# Retrieving Consistent Multimedia Presentation Fragments

**A. Celentano, O. Gaggi**
Università Ca' Foscari di Venezia
(auce,ogaggi)@dsi.unive.it

**M.L. Sapino**
Università di Torino
mlsapino@di.unito.it

## Abstract

*In this paper we discuss some issues about information retrieval in multimedia presentations. We introduce a class of multimedia presentations made of independent and synchronized media, and discuss retrieval requirements of presentation fragments. Then we discuss a retrieval model able to reconstruct the fragments of a presentation from the atomic components returned by the execution of queries to multimedia presentation repositories. The retrieval model is based on an automaton which formally describes the presentation states entered by the events which trigger media playback. Retrieving a consistent fragment corresponds to building a new presentation with all the media related to the retrieved ones, with their original structural and synchronization relationships.*

## 1 Introduction

In this paper we discuss information retrieval in multimedia presentations. We introduce a class of multimedia presentations made of independent and synchronized media, and discuss retrieval requirements of presentation fragments. Then we discuss a retrieval model able to reconstruct the fragments of a presentation from the atomic components returned by the execution of queries to multimedia data repositories.

A multimedia presentation can be modelled as the composition of one or more continuous media file, such as video or audio streams, which are presented to a user together with static documents, such as images or text pages, which are displayed in synchrony with them. From the user point of view the different documents constitute a whole presentation which is ruled by the synchronization relationships among the component media items.

News-oriented applications such as news-on-demand and Web advertising, and virtual museums are good representative of such presentations. News are presented through audio or a video clips, and images and texts are displayed according to the particular subject the speaker is talking about. Artworks in a virtual museum are illustrated by multimedia presentations made of text, images and videos, possibly commented by voice narrations and music.

In both cases the whole presentation is a collection of sections each devoted to a single article or to a single artwork according to a recurring schema. Sections can be grouped in larger entities (e.g., museum rooms) giving the whole presentation a structure which helps the user to navigate its contents.

Information retrieval on multimedia presentations can be defined as the task of retrieving items of different media type from a repository of multimedia presentations, satisfying some query parameters, and presenting the user all the fragments of the presentations in which the retrieved items were originally located. As such, this task has a number of peculiarities.

The user queries a repository of presentations, expecting (fragments of) presentations as results, but the query processor plausibly retrieves the component media items, which must be integrated with other presentation components in order to build a meaningful answer.

More specifically, coherent and understandable segments of the original presentations must be returned to the user. The query result extent must be identified according to a context which takes into account the presentation structure and dynamics.

We do not address here issues related to query processing and retrieval of multimedia data. We simply assume that the user formulates a query with proper parameters which identify media items whose content is relevant, belonging to one or more presentations. The retrieval system executes the query and returns a (ranked) list of media items, possibly of different media types. Each returned item is a triple $\langle Mid, MT, P \rangle$ where $Mid$ is the media item identifier (e.g., a file locator or a URL), $MT$ is a media type and $P$ is a reference to the presentation in which it is located. If a media item belongs to several presentations, a set of triples is returned with different values of $P$. Our goal is to re-construct, for each triple, a coherent (and complete) fragment of the presentation $P$ to be displayed as result of the query, identifying all objects which belong to the same fragment of P which contains the media referenced by $Mid$. Therefore we have:

- to identify the missing media items to be supplied, and
- to identify the fragment scope.

We assume also that each presentation is described by a schema according to a model able to reveal the relationships among the media components, i.e., the static organization of a presentation, which relates media together, and the temporal aspects of its behavior, which describe how the different segments evolve. We'll discuss in Section 3 such a model and in Section 4 a suitable schema.

## 2 Related work

While multimedia information retrieval is an issue largely investigated in literature, only few works focus on the issues of querying and browsing excerpts of multimedia presentations.

In [1] Adali et al. present an algebra for creating and querying interactive multimedia presentation databases. The authors model a hypermedia presentation as a tree whose branches represent different playouts due to user interactions and nodes contain the set of active objects and the spatio-temporal constraints between them. The algebra defines operators to select and present paths of the presentation the user is interested to; this allows users to create new presentations merging parts of existing ones.

In [10] the authors propose an integrated query and navigation model which provides facilities for query writing activities to users that have little familiarity with databases contents and schemas. Differently from the approach we propose, the authors are not interested in modelling the temporal behavior of the retrieved results, but only in hiding problems relating querying of heterogeneous data in a distributed environment. They provide an interactive user interface which permits both the use of conventional declarative queries and navigational techniques.

The same approach is used by Chiaramella in [5]. He presents a model which fully integrates browsing and querying of multimedia data capabilities, giving particular emphasis to structured information, but problems related to temporal synchronization in composite documents are not discussed. The model contains two components: the hypermedia component and the information retrieval (IR) component. The hypermedia component allows the user to formulate queries (and browse the results) which explicitly reference both *content knowledge*, i.e. the semantic content of atomic data, and *structural knowledge*, i.e. the types of documents and cross-reference links between documents. The IR component contains information about the resolution of the queries.

