A Synchronization Model for Hypermedia Documents Navigation

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

ABSTRACT

This paper presents a model for describing the synchronization between several media delivered over a network in a Web-based environment. Synchronization concerns the download and the activation of coordinated media files according to the structure of the whole hypermedia document, the playback status of the media objects, and the user interaction. The model defines five synchronization primitives some of which can be automatically deducted from the document structure and the media-related events. The model is oriented to describe applications based on a main video or audio narration to which static or dynamic documents are attached. The model is suited for applications like distance education, professional training, Web advertising and selling, and news-on-demand.

General Terms

Hypermedia, World Wide Web, media delivery, synchronization, interactive presentation

1. INTRODUCTION

In this paper we present a synchronization model for navigating hypermedia documents in a distributed environment like the World Wide Web. The model is targeted to a class of applications where one or more continuous media files (video or audio streams) are presented to the user and, as streams play, other documents (text and images) are sent to the browser and displayed in synchrony with them. The user can interact with the presentation by pausing and resuming it, by moving forward or backward, and by following hyperlinks that lead him/her to a different place or time in the same hypermedia document or to a different document. We use the term of "video-centered" hypermedia to denote such documents.

Even if the model does not cover efficiently all hypermedia document types its reference context is wide, and includes applications like self and distance education, professional training, Web advertising and selling, and news-on-demand. As an example, in a Web application for estate selling a video could show the property and, as details are displayed, text pages describe the main features. From time to time maps and façade prospects are shown, giving the user the possibility to explore specific issues in a flexible way. Another example is the remote delivery of a lesson composed of the teacher's lecture video or audio recording that illustrates the issues discussed in the classroom, together with the set of slides which are shown at proper times during the lecture playback.

We'll refer mainly to distance education with some examples to illustrate our work. As a general framework, we assume that a lesson is delivered to the user as a hypermedia document composed of several modules hierarchically organized. Each module is composed of a continuous media stream (e.g., an audio stream) which, as it plays, makes texts, images and possibly other continuous media documents to be displayed to the user screen. The user may adapt the pace of the lesson to his/her needs by moving in the module or across the modules in a controlled way: e.g., by skipping some parts, playing again other parts, temporarily suspending the lesson to browse supplemental material, and so on. The system re-synchronizes (i.e., delivers to the browser) the different media in order to preserve the consistency of the whole lesson.

The paper is organized as follows: in Section 2 a model for video-centered hypermedia documents is illustrated. Section 3 describes the primitives for synchronized delivery of different media and basic user interaction. Section 4 briefly discusses additional synchronization issues due to user interaction. Section 5 comments the related literature, and Section 6 presents some final remarks.

2. A MODEL OF VIDEO-CENTERED HYPERMEDIA DOCUMENTS

Hypermedia documents are modeled along two directions, one describing the hierarchical structure of the document components, the other describing the synchronization primitives and the delivery dynamics.

A hypermedia document is generally composed of a set of *modules*, which the user can address through some table of contents. For example, a multimedia course is divided into lessons, which can be indexed through a syllabus or a time schedule; a news-on-demand application delivers independent articles selected from an index; an estate selling application lets the user choose the relevant property from a catalogue, and so on.

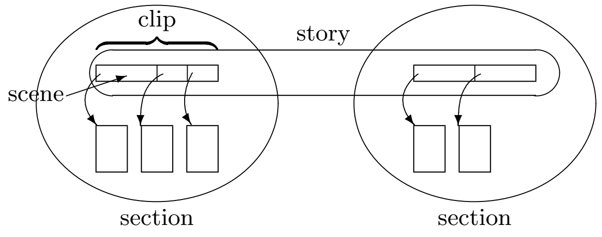

Figure 1: Structure of video-centered hypermedia document

Coming back to the distance education scenario, we assume that from the presentation dynamics point of view each lesson is completely autonomous and self-contained. We do not elaborate further on this level of access since it is not relevant for delivery and synchronization issues, and assume a module, whatever be its definition, as the topmost level of aggregation of media components which is relevant for our discussion.

