Authorware [11] uses a flowchart-based paradigm where media objects are placed in sequence and grouped into sub-routines, like commands in procedural programming. With this metaphor the author cannot explicitly define the time intervals ruling the multimedia presentation, which are computed according to the execution order of the media components.

Microsoft Producer [13] is useful to create web presentations which synchronized an audio or a video file with a set of slides, i.e. images or HTML pages. It divides video files into scenes and helps the author to places the images on a timeline by automatically calculating their download time. Transition effects can be added to enrich the presentation. Some templates are available to design the layout of the presentation. In a very similar approach, Accordent's PresenterOne [14] creates the same type of presentation synchronizing audio and video files with static media such as web pages or images.

Besides commercial products, many research works have designed multimedia authoring models based on different paradigms, able to create and manage temporal scenarios.

In [2] Bulterman et al. present GRiNS, a GRaphical INterface for creating SMIL documents [16]. GRiNS provides three views, the *logical structure* view to define the temporal structure, the *virtual timeline* view to define and adjust fine-grain temporal interaction and the *playout* view to preview the presentation behavior and to define the spatial layout.

Madeus [10] is an authoring and presentation tool which uses graphs to represent both temporal and spatial constraints of a multimedia document. Graphs are used also for scheduling and time-based navigation.

The Hypermedia Presentation and Authoring System (HPAS) [18] is based on the Media Relation Graph, a simple and intuitive hypermedia synchronization model. A hypermedia presentation is modelled by a direct acyclic graph, where vertices represent media objects, and directed edges represent the flow of time.

HyperProp [15] offers three graphical views of the document: the structural view, the temporal view and the spatial view. The author can use the structural view to browse and edit the logical structure of the document, the temporal view to represent the objects along a timeline and the spatial view to preview objects layout.

The MPEG-4 standard [8] has solicited a number of authoring tools for producing coded media streams and scene descriptions. A critical review of the issues related to the specific aspects of MPEG-4 standard is in [17].

iVAST Studio Author [9] is a commercial product, a visual environment dedicated to the creation of interactive MPEG-4 content that provides fine-grain control over audio, video, 2D graphics, animation and interactivity. It allows the user to build rich media presentations using familiar graphical user interface paradigms.

Envivio Broadcast Studio [6] is another commercial product for MPEG-4 composition and post-production. It supports both spatial composition and temporal composition of media objects.

3. A Model for Hypermedia Presentation

The authoring environment we present is based on an underlying model for multimedia presentations we have defined and discussed in [3, 7, 5]. The model is based on a small number of synchronization primitives, but in spite of its simplicity it is able of describing a large spread of multimedia presentations. We briefly review the components of the model that are necessary for understanding the editor and the overall philosophy of the authoring environment.

A hypermedia document is modelled as a set of modules hierarchically organized. We refer to a module as a container of continuous media, like video or audio clips, and non-continuous media like images or text pages.

Continuous media are organized in *stories* which are made by *clips*, i.e., media files which are played continuously. A clip is divided into *scenes*, which are the media units to which synchronization events are associated. Static media objects, i.e., texts and images, are called *pages*, and are synchronized by the continuous media behavior. As a clip plays, scenes are played in sequence, and generate events (like media start and media end), which cause the other media to be synchronized and played.

Each medium requires a device to be rendered or played. A *channel* is a virtual device which is used by one document component at a time, for all the duration of its playback. A channel can be a new window browser on the user screen, a frame in a window, an audio channel, or a combination of video and audio resources like the one required to play a movie with an integrated soundtrack.

Synchronization inter and intra modules is achieved by five synchronization primitives, which describe the behavior of the presentation by defining the object reaction to events.

An *internal* event is generated by a component of the presentation, e.g., the start or the end of an object playback. An *external* event depends on external actions, like a user interaction.

The parallel composition of two objects is described by the relationship "*A plays with B*", symbolically $A \Leftrightarrow B$, which states that, if one of the two objects is activated by the user or some other event, the two objects play together. Object *A* acts as a "master" in this relation: as soon as it ends, object *B* terminates too, if it is still active.

The sequential composition of two objects is defined with the relation "*A activates B*", symbolically $A \Rightarrow B$. When object *A* comes to its end, object *B* begins playing. If object *A* is terminated before its natural end, e.g., another object or the user stops it, object *B* is not activated.