In [8, 9] the authors present GVISUAL, a graphical query language to formulate query on multimedia presentations based on content information. Multimedia presentations are modelled as directed acyclic graphs. A query consists of a head with its name and parameters, and of a body. The query body contains an iconized representation of the objects involved and a *conditions box* with the conditions required. An objects icon contains a name and can be nested to represent a composition relationship. Media items can be connected by edges which represent temporal operators. The authors aim at querying not only the information stored in individual streams but also the *flow of information* which represents the theme of a multimedia presentation.

Other works are more oriented to building new presentations out of existing multimedia material, thus covering only partially the issues related to the retrieval of presentations. Delaunay$^{MM}$[6] is a visual tool for querying and presenting multimedia data stored in distributed data repositories, including the Web. It allows the user to define the layout of a virtual document, by arranging graphical icons inside style sheets, and document's content, through ad hoc queries to multiple repositories. Also in this proposed paper, the authors did not address any solution for the specification of temporal synchronization among the objects.

In [2, 3] an extension to SQL language, named SQL+D, is presented, which allows users to include spatial and temporal specifications to an SQL query. The user can select data needed and specify how it should be displayed. The clause DISPLAY-WITH describes the screen areas, called *panels* in which the retrieved items are placed, and the clauses AUTOSKIP and SHOW define respectively the duration of the media items playback and the temporal constraints among the objects of a presentation.

## 3 Synchronization Description in Hypermedia Presentations

For modelling multimedia presentations we refer to a synchronization model we have defined in [4, 7]. The model is oriented to web-based hypermedia presentations, and describes the hierarchical structure of a presentation and the temporal behavior of its components.

A hypermedia document is a collection of *modules*. A module is a container of different kinds of continuous and static media objects. Static objects like texts or images are referred to as *pages*. Continuous media objects are hierarchically organized: a video or an audio stream is divided in *stories* which are made by *clips*, which are media files played continuously. A clip can be divided in *scenes* which are temporal intervals inside the time span of an object. The scenes are the media units to which synchronization events are associated. As a clip plays, the scenes are played in sequence, thus generating events (i.e. the beginning and the termination of media) which cause other media to be displayed or played, or terminated.

Each medium requires some device to be rendered or played. A media object can use a new browser's window, a frame inside the window, an audio channel, or a combination of audio and video resources (as required by a video file with integrated audio). The model calls *channel* such a virtual device, which is used by the medium for all the duration of its playback. A channel is *busy* if an active object is using it, otherwise it is *free*. Free channels may hold some content, for example the last frame of a video clip or a page that has not been replaced yet by a new one.

Synchronization between different components of a hypermedia document is described by five synchronization primitives, which define object reactions to some events. Two relations describe the sequential and the parallel composition of two media; the others model objects reactions to user's interactions.

The relationship *"A plays with B"*, written $A \Leftrightarrow B$ models the parallel composition of media $A$ and $B$: it 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 relationship *"A activates B"*, written $A \Rightarrow B$ models the sequential composition of two objects: when object $A$ naturally ends, object $B$ begins its playback. These two relationships are similar to the tags `<par>` and `<seq>` of SMIL [11] but some differences exist that are detailed in [7]. We distinguish between *internal* events which are generated by some components of the presentation and *external* events, which are generated by the user. In particular we distinguish between the *natural termination* of an object, i.e. when it reaches its ending point, and the *forced end*, i.e. when the user stops it. In the relationship $A \Rightarrow B$, if the user stops object $A$, object $B$ is not activated and in the same situation, if the relationship $A \Leftrightarrow B$ holds, the object $B$ is not terminated.

The relationship "*A is replaced by B*", denoted by $A \rightleftharpoons B$, is mainly used with static objects whose duration is potentially infinite. It states that, when object $B$ starts it forces $A$ to end, so its channel is released and can be used by $B$.

Two other relationships model the objects reactions to user interactions. We cite them here for completeness but they have a little role in the context of multimedia presentation retrieval. The relation "*A is terminated with B*", written $A \Downarrow B$, terminates two objects at the same time as a consequence of the forced termination of object $A$ by the user or some other external event. The relationship "*A has priority over B with behavior $\alpha$*", symbolically written $A \overset{\alpha}{>} B$, 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 (the object $B$). When object $B$ comes to the end, object $A$ is resumed, if it was paused. The reader is referred to [4, 7] for a discussion of details and motivations about this synchronization model.

We introduce a working example in order to show the model application to a web museum domain. In a collection of multimedia presentations each presentation is a guided tour in a virtual reconstruction of a real museum. Every presentation is organized in modules, one for each museum section. An animation simulates the user walking through the rooms. As the user moves to a new museum section, a different soundtrack is played, to emphasize the change of context. Each time the animation focuses on an artwork, a text page with detailed information about the artwork is displayed. Several examples of presentations associated to virtual museums can be found on the Web. The reader is referred to the Web site of Palazzo Grassi, a cultural institution in Venice featuring several interesting virtual exhibitions at URL http://www.palazzograssi.it. The exhibitions are based on guided and free tours in a 3D scenario, and contain, besides the tour animations, text information and rich audio support. Some of them have been used to validate the module.