Borrowing the terminology from document structuring, we call *chapter* such a module. A chapter is divided into *sections*: a section is a multimedia document which is normally played without requiring user intervention, while moving from a section to the next one can be automatic or not depending on the document purpose and application context. Sections could be addressable from an index as chapters are, but still we shall not enter into such details.

As a dynamic document, a section is composed of continuous media clips and static pages of text and images. Section playback is ruled by the continuous media, with static pages delivered and displayed at specific narration points. A section is thus divided into smaller components that correspond to the time intervals between two narration points in which one or more static documents must be delivered to the user.

If we analyze the structure from a dynamic point of view, we can better refer to a terminology borrowed from movies context. The continuous media presentation which constitutes the main chapter content is a *story*, which is made of a sequence of *clips*[1], each of which corresponds to a section of the chapter. Clips are divided into *scenes*, each of which is associated to a different static document, e.g., a text page or an image, which is displayed during the scene playback. The term *narration* introduced above denotes the playback of the continuous components of a hypermedia document. Figure 1 pictorially shows this structure. More complex structures with nested multimedia components will be introduced later.

2.1 Logical vs. physical structure

The structure illustrated in figure 1 is a logical structure, which is in principle unrelated to the physical organization of the media objects. A video file could correspond to a whole clip, to a scene within it, or to a sequence of clips that can be played continuously or as independent segments. Indeed, the correspondence between logical and physical organization should be hidden in an implementation layer, without influencing the model organization. In practice this cannot be done without paying a price in terms of performance and manageability even in a local environment (e.g., a CD-ROM based multimedia presentation).

We distinguish between two kinds of media objects: continuous media files and document files like texts and images. Continuous media files are video and audio recordings, usually called *videoclips* or *audioclips*. Texts and images are contained in static documents usually called *pages* in the WWW parlance[2].

A correspondence between the logical and the physical structure exists:

- a clip, i.e., the narrative content of a section, is a video or audio file which is assumed to be played continuously from beginning to end, unless user interaction modifies this behaviour;

- a static document is a file which is supported and displayed as a whole by the browser, both directly or through a plug-in or helper application. We implicitly refer to HTML documents using the generic term of *page*, without entering in further detail about other formats (XML, PDF, etc.).

From the above assumption it comes that one logical clip (i.e., a section) is implemented by one media file. Scenes are defined by time intervals in the clip: they must be contiguous and are played one after the other. Also pages are bound to specific contents associated to a scene. We can thus say that a scene with its associated page is the smallest amount of information that is delivered as a whole, and a clip and its associated set of pages are the smallest self-consistent and autonomous information with respect to the application domain.

A page must be associated with a scene in order to be displayed. According to the synchronization primitives that we shall discuss later, playing a scene causes the corresponding page to be displayed. Conversely, accessing a page via a hyperlink causes the corresponding scene to be played. Coherently with the class of applications we want to model, a master-slave relationship is established between the dynamic and the static components of a hypermedia document: the dynamic components control the hypermedia document playback, and the static components are synchronized accordingly. We shall come back to this matter later.

The model can also be applied to the synchronization of more than two media, for example a video, a background music, and a set of pages. A master-slave relationship is establish not only between the video and the text pages, but also between the video and the music track. In general, one of the media is defined as the master medium for the whole presentation, and the other are played according to the events defined and raised by the master.

2.2 Channels

Media objects require proper *channels* to be displayed or played. Channels are basically windows or frames on the user screen[3]. Several compatible media objects may share

[1]The reason for such a name will be clear later.

[2]We do not consider text documents with embedded dynamic components like, e.g., video sequences. To be precise, we do not consider them as "dynamic" with reference to the control of the whole hypermedia delivery or playback.