The "*plays with*" and the "*activates*" relationships can describe the behavior of rather complex presentations. They roughly correspond to the par and seq tags of SMIL, but some differences exist which are not discussed here. However they do not consider user interaction, for which we introduce three additional synchronization relationships.

The relation "*A is terminated with B*", written $A \Downarrow B$, states that the termination of object *A* before its natural end forces object *B* to terminate too. It is usually used to propagated a "stop" of the user from an object to all the other objects which play in parallel, and to model skipping forward or backward inside a timed presentation.

The relation "*A is replaced by B*", denoted by $A \rightleftharpoons B$, says that when object *B* is activated, it forces *A* to terminate so that its channel can be freed and used by *B*. This relation is used mainly with static objects, whose time length is in principle infinite.

The presentation behavior in presence of a hyperlink involving continuous media can be described by the relationship "*A has priority over B with behavior* α", symbolically written $A \overset{\alpha}{>} B$. This relationship means that object *B* is paused (if $\alpha = p$) or stopped (if $\alpha = s$) when object *A* is activated; *A* is supposed to be the target of a hyperlink that moves the user focus from the current document to another document or to another presentation. When object *B* comes to the end, object *A* is resumed, if it was paused.

The reader is referred to the cited works for a discussion of details and motivations about this synchronization model.

4. A Visual Authoring Environment for Hypermedia Documents

We have developed a complete visual environment for authoring and prototyping multimedia presentations, using the Java language. Due to the event-based style of the synchronization relationships, a graph notation was a natural choice for the visual editor: media are represented as nodes, and synchronization relationships are represented as typed edges. Authoring consists mainly in drawing the synchronization graph of the presentation, and all the temporal aspects of the presentation are defined by manipulating the graph. The editor provides also facilities to design the screen layout and to define the playback channels, to simulate the presentation dynamics, and to generate an XML descriptions suitable for translation into a standard delivery language like SMIL.

The visual editor does not provide functions for editing the media components, but only to assemble the corresponding files: a wide choice of good programs for digital media manipulation is commonly available.

4.1. The User Interface

The visual editor provides two views to create and edit a multimedia presentation: one to define the channels and the layout of the presentation, and one to define the temporal behavior.

Channels can be of two types: regions, i.e., screen areas, and audio channels. The user defines the size of the multimedia presentation window, and creates the regions by drawing rectangles on it; each region has a unique name. The audio channels are defined only by assigning them a name. Colors distinguish the different regions, and are used consistently in the authoring process. Regions can be moved and resized by direct manipulation.

The temporal behavior view provides the author with a panel on which to draw the graph which describes the dynamics of the presentation in terms of synchronization among media.

With respect to a time-line description an event-based representation has some drawbacks and many advantages. Even if an event-based description is less intuitive, it allows a better specification of relationships among objects, mainly in an environment like the one we face, i.e., presentations designed to be delivered on the World Wide Web. In the WWW media are delivered independently, and timing is subject to delays due to server and network load; for the applications we address, the synchronization needs not to be fine-grained (e.g., lip synchronization is achieved by merging into the same file the video and the audio tracks), while it is important to address issues like the user interaction, that can modify the temporal behavior of a presentation with stops, reloads, VCR commands provided by players, hyperlink jumps, and so on.

Moreover, while continuous media have a duration defined by their playing time, for static media the concept of duration is fuzzy, since they do not evolve in time, but their life depends on the timing of other media. Therefore a time line specification for hypermedia documents that integrate continuous and non continuous media with user in-