A typical presentation requires three channels: two regions of the user screen, *an* for the animations and *info* for the text pages, and an audio channel *sound*, for the soundtracks. The presentation is organized in modules which correspond to the sections of the museum. Each module plays a different soundtrack which helps the user to locate where he/she is.

The animation which moves inside a section is a story, divided in clips (which correspond to files), one for each artwork. Each clip is divided in two scenes: the first simulates the movement of the user from the previous position to the next point of view and the second focuses the user attention on the artwork. Figure 1 shows the hierarchical structure of a section of the museum (i.e., a module) and the relationships needed to synchronize the media components involved.

As the user enters a section (e.g., module $M_1$), the soundtrack and the first story begin their playback; the relations $M_1 \Leftrightarrow sound$ and $M_1 \Leftrightarrow story$ model this behavior. The relations $story \Leftrightarrow c_1$ and $c_1 \Leftrightarrow sc_{1,1}$ start the first animation (i.e., the first scene of the first clip). The soundtrack is a short audio file which is played continuously ($sound \Rightarrow sound$). At the end of the first scene the relation $sc_{1,1} \Rightarrow sc_{1,2}$ activates scene $sc_{1,2}$ which, as stated by relation $sc_{1,2} \Leftrightarrow p_1$, displays a text page with some information

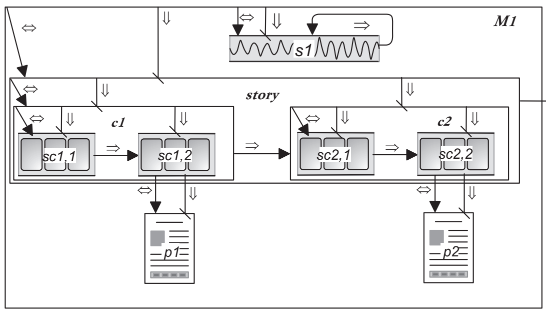

**Figure 1. Structure of a tour in a virtual museum**

about the current artwork. Other clips are modelled in the same way:

- $c_i \Leftrightarrow sc_{i,1}$ starts the first scene of clip $c_i$,
- $sc_{i,1} \Rightarrow sc_{i,2}$ plays the scenes of a clip sequentially,
- $sc_{i,2} \Leftrightarrow p_i$ displays the text page associated to the artwork.

When clip $c_i$ ends playing (without being interrupted by the user, otherwise the presentation will behave differently), a relation $c_i \Rightarrow c_{i+1}$ starts the next clip.

At the end of each museum's section, the virtual tour steps to the next one. In terms of our model, this means that the modules are to be played sequentially. Each module ends when its last story ends. At the end of module $M_1$, as depicted in Figure 1, the relation *story* $\Rightarrow M_2$ activates module $M_2$, which takes the channels used by previous module ($M_1 \rightleftharpoons M_2$).

Figure 1 also contains other relationships which model the forced stop of the presentation when the user stops the module active in that moment. The forced stop is propagated from the module to the story and to the soundtrack by the relationships $M_1 \Downarrow story$ and $M_1 \Downarrow sound$, and from story to clips (*story* $\Downarrow c_i$) and scenes ($c_i \Downarrow sc_{i,j}$ where j = 1, 2) through the hierarchical structure. Each scene stops the associated page, if it exists, ($sc_{i,2} \Downarrow p_i$) thus freeing all the channels of the presentation.

## 4 A Formal Approach to Multimedia Presentation Modelling

In previous section we have introduced the relevant component of our model. In particular, we have shown that the playout of a hypermedia presentation can be easily described in terms of

- a set $\mathcal{MI}$ of media items (both continuous and non continuous);
- a set $\mathcal{CH}$ of delivery channels;
- a set $\mathcal{E} = \{start(\_), end(\_)\}$ of possible events
- a set $\mathcal{SP}$ of synchronization primitives.

The above elements provide an intensional representation of the evolution of the presentation along time. We shall refer to a generic presentation $P$ as $P(\mathcal{MI}, \mathcal{CH}, \mathcal{E}, \mathcal{SP})$, to point out what the structural components of the presentation are.

At any time instant, the hypermedia presentation is completely described by the set of media that are active at that time, and the corresponding channel occupation. This relevant information is captured by the following notion of *state* of the presentation.

**Definition 4.1 (State)** The state of a hypermedia presentation is a pair $\langle MA, ch \rangle$, where $MA$ is the set of media that are active, and $ch : \mathcal{CH} \rightarrow \mathcal{MI} \cup \{\_\}$ is a function, that returns, for every channel, the media item that occupies it. The underscore symbol _ denotes the absence of media item; it is the value used to identify empty channels.

**Remark 4.1** The set § of the possible states for a hypermedia presentation is finite, since both the set of media items and the set of channels are finite.

Before the presentation starts, no media item is active, thus all channels are free. When an event occurs, the overall status of the presentation changes: some items that were not active become active, some active items end, other items could be forced to stop because of some interruption.

If we observe the system along time, the only relevant time instants are the *observable time instants*, that is, the time instants in which an event occurs. Indeed, these are the time instants in which something in the state of the presentation might change, as a consequence of the occurred event.