[3]Audio channels do not use windows, but must be handled the same way.

in different times the same channel; the model must describe how channels are reserved, used ad released during hypermedia documents delivery.
A channel is a (virtual) display or playback device like a window, a browser frame, an audio channel or an media player, that can be used by one media at a time. The model does not enter into details about channels properties, the only relevant properties being an identifier that uniquely identifies it on the user workstation, and a type that defines the media types that can use it.
Channels are defined and associated to media objects at design phase, and defaults are established consistently with the WWW environment (e.g., the main browser window is the default visual channel for all the media).
A channel is *busy* if an active media is using it, otherwise it is *free*. Free channels may still hold some content: for example at the end of a movie playback the movie window can still display the first or the last frame.

3. SYNCHRONIZATION AND MEDIA DELIVERY RELATIONSHIPS

If the media objects are the elements of a hypermedia document, the synchronization primitives are the rules that describe their behaviour. A distributed hypermedia document is composed of several media objects that interact based on events raised by their execution. An event is the occurrence of a change in an object state which affects the presentation, e.g. the start of a movie clip, the download of a text page, the end of an audio track, or a user command that moves to another movie frame or activates a hyperlink. In order to keep the model as simple as possible, we define only three events: the *activation* of a media object, the *deactivation* due to normal playback termination, and the *forced deactivation* due to external causes like a user action or a synchronization action with other objects. We do not consider events like windows resizing, or changes in the visualization properties since they do not modify the synchronization relationships among the document components.
In order to define the synchronization relationships we must first define the concept of *active media object* with reference to a WWW-based environment. A continuous media object is *active* when it is playing at the user site, and becomes *inactive* when playback terminates, both naturally and by explicit user action [4].
For non continuous media the concept of active object is more conventional than substantial. For example, a text page is active while it is displayed; indeed, it should be considered active only while the associated scene is playing, even if it is still displayed after the scene ends. In order to simplify the model we associate to a static object an unlimited time length, and give the synchronization primitives the role of removing it from the used channel when no longer needed.
The synchronization primitives are defined as relationships between media objects. We define five synchronization primitives:

- A activates B, denoted by $A \Rightarrow B$
- A plays with B, denoted by $A \Leftrightarrow B$
- A is replaced by B, denoted by $A \rightleftharpoons B$
- A has priority over B, denoted by $A \succ B$
- A is terminated with B, denoted by $A \Downarrow B$

[4]The actual implementation with current WWW servers and browsers assumes that a streaming technology is used to deliver such media, instead of downloading and local playing, thus associating the active phase both to the object download and playback.

which are detailed in this section and in the next one. Some of these relationships need to be explicitly defined by design when structuring the hypermedia document, while other can be automatically inferred. Globally, they define the synchronized behaviour of the whole hypermedia application during sequential play and user navigation.

3.1 Basic synchronization primitives

Two basic relationships must be defined for the hypermedia design phase.

Definition 1. Let A and B be two continuous media objects. We define "A activates B", written $A \Rightarrow B$, the relationship such that object B is activated (i.e., played) immediately after deactivation of object A due to normal termination. If A and B are built on the same media (e.g. both are videos) B may occupy the same channel as A since it is free after A termination.

Definition 2. Let A and B be two generic media objects. We define "A plays with B", written $A \Leftrightarrow B$, the relationship such that the activation of A or B, i.e., the beginning of playback (for continuous media) or display (for static media), causes the other object to be activated too. Each object occupies a different channel.

The "plays with" relationship is not symmetric: there is a master-slave dependence between one object, say A, which plays by virtue of a user action or by synchronization with another object, and the other object, say B, which plays by virtue of the $\Leftrightarrow$ relationship. If A is put into active state, it puts B into active state too, and viceversa.
The activation of the whole hypermedia document is always initiated by the user that accesses the starting point through an index. According to the structural model the starting point is a chapter or a section, therefore it is a story or a clip. If it is a story, the first clip of the story must be activated. The "plays with" relationship $\Leftrightarrow$ can therefore be automatically *inferred* (as a form of inheritance) between a story and its first clip, with the story acting as the master. Similarly, the same relationship is inferred between a clip and its first scene.
The "activates" relationship $\Rightarrow$ can be inferred between the scenes of a clip, since a clip is played continuously (in absence of user interaction). As a result, only the relationships between a chapter and a story and between the clips of a story have to be defined explicitly.
Synchronization between continuous and static documents (i.e., scenes and pages) must be defined explicitly at design phase, since in principle pages are not ordered: their ordering comes as a result of scene ordering and scene-page relationships. We introduce them on a working example used consistently in the remainder of the paper. The example models a small lesson in a self-learning application.
The lesson is composed of one chapter, therefore of one story, composed by two video clips $c1$ and $c2$ which must be played one after the other, and by five text documents (pages) $p1$