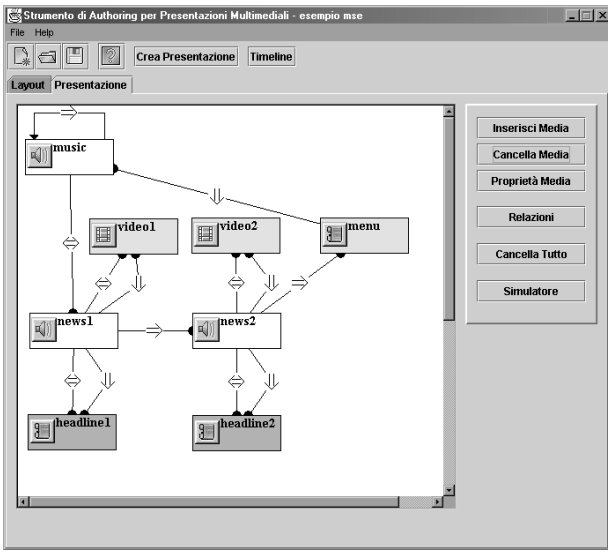


Figure 1. The synchronization of a news-on-demand cover

teraction can only be an approximation of the real behavior. An event-based description is simpler to draw and to adapt, since it allows the designer to concentrate on the relationships between media-related events without anticipating the actual behavior of the media items.

Each media object has some attributes: a unique name, a link to the corresponding file, the media type, (text, image, video, etc.) and the channel used for its playback. It is drawn as a rectangle with an icon which identifies the type. The rectangle is colored like the associated region (hollow for audio objects) to show the channel used. Consistency between channel definition and usage is checked by the editor.

Synchronization relations are established by selecting the two objects and the relation from a menu. The relation is drawn in the graph as an edge from the first to the second object, labelled with the icon of the relation type.

As an example, we describe the cover presentation of a news-on-demand Web service which summarizes the articles available. In the presentation a short video of each report is shown, with a headline on the top of the video, while the speaker's voice reads a brief summary of the news. As the speaker reads the following summaries, the headline and the video change their contents. During the presentation a background soundtrack is played continuously. After the last news a text page is displayed, and the user can select the article from a menu.

Figure 1 shows the synchronization graph for such a presentation. Two audio channels are defined, one for the speaker voice and one for the soundtrack, and two regions, one for the video and one for the headline. In figure 1 we consider only two articles, but the synchronization schema can be extended in a straightforward way. The voice of the speaker, the video and the headline of an article are played in parallel as described by the relationships $news_i \Leftrightarrow video_i$ and $news_i \Leftrightarrow headline_i$, with $i = 1, 2$. As the speaker ends reading the summary of the first article, the second is activated by the relationship $news_1 \Rightarrow news_2$. At the end of the second article the relation $news_2 \Rightarrow menu$ displays the article menu. The soundtrack is repeated continuously as modelled by the relationship $music \Rightarrow music$. When starting it activates the first summary ($music \Leftrightarrow news_1$).

It is worth to note that this example, while giving no information about the time length of the media, states that the duration of each article is equal to the duration of the speaker's comment. The relationship $news_i \Leftrightarrow video_i$ in fact causes the video to end its playback when the audio ends. In the same way, the headline, whose duration is in principle infinite since it is a static medium, is terminated when the speaker ends the comment.

Five other relationships are needed to model the situation in which the user stops one of the master media during its playback: the relationships $news_i \Downarrow video_i$ and $news_i \Downarrow headline_i$ with $i = 1,\ 2$ stop the video and the headline if the user stops the corresponding audio news. The relationship $menu \Downarrow music$ forces the background soundtrack to end when the user stops (i.e., closes) the menu, since this action means that the user wants to exit the presentation.

4.2. A Timeline Style View

The authoring tool provides a third view, a timeline style view, that helps the author to check the presentation behavior in terms of sequential and parallel execution of the media objects. However, this view cannot be used to edit the presentation, since it is defined only by the synchronization relationships.

The time-based view translates the synchronization graph into a tree structure which represents the parallel and the sequential composition of the media, much as in a SMIL program. The media objects which play in parallel are drawn as children of a node labelled with the name of the "master" object, i.e., object A in the relationship $A \Leftrightarrow B$. Objects played sequentially are drawn in a father-child relationship. Figure 2 shows the time diagram of the example illustrated in Figure 1.

The tree has a root labelled with the name of the presentation. The first child is the object which starts the presentation playback. If it has children, and one of them is labelled with the same name of the father, the children will play in parallel with it, otherwise they will be activated at the end of the first object playback. The execution continues until the tree has been traversed down to the leaves.