The state of a hypermedia presentation is thus a function of time. More specifically, it is a function of *observable time instants*. We assume that at any observable time instant exactly one "master event" occurs, that is, either an item is activated, or an item naturally ends[1]. It might be the case, anyway, that at the same time instant other items are activated/stopped, according to the synchronization rules characterizing the presentation.

It is important to notice that, given the set of synchronization primitives associated to the presentation, the effects of any observable event are deterministically implied by (i) the set of currently active media, (ii) the current channel occupation, and (iii) the occurred event. Thus, all the possible

[1]We don't consider users' interruptions, since in this paper we are interested in extracting portions of presentations on the base of their natural evolution.

evolutions along time of a presentation $P$ can be described by an automaton, defined as follows.

**Definition 4.2** Let $P(\mathcal{MI},\mathcal{CH},\mathcal{E},\mathcal{SP})$ be a presentation. Its corresponding automaton is the 5–*tuple* $AUT(P) = \langle \S, \mathcal{E}, s_o, next, Final\rangle$, where

- $\S$ is the set of possible states for the presentation $P$;
- $\mathcal{E}$ is the set of possible events of the form *start(a), end(a)*, being $a$ any media item in $\mathcal{MI}$;
- $s_0$, the initial state, is $\langle MA_0, ch_0\rangle$, where $MA_0 = \emptyset$, and $ch_0(c) = \_$, for all $c \in \mathcal{CH}$;
- the transition function $next : \S \times \mathcal{E} \rightarrow S$ is the mapping that (deterministically) associates any state $s$ to the state $s'$ in which $s$ is transformed by an event $e$;
- $Final$, the set of accepting states, is the empty set[2].

While it is clear, from the above definition, what $\S$, $\mathcal{E}$, $s_o$ and $Final$ are, we still have to define formally the deterministic behavior of the function $next$. This definition requires some extra notions, that we introduce in the following.

First, we need to be able to capture all the consequences that an event might have on the presentation, given the current state, according to the synchronization rules. Starting and ending events might indeed activate a cascade of simultaneous media activations or stops. The following notions of *closures of an item*, with respect to some category of rules, capture these effects.

**Definition 4.3** **($(\Leftrightarrow)Closure$)** Let $A$ be any data item in $\mathcal{MI}$. The $(\Leftrightarrow)Closure$ of $A$ is the set inductively defined as follows:

- $A \in (\Leftrightarrow)Closure(A)$;
- for any item $B$, if $B \in (\Leftrightarrow)Closure(A)$ and the synchronization rule $B \Leftrightarrow C$ exists, then $C \in (\Leftrightarrow)Closure(A)$;

$(\Leftrightarrow)Closure(A)$ captures the non symmetry and the transitivity of the *plays with* relation. In particular, we will use the set $(\Leftrightarrow)Closure(A)$ to take care of the (non symmetric) consequences of the end of $A$, in the presence of *plays with* rules.

**Definition 4.4** **($(\Leftrightarrow)1step$)** Let $A$ be any data item in $\mathcal{MI}$. The $(\Leftrightarrow)1step$ of $A$ is the set defined as follows:

- for any item $B$, if the synchronization rule $A \Leftrightarrow B$ exists, then $B \in (\Leftrightarrow)1step(A)$;
- for any item $B$, if the synchronization rule $B \Leftrightarrow A$ exists, then $B \in (\Leftrightarrow)1step(A)$;

$(\Leftrightarrow)1step(A)$ captures the symmetry of the *plays with* relation. In particular, we will use the set $(\Leftrightarrow)1step(A)$ to take care of the (symmetric) consequences of the start of $A$, in the presence of *plays with* rules.

**Definition 4.5** **($(\Downarrow)Closure$)** Let $A$ be a data item in $\mathcal{MI}$. The $(\Downarrow)Closure$ of $A$ is the set inductively defined as follows:

- $A \in (\Downarrow)Closure(A)$;
- for any item $B$, if $B \in (\Downarrow)Closure(A)$ and the synchronization rule $B \Downarrow C$ exists, then $C \in (\Downarrow)Closure(A)$;

Intuitively $(\Downarrow)Closure(A)$ contains all the media items that according to the *terminated with* relation, are required to stop if $A$ terminates.

The following predicate *ComponentOf(_,_)* is introduced to take care of the hierarchical structure of the media items.

**Definition 4.6** **($ComponentOf(A,B)$)** Let $A$ and $B$ be a data item in $\mathcal{MI}$. $ComponentOf(A,B)$ evaluates to true if and only if at least one of the following conditions holds:

- for any items $A$ and $B$, if $A$ is a story (clip) and $B$ is a clip (resp. scene) and the synchronization rule $A \Leftrightarrow B$ exists;
- for any items $A$ and $B$, if $A$ is a story (clip) and $B$ is a clip (scene) and a clip (scene) $C$ exists such that $A \Leftrightarrow C$ and $C \Rightarrow \ldots \Rightarrow B$.

The state transformation caused by an event might be a complex action. For the sake of readability, we define some parameterized *macro actions* that group together semantically related operations caused by the occurrence of an event, to apply the synchronization rules. Basically, any macro action takes care of the effects of the start, the stop, or the replacement of a media item, both on the set of active media, and on the function modelling the occupation of channels.