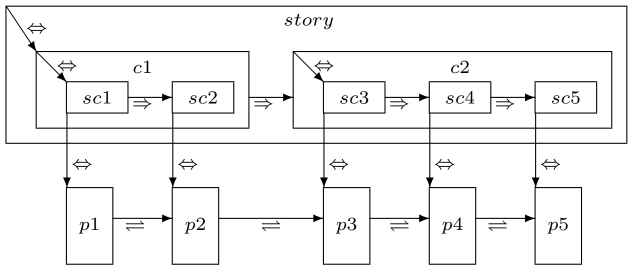

Figure 2: A synchronization example

through $p5$ which must be displayed during the clips playback. Clip $c1$ is composed of two scenes $sc1$ and $sc2$, whereas clip $c2$ contains three scenes, $sc3$, $sc4$ and $sc5$. Each scene must be associated to a different text page.

Once accessed by the user (through some index), the chapter makes the story to become active. Since we are still at the beginning of the hypermedia document delivery, this action is not a true synchronization action. A "plays with" $\Leftrightarrow$ relationship exists between the story and its first clip $c1$, and between it and scene $sc1$. Both relationships are automatically inferred by the story-clip-scene hierarchical structure. Scene $sc1$ activation must in synchrony display the first page of the accompanying text $p1$. Similarly, $sc2 \Leftrightarrow p2, \ldots, sc5 \Leftrightarrow p5$. Finally, since the two clips of the story must be played one after the other, a relationship $c1 \Rightarrow c2$ must be explicitly defined.

Figure 2 pictorially shows these relationships. The figure contains also other relationships ($\rightleftharpoons$) needed to further specify the dynamic behaviour, which are automatically inferred by the lesson structure and will be introduced in next section.

3.2 Media delivery and channel synchronization

The relationships introduced in previous section describe also the media delivery requirements. Document components corresponding to physical media objects, i.e., sections (clips) and pages, must be sent to the user before being activated. From Figure 2 we can deduct that accessing the story requires the first clip to be activated. Since it is a file, it must be delivered (downloaded or streamed) before activation. When activated, its first scene is also activated, and this requires the first text page to be activated too. Therefore page $p1$ must be downloaded to the user browser.

Each media object knows which channel to use. If it is busy, the system can free the channel by deactivating the object that keeps it, or can create a new channel instance (e.g. a new window or audio track) for the media playback. The choice is of course not arbitrary, since it depends on the presentation meaning, therefore it must be described in the synchronization model.

In our example the video presentation uses channel $cn1$ (a browser frame) while the text pages are displayed in channel $cn2$ (another browser frame). When the first scene is played in channel $cn1$ page $p1$ is displayed in channel $cn2$. When scene $sc1$ terminates, playback continues with scene $sc2$ in the same channel. Since $sc2 \Leftrightarrow p2$, page $p2$ must be activated, but channel $cn2$ is still busy because page $p1$ is active. We need therefore to introduce additional synchronization primitives to define channel deallocation.

Definition 3. Let A and B be two objects of the same media type. We define "A is replaced by B", written $A \rightleftharpoons B$, the relationship such that the activation of B forces A to go into an inactive state, i.e., to terminate its playback or display. B can occupy the channel released by A.

Figure 2 shows, the relationships $p1 \rightleftharpoons p2$, $p2 \rightleftharpoons p3$, $p3 \rightleftharpoons p4$ and $p4 \rightleftharpoons p5$ describe how text pages replace each other by sharing the same window. In order to keep the whole presentation semantically consistent, replacement may happen only if the replaced document is associated to a scene no longer active, and can thus be moved to the inactive state without problems. For this reason the $\rightleftharpoons$ relationship need not be defined at design phase; it can be automatically inferred from the other relationships.