The depth of a node is a measure of its position in the

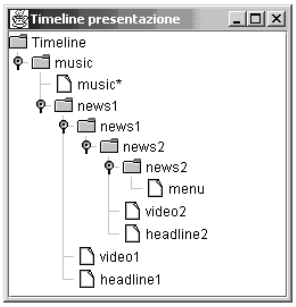


Figure 2. A timeline style view

presentation: objects closer to the root will be played before objects in a deeper position in the same branch of the tree. The tree therefore represents a family of timelines, where the exact duration of each medium is not defined but the mutual occurrences are visible.

The reason why we speak of a family of timelines comes from the user interaction possibility. In a multimedia document which provides user interaction the real duration of each component cannot be stated, since it depends on events external to the natural flow of the presentation, like pausing and resuming it, or jumping forward or backward. In a timeline each event occurs at a specific time, but a family of timelines is able to describe also events occurring as a consequence of other, unanticipated events.

The timeline style view also provides some additional information that are not easily visible in the synchronization graph. E.g., the object which starts the presentation is the root first child, but in the graph diagram is simply a node which is not activated as a consequence of internal events. The tree describes which objects are reachable within the normal flow of the playback (i.e., there is a path from the start of the presentation to the object) and which can be reached only by a user interaction (i.e., such path does not exist and they are drawn as subtrees of the root).

5. An XML Representation of the Multimedia Presentation

The editor is able to export an XML description of the presentation to take full advantages of XML processing tools; e.g., information in XML documents can be presented according to several different presentation styles, and XML query languages provide facilities to retrieve data. We could use an existing language like SMIL[16] to this purpose, but we aim at using such a description not only for playing the presentation in a real execution environment, but also for other purposes, for example to allow the reuse of a coherent fragment of a presentation in other contexts, or for querying a presentation about its content.

```
<presentation xmlns="model.xsd">
  <layout width="500" height="400" >
   <channel name="headline" SupX="19" SupY="13" InfX="471" InfY="321"/>
   <channel name= "voice"/>
    ...
  </layout>

  <components>
   <module id="news">
    <clip id="news1" file="news1.wav" channel="voice" type="audio"/>
    <page id="headline1" file="headline1.txt" channel="headline" type="text"/>
    ...
   </module>
  </components>

  <relationships>
   <play>
    <master><cont_object id="news1"/></master>
    <slave><object id="headline1"/></slave>
     ...
   </play>

   <act>
    <ended><cont_object id="news1"/></ended>
    <actvated><cont_object id="news2"/></activated>
     ...
   </act>
     ....
  </relationships>
 </presentation>
```

Figure 3. A fragment of the XML representation

In order to support content independent structure processing we keep not only multimedia data separated from the structure definition (which is quite obvious) but also structure-related information separated from spatio-temporal information. In most existing models such information is often mixed. For example in SMIL, spatial information is declared separately in the head section, but the synchronization definition includes the declaration of the media objects. Such integrated definition does not encourage object re-use, mainly in complex documents, where it would be especially useful. Redundancy is generated, which requires cross-checking between different document sections.

We have defined an XML schema that organizes data in three sections: the layout section, the components section and the relationships section. The layout section describes the spatial layout of the channels, the components section contains information about the media objects, and the relationships section describes the synchronization among the media. This solution allows the definition of spatio-temporal relationships among media objects without knowing any information about their location or duration.

The layout section contains the definition of the channels used by the presentation. Figure 3 shows an excerpt of the XML file produced for the presentation described above. The channel headline is a rectangular area of the user screen delimited by the corner coordinates SupX, SupY, InfX and InfY. Voice is an audio channel, therefore it has no layout.

Each channel has a unique name.

The components section contains the description of the media involved in the presentation. According to the model described in section 3, media objects are organized into modules enclosing continuous media objects like clips and static media objects like pages. Each element has a unique identifier id, which is used to reference it from other sections, a type (but for modules), a channel and a file, which addresses the actual multimedia data URL. The attributes channel and file are required for clips and pages.

The relationships section describes the synchronization among the objects through a list of synchronization relationships of the presentation. Each relationship type is coded with a different tag, enclosing two children tags for the two sides of the relationship. In Figure 3 the tag play codes the relationship $news_1 \Leftrightarrow headline_1$, where object $news_1$ is defined within the tag master and object $headline_1$ is enclosed in the tag slave. The relation $news_1 \Rightarrow news_2$ is coded with the tag act with children ended and activated.