**START (X: media item; $\Delta$ :set of media items; ch: channel association function)**

```
begin
    $\Delta = \Delta \cup \{X\}$;
    $ch(channel(X)) = X$;
end
```

[2]The choice of letting $Final$ be the empty set is due to the fact that we are not interested in recognizing acceptable streams of events. Instead, our investigations will be based on the information content of the states. If we also wanted to recognize the sequence of events that naturally lead from the starting point of the presentation to its end, that is, to the point where no media item is active, and all the channels are free, we would let $Final = \{s_0\}$

```
REPLACE (X, Y: media item; $\Delta_{add}$, $\Delta_{del}$ :set of media items; ch: channel association function)
  begin
    $\Delta_{del} = \Delta_{del} \cup \{Y\}$;
    $\Delta_{add} = \Delta_{add} \cup \{X\}$;
    $ch(channel(Y)) = X$;
  end
```

```
STOP (X:media item; $\Delta$ :set of media items; ch: channel association function)
  begin
    $\Delta = \Delta \cup \{X\}$;
    $ch(channel(X)) = \_$;
  end
```

```
ACTIVATE (A: media object; $\Delta^+$, $\Delta^-$: set of media items; oldch, newch: channel association function)
  begin
  if $A \in MA_n$ then don't care [3]
  else
    $Set = \{A\}$;
    while $Set \neq \emptyset$ do
      begin
      pick any $B$ from $Set$
      if $oldch(channel(B)) = \_$
        then START(B, $\Delta^+$, $newch$);
      else
        if $oldch(channel(B)) = C$
          and there exists the rule $C \rightleftharpoons B$
        then
          begin
          REPLACE(B, C, $\Delta^+$, $newch$);
          for all items $D \in (\Downarrow)Closure(C)$
            if $D \in MA_n$ then STOP(D, $\Delta^-$, $newch$);
          end
        if $oldch(channel(B)) = C$
          and there exists $B \overset{s}{>} C$,
        then
          begin
          REPLACE(B, C, $\Delta^+$, $\Delta^-$, $newch$);
          for all items $D \in (\Downarrow)Closure(C)$
            if $D \in MA_n$ then STOP(D,$\Delta^-$, $newch$);
          end
        if $oldch(channel(B)) = C$
          and $ComponentOf(B, C)$
        then
          START(B, $\Delta^+$, $newch$);
        if $oldch(channel(B)) = C$
          and $ComponentOf(C, B)$
        then
          $\Delta = \Delta^+ \cup \{B\}$;
      if $B \in \Delta^+$ then
        $Set = Set \cup ((\Leftrightarrow)1step(B) \setminus \Delta^+)$;
      end
  end
```

[3] $A$ is already active, thus the event is ignored.

The state transition function is defined as follows.

**Definition 4.7 (State transition function)** The state transition function $next : \S \times \mathcal{E} \to \S$, where $\S$ is the set of all possible states, and $\mathcal{E}$ is the set of events, is the function that, given a state $s$ and an event $e$, returns the state $s' = next(s, e)$ iff $s = \langle MA_n, ch_n \rangle$, $s' = \langle MA_{n+1}, ch_{n+1} \rangle$, $MA_{n+1} = MA_n \setminus \Delta^- \cup \Delta^+$, and $\Delta^-$, $\Delta^+$ and $ch_{n+1}$ are defined according to the following iterative process, depending on the event $e$.

```
1. [e = start(A)]
   Let $\Delta^- = \emptyset$; $\Delta^+ = \emptyset$; $ch_{n+1}(c) = ch_n(c)$, for all $c \in \mathcal{CH}$;
   ACTIVATE (A, $\Delta^+$, $\Delta^-$, $ch_n$, $ch_{n+1}$);

2. [e = end(A)]
   Let $\Delta^- = \emptyset$; $\Delta^+ = \emptyset$; $ch_{n+1}(c) = ch_n(c)$, for all $c \in \mathcal{CH}$;
   if $A \notin MA_n$ then don't care [4]
   else
     begin
     STOP( A, $\Delta^-$, $ch_{n+1}$);
     for all items $B \in (\Leftrightarrow)Closure(A)$
       if $(B \in MA_n)$ then
         begin
         STOP(B, $\Delta^-$, $ch_{n+1}$);
         for all items $C \in (\Downarrow)Closure(B)$
           if $C \in MA_n$ then STOP(C, $\Delta^-$, $ch_{n+1}$);
         end
     for all items $B$ such that there exists $A \Rightarrow B$ do
       ACTIVATE (B, $\Delta^+$, $\Delta^-$,$ch_{n+1}$);
     end
```

To make the definition of the automaton clearer, let's apply it to the example introduced in Section 3. The automaton is in Figure 2. The set of media items for the first module is $\mathcal{MI} = \{M_1, sound, story, c_1, c2, sc_{1,1}, sc_{1,2}, sc_{2,1}, sc_{2,2}, p_1, p_2\}$ and the initial state is $s_0 = \langle MA_0, ch_0 \rangle$ where $MA_0 = \emptyset$ and $ch_0(c) = \_$, for all $c \in \mathcal{CH}$.

The activation of the presentation is given by the event $e = start(M_1)$, that starts the first module.