Referring to figure 2, given four objects A, B, C and D, whenever relationships $A \Rightarrow B$, $A \Leftrightarrow C$ and $B \Leftrightarrow D$ hold and C and D are compatible media types sharing the same channel, we can derive the relation $C \rightleftharpoons D$. In plain words, pages associated to sequential scenes are correctly assigned the output channel at right time. Other cases can be defined where additional synchronization relationship can be inferred from the ones defined at design phase, but we cannot examine them at deep extent in this paper.

3.3 Following hyperlinks

Text pages may embed references to other documents which are external to the current hypermedia document. If the user jumps, via a hyperlink, to another static document (say $p6$ from $p3$), it can be displayed in another window, or can use the same window used by current text document, replacing it. To specify this behaviour we need to introduce a $\rightleftharpoons$ relationship between the origin and the destination of a hyperlink to a compatible object. This relationship can be inferred from the presence of a hyperlink[5].

By dynamically propagating the effect of this relationship, we can also infer $p6 \rightleftharpoons p4$, which specifies the synchronization action to be performed at the end of scene $sc3$ play. The modeling of the lesson is complete.

We resume the synchronization relationships defined, noting whether they are defined at design phase or inferred from other repationships. Figure 4 illustrates the whole set of relationships. The arcs labeled with the symbol $\Downarrow$ are described in Section 4.

[5]A different solution should be adopted if the hyperlink refers to the target document as a new window.

- $story \Leftrightarrow c1$ for association between the story and clip, which is inferred from the document structure;
- $c1 \Leftrightarrow sc1$ for association between clip and its first scene that is inferred from the relation above;
- $sc1 \Leftrightarrow p1$, $sc2 \Leftrightarrow p2$, $sc3 \Leftrightarrow p3$, $sc4 \Leftrightarrow p4$ and $sc5 \Leftrightarrow p5$ for associations between scene and page that must be explicitly defined by the authors of the presentation;
- $sc1 \Rightarrow sc2$, $sc3 \Rightarrow sc4$ $sc4 \Rightarrow sc5$ for the scenes succeeding that could be inferred from the clip-scene organization;
- $c1 \Rightarrow c2$ for the clips succeeding that are defined by design;
- $sc2 \Rightarrow sc3$ which is inferred (inherited) by the *activate* relationship between the two clips;
- $p1 \rightleftharpoons p2$, $p2 \rightleftharpoons p3$, $p3 \rightleftharpoons p4$, $p4 \rightleftharpoons p5$ for the page termination that could be inferred by tracing the scene sequence, by observing that pages share the same window.

for a total of 16 relationships, 6 of which are defined by design, while the remaining 10 can be automatically inferred from the document structure and the relationships implications.

3.4 Nested continuous media objects

The user might need to switch temporarily to another multimedia document during the presentation. The contemporary presence of two (or more) continuous media presentations which are not coordinated (like a video with an associated audio channel) cannot be correctly described by the synchronization primitives so far introduced. Paying attention to two video or audio clips simultaneously is not possible, and resource sharing cannot simply solved by brute force replacement of channel usage. We need to set up a hierarchy between two continuous media, to define how they take precedence over each other.

Definition 4. Let A and B be two continuous media objects. We define "A has priority over B", written $A \succ B$, a relationship such that when object A becomes active while B is active too, B is moved to a suspended state, and playback of A begins. B waits up to deactivation of object A, then goes back to active state resuming its playback.

We note that the suspended state was not considered as a relevant state in our previous discussion. Indeed, this state does not cause any modification in the other media objects, that may however be influenced by the activation of the object with higher priority.
As an example, let us suppose that a page, say $p2$, contains a hyperlink to a multimedia clip $c3$. While $c1$ and $c2$ are the main thread of the presentation, $c3$ is a complementary section that the user may consult or not. The relationship $c3 \succ c1$ implies that if the user follows the hyperlink to $c3$, $c1$ must be suspended in order to allow $c3$ to get the user attention. If $c3$ contains scenes to which text pages are associated, these will replace $p2$ as usual (if assigned to the same channel). Figure 3 illustrates this case. The relationship $c3 \succ c1$ may also solve conflicts about the use of physical resources (like audio channels), but we do not enter into such details here.