The relation "is replaced by" ($\rightleftharpoons$) is coded with the tag repl with two children, tags before and after while the relation "is terminated with" ($\Downarrow$) is coded with the tag stop with children tags first and second.

The relation "*A* has priority over *B* with behavior α" is coded with the tag link and the attribute behavior holding the value of the parameter α. The tag from contains the source of the link while the destination is described by the tag to. A more detailed description can be found in [7].

6. Simulating the presentation execution

An execution simulator allows the author to check the temporal behavior of the presentation. The simulator places the media placeholders in the appropriate channels they would use in the real execution. Then, without being compelled to follow a real time scale, the author can generate all possible events, both internal (e.g., the end of an object play), and external (e.g., a user-executed stop or a hyperlink activation) to see how the events are propagated and how the presentation evolves.

The simulator provides two interfaces for two different simulation styles. According to the first style, media placeholders are used to show the presentation dynamics as perceived by the final user. The author needs also to understand how such a behavior is obtained. Therefore, as a second simulation style, the synchronization graph is animated to show which part of the presentation is currently in execution and which relationships are activated by the event triggered by the user.

The simulator opens a new window which lists the media components of the presentation and displays the channels as designed by the author in the layout view. Audio channels, which do not have a layout, are represented as small rectangular areas outside of the presentation window.

The author can select a medium from the list and can send events to it: start, stop and end, thus simulating the beginning, the forced stop or the natural end of the selected medium. Otherwise, the author can select a file of pregenerated events, and see the presentation evolution step by step.

At the beginning of the simulation all channels are free, therefore they are drawn as uncolored areas. Each channel is marked with its name. When the author starts a media item, the channel it would use in the real execution is filled with the channel color, as set in the layout view, and the media item name (i.e., the file name or URL) is displayed in the channel area. A number of incoherences are checked and reported, e.g., if the channel is still busy because it was used by another media item and not released, an error message is displayed.

Then the simulator checks the relationships defined by the author looking for the ones which involve the selected media item (let us call it *A*):

- for each object *B* such as the relation $A \Leftrightarrow B$ exists, the simulator starts object *B*, i.e., it fills the corresponding channel with the channel distinctive color and object *B* name;
- for each object *B* such as the relation $A \rightleftharpoons B$ exists, the simulator first stops object *B* (releasing the channel) and then activates object *A*;
- for each object *B* such as the relation $A \overset{\alpha}{>} B$ exists, the simulator first stops (if $\alpha = s$) or pauses (if $\alpha = p$) object *B* and then starts the execution of object *A*.

A trace of the events triggered and media activated is logged for tuning and debugging purposes.

It is worth to note that the simulation does not involve the actual files that will be used in the presentation, each media object being represented by a visual modification of the associated channel appearance. An improvement is under development, which allows for displaying an image or a text, taken from the media object file, if available, in order to enhance the user perception of how the presentation will evolve.

The author can end or stop an active media. In both cases, the simulator releases the corresponding channel by removing the channel color and replacing the object name with the channel name. Then, as for media start, it analyzes the relationships between the ended or stopped media item and the other media items. If the author ends media item *A*:

- for each object *B* such as the relation $A \Leftrightarrow B$ exists, the simulator ends object *B*;

- for each object B such as the relation $A \Rightarrow B$ exists, the simulator activates object B;
- for each object B such as the relation $A \overset{p}{>} B$ exists, the simulator resumes object B.

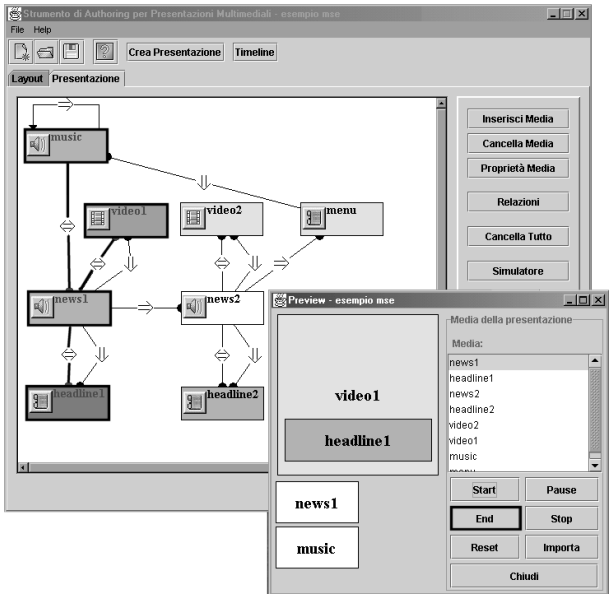
If the author stops a media item A, the simulator forces the termination of all objects B for which the relation $A \Downarrow B$ holds.