[4] There is no way to end a non active item.

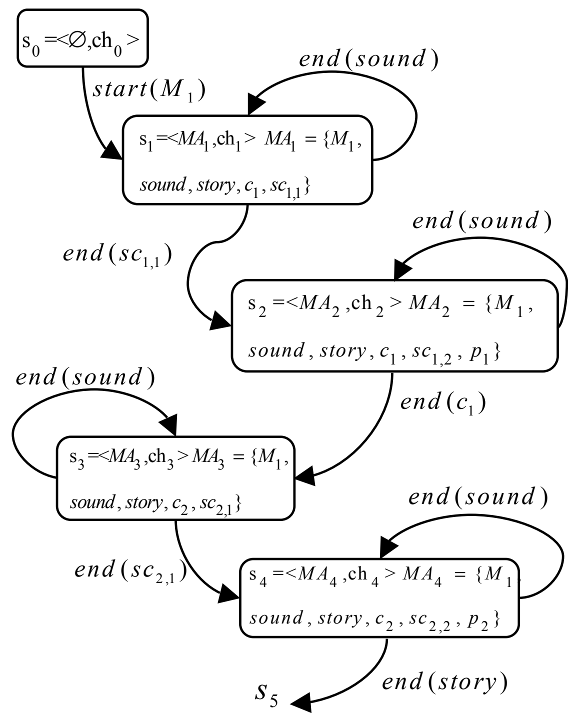
s0 =<∅,ch0 >
start(M1)
end(sound)
s1=<MA1,ch1> MA1 = {M1, sound, story, c1, sc1,1}
end(sc1,1)
end(sound)
s2 =<MA2 ,ch 2 > MA2 = {M1 , sound , story, c1 , sc1,2 , p1 }
end(sound)
end(c1)
s3=<MA3,ch3 > MA3 = {M1, sound, story, c2 , sc2,1 }
end(sound)
end(sc2,1)
s4 =<MA4 ,ch 4 > MA4 = {M1 sound , story, c2 , sc2,2 , p2 }
end(story)
s5

**Figure 2. The automaton for the first module of the presentation**

The function $next(s_0, start(M_1))$ returns the state $s_1 = \langle MA_1, ch_1\rangle$ where $MA_1 = \{M_1, sound, story, c_1, sc_{1,1}\}$ and $ch_1(channel(M_1)) = M_1$, $ch_1(sound) = sound$, $ch_1(an) = sc_{1,1}$ and $ch_1(info) = \_$. Thus, the transition from $s_0$ to $s_1$ captures the fact that the first module activates the sound track and the first animation.

Without user interaction, the set of possible events is $\{end(sound), end(sc_{1,1})\}$. In the first case the state does not change, since the relationship $sound \Rightarrow sound$ plays the sound track continuously. Thus, $next(s_1, end(sound)) = s_1$.

In the case of the natural termination of the first scene ($e = end(sc_{1,1})$) the second scene starts, together with its associated page. The automaton reaches a new state $s_2 = \langle MA_2, ch_2\rangle$ where $MA_2 = \{M_1, sound, story, c_1, sc_{1,2}, p_1\}$ and $ch_2(an) = sc_{1,2}$, $ch_2(info) = p_1$ and $ch_2(c) = ch_1(c)$ for all $c \in \mathcal{CH} \setminus \{an, info\}$.

Once more, if the sound track naturally ends, the state does not change. When the second scene naturally ends, the clip reaches its ending point, thus event $e = end(c_1)$ occurs. The automaton reaches a new state $s_3 = \langle MA_3, ch_3\rangle$ where $MA_3 = \{M_1, sound, story, c_2, sc_{2,1}\}$ and $ch_3(an) = sc_{2,1}$, $ch_2(info) = \_$ and $ch_3(c) = ch_2(c)$ for all $c \in \mathcal{CH} \setminus \{an, info\}$.

As for the first clip, at the end of scene $sc_{2,1}$, scene $sc_{2,2}$ displays the associated text page, as shown in state $s_4 = \langle MA_4, ch_4\rangle$ where $MA_4 = \{M_1, sound, story, c_2, sc_{2,2}, p_2\}$ and $ch_4(an) = sc_{2,2}$, $ch_4(info) = p_2$ and $ch_4(c) = ch_3(c)$ for all $c \in \mathcal{CH} \setminus \{an, info\}$.

At the end of scene $sc_{2,2}$, clip $c_2$ ends, together with the $story$ in which it is contained. Since $(\Leftrightarrow)Closure(sc_{2,2}) = \{sc_{2,2}, p_2\}$ channels $an$ and $info$ are released. By the relations $story \Rightarrow M_2$ the second module is activated, which replaces $M_1$ ($M_1 \rightleftharpoons M_2$). Since $(\Downarrow)Closure(M_1)$ contains all the active media, all channels are released to be used by the media components of $M_2$.

## 5 Retrieving Consistent Presentation Fragments

The automaton, besides describing formally how the presentation evolves, contains in each node the information about which media play together, under all possible conditions about triggering of events related to normal play and user interaction. It is therefore the candidate source of information for re-constructing consistent presentation fragments after some media items have been retrieved according to user queries.