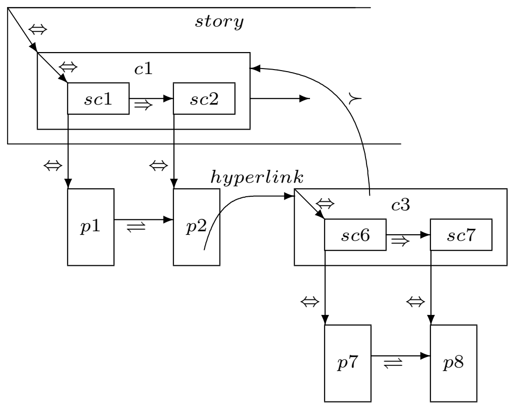

Figure 3: Access to nested multimedia

4. USER INTERACTION

We introduce a few issues about synchronization requirements due to user interaction, without entering into details that would require a much more thorough discussion, far beyond the size of this paper.
When a multimedia presentation starts, its behaviour in absence of user actions is defined by the synchronization primitives we have discussed, according to the document main structure. Any termination not caused by the natural ending of a continuous media can however bring the presentation into an incorrect state.
Let us suppose that the user stops a videoclip playback during a scene. Since the scene is interrupted, it does not activates the following scene. The last frame may still hold the clip window, but since the clip is no longer active it may be replaced by any new clip the user decides to start, e.g., by selecting another chapter or section from the document index. Since the page associated to the interrupted scene is still active, it also must be terminated in order to release the channel. Therefore, we need to define another synchronization primitive to correctly describe this case.

Definition 5. Let A and B be two generic media objects. We define "A is terminated with B", written $A \Downarrow B$, the relationship such that the forced ending of object A causes the forced termination of B object.

This relationship is authomatically inferred for all the media objects between which a relationship of the type $\Leftrightarrow$ exists, as illustrated in figure 4, i.e., $sc1 \Downarrow p1$, $sc2 \Downarrow p2$, $sc3 \Downarrow p3$, $sc4 \Downarrow p4$ and $sc5 \Downarrow p5$.
The case when the user follows a reference to a different document has already been discussed in previous section. We can now point out that, with reference to figure 4, $p6$ takes the place of $p3$ in the relationship $p4 \rightleftharpoons p3$, therefore the forced termination of $sc3$ must terminate $p6$ instead of $p3$; the relationship $sc2 \Downarrow p3$ becomes $sc2 \Downarrow p6$. This is a dynamically established relationship. In any subsequent access to the same document the second clip remains tied up to its first page and not to the external document.
Jumps forward or backward in a continuous media may be managed in the same way: a jump is a stop at the actual playback point, and a start at the destination point. Entering into details is however out of the size of this paper.

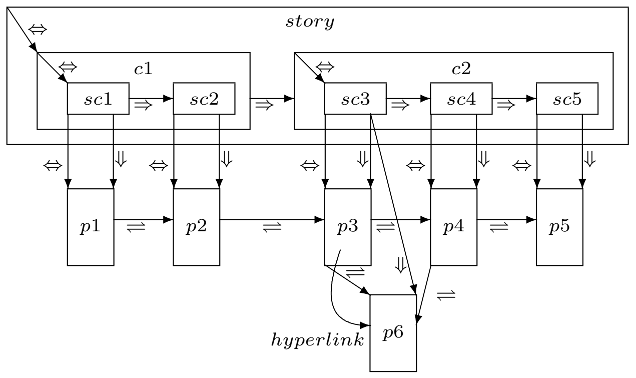

Figure 4: The complete synchronization scheme

5. RELATED WORK

Media synchronization in hypermedia documents has been largely investigated. The first serious attempt to introduce multimedia synchronization in hypertext document is the Amsterdam Hypermedia Model [1; 2], that distinguishes between atomic and composite components. Synchronization inside composite components is made by synchronization arches that establish time relationships between two components or two anchors.