In this way the author can see in every moment which channels are busy and which media use them. The interface provided is a simulation of real media layout and dynamics during the presentation playback. However, understanding which part of the synchronization graph is currently executed is not simple. Except for the name of the media active in the channels, no other information is provided. In order to improve the user perception of the events and their relationships with the media, the simulation animates also the graph of the presentation: when the user activates an object, the corresponding node in the media-relationships graph is highlighted. Then the relations triggered by the activation of the object are also highlighted, and their effect is propagated to the related objects, giving the author, in a short animation, the visual perception of how media synchronization is achieved. If the author ends or stops a medium the graph is animated by showing how this event is propagated to the related media objects.

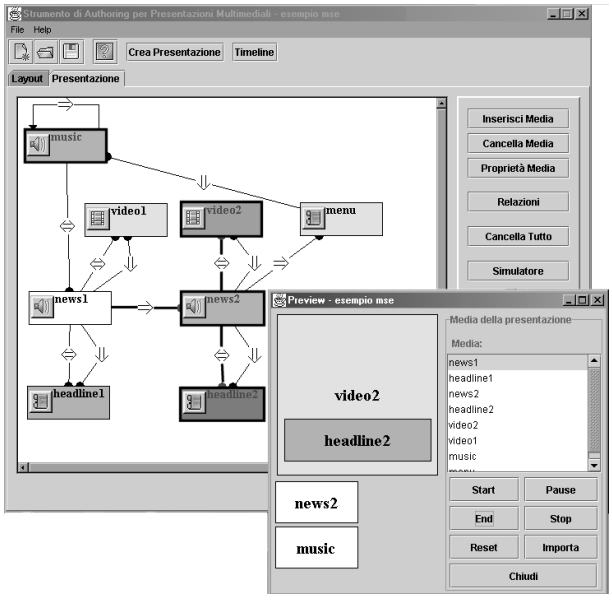
Figure 4 shows a step of the simulation of the news-on-demand cover illustrated in Figure 1. In Figure 4a the presentation is playing the first news $news_1$, which plays together with the associated video file $video_1$ and the title $headline_1$[1]. The soundtrack *music* plays continuously. The author simulates the end of $news_1$ by selecting it in the list of media items (on the right in Figure 4a) and clicking on the button "End". The result is shown in Figure 4b: $news_2$ starts as a consequence of relationship $news_1 \Rightarrow news_2$, while the relationships $news_2 \Leftrightarrow video_2$ and $news_2 \Leftrightarrow headline_2$ activate $video_2$ and $headline_2$. The soundtrack continues playing. In the simulator windows the author can see both the result (in the lower right window of Figure 4b, where the media placeholders for the channels *video*, *voice* and *headline* are changed), and the chain of relationships activated (in the left window of Figure 4b). With this representation it is always clear to the author which part of the presentation he/she is simulating, which are the active media and how the end-user interface is organized.

The simulation is independent from the actual media file duration, since the author can simulate any combination of object duration by ending or stopping them explicitly. This allows the author to simulate (therefore to model) a great variety of situations and to design the graph of a presentation even if the length of some related media is unknown.

[1]These objects are highlighted with thick borders.



(a)



(b)

Figure 4. The simulation of a news-on-demand cover

7. Conclusion

In this paper we have described an authoring environment based on a graph representation of a multimedia synchronization model. It provides facilities to define the layout of a hypermedia presentation, to set up the synchronization relationships, to examine the parallel and sequential execution of media, and to save an XML description of the presentation. A simulator allows the author to test the relationships between media by triggering events related to media start and end, and to user interactions like hyperlink activation.