Recalling what we said in Section 1, the retrieval engine is assumed to return a set $\mathcal{R}$ of triples $\langle Mid, MT, P\rangle$ each triple denoting a media item $Mid$ with its type $MT$ and a reference to the presentation $P$.

The set of consistent presentation fragments containing the retrieved media items is built according to the following procedure.

1. For each item $r \in \mathcal{R}$ returned let $Mid_r$ be the media item identifier and $S_r$ the set of states in which $Mid_r$ is active. Each state $s_i \in S_r$ identifies the set of media items $MA_i$ which play in parallel, together with the proper channel assignments.

   Let $S_c$, be the set of states for which a fragment must be returned to the user as the answer to his/her query. Initially, $S_c = S_r$. If $Mid_r$ is active in two states $s_1$ and $s_2 = next(s_1, e)$ where $e$ is any event we consider only state $s_1$, therefore, $S_c = S_c \setminus \{s_2\}$. If $Mid_r$ is active in two states which are not sequential, and inactive in the middle, we take both states for subsequent analysis.

2. For each state $s_i \in S_c$ identify the event which activates the retrieved media. Since media are activated by *plays with*($\Leftrightarrow$) or *activate*($\Rightarrow$) relations, all states are entered by *end* events, but for state $s_1$ which is entered by event $start(M_1)$. In terms of the presentation synchronization relationships state $s_i$ is entered under one of two conditions:

(a) a relation $x \Rightarrow mi$ exists, where $mi \in MA_i$, and $x$ is any media item, or

(b) an event $start(M)$ has occurred where $M \in MA_i$ is a module.

In other words, in case (2a), only one media item in each state acts as the "starting" object, the others being activated in parallel with it, or being already active by virtue of previous events. Let us call it media item $mi_0$. In case (2b), the module itself acts as the "starting" component of the presentation fragment.

3. If condition (2b) holds, the fragment to be returned is the module itself.

4. Otherwise, the minimum presentation fragment to be returned to the user is described by a schema obtained from the original presentation schema with the following transformations:

   (a) if relation $M \Leftrightarrow mi_0$ does not exist, where $M$ is the presentation module containing $mi_0$, add it, and remove any other existing relation $M \Leftrightarrow mi_{ch}$ where $mi_{ch} \in MA_i$ and $channel(mi_0) = channel(i_{ch})$;

   (b) if $ComponentOf(x, mi_0)$ and relation $x \Leftrightarrow mi_0$ does not exist, add it, and remove any other existing relation $x \Leftrightarrow mi_{ch}$ where $mi_{ch} \in MA_i$ and $channel(mi_0) = channel(mi_{ch})$;

   (c) for each media item $mi \in MA_i$, if $mi \notin (\Leftrightarrow)Closure(mi_0)$ and the synchronization rule $mi \Leftrightarrow mi_0$ does not exist, add relation $mi_0 \Leftrightarrow mi$;

   (d) iteratively remove all items $x \notin Reach$, where $Reach$ is the set of reachable objects defined as follows:

      i. for any item $mi \in MA_i$, $mi \in Reach$;

      ii. for any items $x$ and $mi$ such that $mi \in Reach$, if the synchronization rule $mi \Rightarrow x$ exists and $\exists k \mid x \in MA_k \wedge s_k \in S_r \wedge s_{j+1} = next(s_j, e) \wedge i \leq j < k$, for any event $e$, then $x \in Reach$[5];

      iii. for any item $x$, if the synchronization rule $mi \Leftrightarrow x$ or $x \Leftrightarrow mi$ exist and $mi \in Reach$, then $x \in Reach$.

In this way the resulting fragment contains only the media items that in the original presentation play together with the items retrieved by the query. The items which temporally precede the retrieved ones are removed and the fragment ends when the retrieved media items are no longer active. This point deserves some comment. In step 4(d)ii the procedure retains the part of the presentation corresponding to the sequence of states $s_i \dots s_k$ such that $s_{j+1} = next(s_j, e), i \leq j < k$ where the retrieved media are active in all the states $s_j$ and not active in $s_{i-1}$ and $s_{k+1}$.

In terms of the presentation synchronization relations, if state $s_k$ is not a final state, an $x \Rightarrow y$ relation must exist for a medium $x$ in state $s_k$. If this relation is removed from the presentation, and iteratively all the unreachable media items are also removed, the presentation stops playing when leaving state $s_k$.

[5] $S_r$ is defined in step 1 of this procedure.

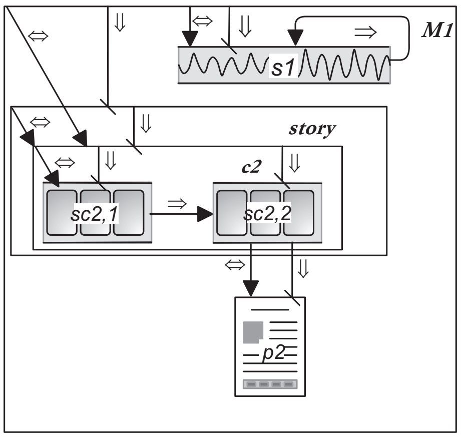

**Figure 3. The resulting presentation**

This procedure assures also that if two media items $mi_1$ and $mi_2$ belonging to the same automaton state $s$ are retrieved, the same presentation fragment is returned for both. Finally, channels are preserved since they are associated to the media statically.