In [4] Standard Reference Model for Intelliget Multimedia Presentation Systems is presented. It describes an implementation of processes required for the generation of multimedia presentations, dividing them into well-defined layers (control layer, content layer, design layer and realization layer). Each layer has a precise role, but the SRM lacks of a specific document model. In [3] we can find a first attempt to integrate the AHM with the SRM leading to a more precise model without losing expressiveness. [3] describes in details, using AHM as a basis, the stages of generating a hypermedia presentation.

Spatio-temporal relationships of a multimedia presentation are discussed in [5]. The paper defines a model based on temporal and spatial relationships between different multimedia objects that compose a presentation. Temporal and spatial start points are defined, and the objects are set using topological and time relationships between them. The result is a formalism suitable for designing any multimedia composition.

The work [6] proposes a multimedia document model containing temporal information that can be described using traditional document structures (e.g., SGML). The model contains the logical structure, the layout structure, and the temporal structure to formally describe multimedia objects. It pays more attention to user interactions; the authors propose an interactive synchronization based on use of tokens for the control scheme to solve problems due to resynchronization after any user interactions.

PREMO[7], Presentation Environment for Multimedia Objects, primarily focuses on the presentation aspects of multimedia. PREMO aims at describing a standard programming environment in a very general way, through which media presentation can be constructed, processed and rendered as part of an overall application. The model can be divided into four components: each component is described in an object-oriented way. The model supplies an abstraction for the various multimedia virtual devices that may produce different multimedia data of a presentation. An object type and its subtypes set up time relationships. Two objects can be rendered sequentially, in a parallel way or in alternative according to presentation state.

The work [8] presents a scheme for intra- and inter-stream synchronization of stored multimedia. It proposes a new protocol to ensure continuous and synchronous delivery of distributed stored multimedia objects across a network community. Three models are described that assure synchronization even in environment with different delays, jitters, sever drop-outs, clock drifts and alternation of the average delay using a resynchronization protocol based on buffer level control. Unlike many other works, the models here defined does not require synchronized clocks on the network.

In [9] a model for synchronization of a hypermedia application is described. It is based on hypercharts, a notation that extends the statechart formalism in order to make it able to describe temporal constrains of a multimedia presentation.

Two other works discuss issues very close to the one approached by our model. In [10] a model for flexible presentations is defined, called FLIPS (for FLexible Interactive Presentation Synchronization). Media objects have no predefined time length, but it can change according to user interactions. The synchronization scheme is based on two temporal relationships, called *barriers* and *enablers*. If event A is a barrier to event B then B is delayed until event A occurs. If event A is an enabler for event B when event A occurs event B also occurs as soon as it is barrier-free. FLIPS defines five object states: idle, ready, in-process, finished and complete.

The main differences with our model concern the system environment and the hypermedia dynamics modeling. FLIPS is not oriented to network based applications, and appears more oriented to multimedia presentations, rather than to hypermedia navigation. No structure is in fact provided, other than the ones coming from the objects mutual interrelationships. Synchronization is defined between object's states and not between the objects themselves. Using barriers and enables, the start or end of an object cannot directly cause the start or end of another, but could only change the state of the object at hand. For example, its beginning (activation in our model) depends from the absence of bar-

riers that can be set or reset by complex conditions. In our model the start or end of an object could be caused only by the user or by events associated to another object, independently from the state of other presentation's components.
SMIL[11], Synchronized Multimedia Integration Language, is a W3C recommendation defined as an XML application. It is a very simple markup language that defines tags for presenting multimedia objects in coordinated way. Synchronization is achieved through two tags: `seq` to render two or more objects one after the other and `par` to reproduce them at the same time. Using attributes it is possible to play only segments inside the time span of an object. The screen is divided into regions inside which it is possible to put multimedia objects that form a presentation. Differently from our model, however, SMIL does not consider user interactions. Moreover, it does not define a reference model for the data structure, but only defines tags for describing media object presentation.