Given a set of pre-generated events, the simulator can show the evolution of the presentation step by step. Our future work will be devoted to study the automatic generation of a sequence of events for systematic testing. The main problem with this feature is the generation of event sequences which are coherent with the actual presentation execution: e.g., the user cannot stop a non active media, and cannot activate media out of their natural ordering unless explicit hyperlinks are provided. In [5] we have described all the possible evolutions of a presentation by means of an automaton which formally describes the presentation states entered by the events which trigger the media playback. Each state of the automaton contains the set of active media, and the corresponding channel occupation. For each state an edge for any event that may occur brings to a new state which captures all the consequences of that event. A possible solution for generating coherent sequences of events could be to retrieve them from finite paths of the automaton.

8. Acknowledgements

The authors acknowledge the contribution of Diego Medici to the development of the authoring tool. This work has been partly supported by Italian Ministry of Education, University and Research (MIUR) in the framework of the National Project *Specification, Design and Development of Visual Interactive Systems*.

References

[1] Adobe Systems Inc. Adobe Premiere. http://www.adobe.com/premiere.

[2] D. Bulterman, L. Hardman, J. Jansen, K. Mullender, and L. Rutledge. GRiNS: A GRaphical INterface for creating and playing SMIL documents. In *WWW7 Conference, Computer Networks and ISDN Systems*, volume 30(1-7), pages 519–529, Brisbane, Australia, April 1998.

[3] A. Celentano and O. Gaggi. Synchronization Model for Hypermedia Document Navigation. In *ACM Symposium on Applied Computing (SAC2000)*, pages 585–591, Como, Italy, March 2000.

[4] A. Celentano and O. Gaggi. Schema Modelling for Automatic Generation of Multimedia Presentations. In *Fourteenth International Conference on Software Engineering and Knowledge Engineering (SEKE2002)*, pages 593–600, Ischia, Italy, July 2002.

[5] A. Celentano, O. Gaggi, and M. Sapino. Retrieving Consistent Multimedia Presentation Fragments. In *Workshop on Multimedia Information Systems (MIS 2002)*, Tempe, Arizona, USA, November 2002.

[6] Envivio, Inc. Envivio Broadcast Studio. http://www.envivio.com/products/ebs.html.

[7] O. Gaggi and A. Celentano. Modeling Synchronized Hypermedia Presentations. Technical Report CS-2002-11, Department of Computer Science, Università Ca' Foscari di Venezia, Mestre (VE), Italy, 2002, submitted for publication.

[8] ISO/MPEG. Overview of the MPEG-4 Standard, ISO/IEC JTC1/SC29/WG11 N2725. mpeg.telecomitalialab.com/standards/mpeg-4/mpeg-4.htm, 1999.

[9] iVAST, Inc. iVAST Studio Author. http://www.ivast.com/products/studioauthor.html.

[10] M. Jourdan, N. Layaïda, C. Roisin, L. Sabry-Ismail, and L. Tardif. Madeus, an Authoring Environment for Interactive Multimedia Documents. In *ACM Multimedia 1998*, pages 267–272, Bristol, UK, September 1998.

[11] Macromedia Inc. Macromedia Authorware. http://www.macromedia.com/authorware.

[12] Macromedia Inc. Macromedia Director. http://www.macromedia.com/director.

[13] Microsoft. Microsof Producer for PowerPoint 2002. http://www.microsoft.com/office/powerpoint/producer/.

[14] RealNetworks, Inc. Accordent's PresenterOne. http://www.realnetworks.com/products/presenterone/index.html.

[15] L. F. G. Soares, R. F. Rodrigues, and D. C. M. Saade. Modeling, authoring and formatting hypermedia documents in the HyperProp system. *Multimedia Systems*, 8(2):118–134, 2000.

[16] Synchronized Multimedia Working Group of W3C. Synchronized Multimedia Integration Language (SMIL) 2.0 Specification, August 2001.

[17] *Workshop on MPEG-4 Authoring, 14th ACTS Concertation Meeting*, Bruxelles, May 1999. http://www.cordis.lu/infowin/acts/analysys/concertation/multimedia/reports/mpeg.htm.

[18] J. Yu. A Simple, Intuitive Hypermedia Synchronization Model and its Realization in the Browser/Java Environment. In *Asia Pacific Web Conference*, pages 209–218, Sept. 1998.