As an example, let us suppose that a user's query returns a triple $\langle c_2, animation, P \rangle$ where $P$ refers to the schema presented in Section 3. The set of states in which $c_2$ is active is $S_r = \{s_3, s_4\}$ (see Figure 2). Since $s_4 = next(s_3, end(sc_{2,1}))$ we consider only $S_c = \{s_3\}$. The set of objects to be returned to the user is equal to $MA_3 = \{M_1, sound, story, c_2, sc_{2,1}\}$. Since the synchronization rule $c_1 \Rightarrow c_2$ exists, $c_2$ is the "starting object" $mi_0$. Then the synchronization relationships $M_1 \Leftrightarrow story$ and $story \Leftrightarrow c_1$ are replaced by the relations $M_1 \Leftrightarrow c_2$ and $story \Leftrightarrow c_2$. The resulting fragment is shown in Figure 3 where the "unreachable" item have been removed (and consequently all the synchronization rules involving at least one of them have been removed).

## 6 Discussion

The presentation fragment built by the procedure in Section 5 stops its execution when the media returned as query results are no longer active. We could ask if this behavior, which is consistent with the way the presentation is transformed, is correct for the user's expectation. Indeed, such a sharp identification of the temporal scope of the fragment could lessen the significance of the result.

The presentation once started could follow its complete execution up to its end. In this case, the system returns to the user an *access point* and leaves the user free to stop the presentation playback when he/she wants[6]. Another choice could be to identify the fragment scope based also on the presentation static structure: a meaningful fragment ends together with the module which contains it. These solutions can help the user to better understand the context and the significance of the resulting fragment. Therefore, while it is easy to set the beginning of the relevant scope of the fragment, its end is more a matter of meaning and semantics rather than of structure.

A second comment concerns the hierarchical structure of the presentation in terms of modules, stories and sections. They build up a hierarchy of contexts which can give the user different levels of access to the presentation content. We do not elaborate further on this issue here, suggesting that the retrieval model we have presented identifies only a minimal scope of the fragment of the presentation and a set of media temporally related. The design structure of the presentation can integrate this dynamic information with other information related to the identification of different "meaning scopes" in the presentation.

In the same way, we suggest that a narrower scope than the one defined here could come from computing the $(\Leftrightarrow)$ *Closure* and $(\Leftrightarrow)$*1step* sets on the retrieved media items. The scope is narrower because it does not consider media which are not directly connected by synchronization relationships to the retrieved ones. E.g., a background soundtrack which starts at the beginning of the presentation and lasts up to the end does not bear a meaningful content to the presentation evolution, therefore it could be left out from the presentation of retrieval results, at least in a first browsing phase where the relevance of the returned items is evaluated by the user. In the same way, elements like generic menus, banners, advertisements, sidebars, which often surround the core information in Web documents are not only scarcely relevant, but can deviate the user attention from the primary results of information retrieval. Our future work will better focus such issues.

## References

[1] S. Adali, M. L. Sapino, and V. S. Subrahmanian. An algebra for creating and querying multimedia presentations. *Multimedia Systems*, 8(3):212–230, 2000.

[2] C. Baral, G. Gonzalez, and A. Nandigam. SQL+D: extended display capabilities for multimedia database queries. In *ACM Multimedia 1998*, pages 109–114, Bristol, UK, September 1998.

[3] C. Baral, G. Gonzalez, and T. Son. A Multimedia display extension to SQL: Language and Design Architecture. In *International Conference in Data Engineering*, Orlando, FL, USA, February 1998.

[4] 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.

[5] Y. Chiaramella. Browsing and Querying: Two Complementary Approaches for Multimedia Information Retrieval. In *Hypertext - Information Retrieval - Multimedia '97*, pages 9–26, Dortmund, WA, USA, September 1997.

[6] I. F. Cruz and W. T. Lucas. A Visual Approach to Multimedia Querying and Presentation. In *The Fifth ACM International Conference on Multimedia '97*, pages 109–120, Seattle, WA, USA, November 1997.

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

[8] T. Lee, L. Sheng, N. Hurkan Balkir, A. Al-Hamdani, G. Ozsoyoglu, and Z. Meral Ozsoyoglu. Query Processing Techniques for Multimedia Presentations. *Multimedia Tools and Applications*, 11(1):63–99, 2000.

[9] T. Lee, L. Sheng, T. Bozkaya, N. Hurkan Balkir, Z. Meral Ozsoyoglu, and G. Ozsoyoglu. Querying Multimedia Presentations Based on Content. *IEEE Transaction on Knowledge and Data Engineering*, 11(3):361–385, 1999.

[10] R. J. Miller, O. G. Tsatalos, and J. H. Williams. Integrating Hierarchical Navigation and Querying: A User Customizable Solution. In *Electronic Proceedings of the ACM Workshop on Effective Abstractions in Multimedia*, San Francisco, CA, USA, November 1995.

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

[6]This behavior is obtained removing step 4(d)ii from the procedure described in Section 5.