6. CONCLUSION

In this paper we have presented a model for describing the synchronization between several media delivered over a network in a Web-based environment. Synchronization concerns the download and the activation of the several media files according to the structure of the whole hypermedia document, the playback status of the media objects, and the user interaction. We have introduced five synchronization primitives some of which must be defined at hypermedia design phase while others, describing the playback dynamics, can be deducted from the document structure and the media-related events.
The model is oriented to describe applications based on a main video or audio narration to which static or dynamic documents are attached. The model is suited for applications like distance education, Web advertising and selling, and news-on-demand.
We have verified our model's potentialities using commercial streaming plug-ins for WWW browsers, and servers for delivery of continuous media over a network. Early experience has shown that our model provides a much more sophisticated (even if simple) scheme than provided by available software (e.g., RealMedia servers and players). Future work will be oriented mainly to investigate implementation issues.

7. ACKNOWLEDGMENT

The structure and the name of "video-centered" hypermedia has been suggested by Alberto Colorni, Technical University of Milan.

8. REFERENCES

[1] L. Hardman, D.C.A. Bulterman and G. van Rossum. The Amsterdam Hypermedia Model: adding time and context to the Dexter Model. *Comm. of the ACM*, 37(2), pages 50–62, 1994.

[2] L. Hardman. Using the Amsterdam Hypermedia Model for Abstracting Presentation Behavior. In *Electronic Proceedings of the ACM Workshop on Effective Abstractions in Multimedia*. San Francisco, CA, 4 November 1995.

[3] L. Hardman, M. Warring and D.C.A. Bulterman. Integrating the Amsterdam Hypermedia Model with the Standard Reference Model for Intelligent Presentation Systems. *Computer Standards and Intefaces*, 1998. `http://www.cwi.nl/ftp/mmpapers/AHM-SRM.ps.gz`

[4] M. Bordegoni, G.Faconti, M.T. Maybury, T. Rist, S.Ruggeri, P.Trahanias and M. Wilson. A Standard Reference Model for Intelligent Multimedia Presentation Systems. *Computer Standards and Intefaces*, 1998. `http://www.cwi.nl/ftp/mmpapers/AHM-SRM.ps.gz`

[5] M. Vazirgiannis, Y. Theodoridis and T. Selling. Spatio-temporal composition and indexing for large multimedia applications. *Multimedia Systems*, 6(4), pages 284–298, 1998

[6] Chung-Ming Haung, Jyh-Shiou Chen, Chih-Hao Lin and Chian Wang. MING I: a distributed intractive multimedia document development mechanism. *Multimedia Systems*, 6(5), pages 316–333, 1998

[7] D.J. Duke, I.Herman, T.Rist and M.Wilson. Relating the primitive hierarchy of PREMO standard to the standard reference model for intelligent multimedia presentation systems. Information Systems (INS) ISN-R9704, 1997. `http://www.cwi.nl/static/publications/reports/abs/INS-R9704.html`

[8] A. Biersack and W. Geyer. Synchronized delivery and playout of distributed stored multimedia streams. *Multimedia systems*, 7(1), pages 70–90, 1999

[9] F.B. Paulo, P.C. Masiero, M.C. Ferreira de Oliveira. Hypercharts: Extended Statecharts to Support Hypermedia Specification. In *IEEE Transactions on Software Engineering*, 25(1), pages 33–49, 1999.

[10] J. Schnepf, J.A. Konstan and D. Du. Doing FLIPS: Flexible Interactive Presentation Synchronization. Distributed Multimedia Center, Department of Computer Science, University of Minnesota. `ftp://ftp.cs.umn.edu/users/du/papers/flips.ps`

[11] Synchronized Multimedia Working Group of W3C. Synchronized Multimedia Integration Language (SMIL) 1.0 Specification. W3C Recommendation, 15 June 1998. `http://www.w3.org/TR/REC-smil`