Schema Modelling for Automatic Generation of Multimedia Presentations

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

ABSTRACT

Multimedia documents are an effective way to present different kinds of information, since the integration of different media types gives more expressive power and opportunities to catch the user attention. A multimedia report is a multimedia presentation built on a set of data returned by one or more queries to multimedia repositories, integrated according to a schema with appropriate spatial layout and temporal synchronization, and coherently delivered to a user for browsing. We discuss the problem of defining schemas for such multimedia reports with a focus on the media coordination and synchronization constraints. Multimedia presentations can then be automatically generated according to the schema. We have defined an XML language to describe the spatial layout and the temporal constraints of the media objects, together with a visual authoring system helping the user to design the presentation template.

1. INTRODUCTION

Rapid progress in technology for display, creation, storage and transfer of multimedia documents gives the user new possibilities to access and retrieve information of different kinds. Web sites offer a great variety of media such as text, images, video or audio files; multimedia documents are an effective way to present different kinds of information, since the presence of various media types gives more expressive power and opportunities to catch user attention. As an example, think of the description of a scientific experiment, where an animation can bring more information than a large amount of numeric data, or of a catalogue of hotels where the presence of images and videos helps the user to better compare the different choices.

The authors of multimedia presentations must design the coordinated playback of different media and the consistent interpretation of user interactions. In many applications information displayed comes from a data repository, and its identification and extraction requires additional design care and the use of a schema for data. Re-use of media and information for different purposes, or adaption to user profile and history, are other requirements asking for the design of presentations according to well defined models and schemas.

In such a scenario the automatic generation of standard multimedia presentations with data extracted from a repository is a valuable goal that allows authors to build with limited effort several variants on a same schema without re-designing the whole application from scratch.

We aim at automatically generating multimedia presentations by developing schemas based on recurring patterns, focusing the discussion on coordination and synchronization of continuous media. In previous works we have defined a synchronization model and developed an authoring tool for multimedia presentations [6, 9]. In this paper we extend both the model and the authoring tool in order to model schemas for automatically building multimedia presentations. With a schema authoring system we aim at giving the author the possibility to define the layout and the behavior of the presentation, the characteristics and attributes of the objects involved without knowing the instances that will be used to fill the schema.

The paper is organized as follows: Section 2 introduces multimedia reports as a class of multimedia presentations. Section 3 reviews the relevant literature. Section 4 discusses authoring of multimedia presentations in terms of synchronization among the media objects. In Section 5 an XML language suitable to describe the spatial layout and the temporal constraints of multimedia presentations and report schemas is presented. Section 6 discusses schema and data integration, while Section 7 discusses some problems related to consistency in data selection as concluding remarks.

2. AUTOMATIC GENERATION OF MULTIMEDIA PRESENTATIONS

The automatic generation of multimedia presentations is based on three steps:

1. Data is selected (retrieved) from a data repository. Each item is a media file (or part of it) with attributes that describe its contents (in terms of selection parameters), its playback features (e.g. format, time duration, size on screen, etc.) and possibly additional information useful to relate the item with other data,

e.g., indexes and cross-references.

2. A presentation schema is defined in which the multimedia items can be placed in a coordinated way according to the desired dynamics.

3. The schema is filled with the selected data retrieved, the spatial and dynamic relationships are instantiated, and the presentation is played.

Steps 1 and 2 can be interchanged. In effect, it is plausible that the schema is defined, to some extent, before data selection, since data objects are retrieved also according to presentation properties, e.g., media richness, screen size and layout, target audience, and so on.

Globally, the three steps build a continuous presentation in which data extracted from a multimedia data repository is located, connected, synchronized and coherently presented to the user. We call this activity *multimedia reporting*, i.e the automatic generation of multimedia documents from data retrieved according to selection parameters and modelled with respect to a template [7].

In the most general case multimedia reporting would require the designer to approach and solve many problems about data selection, e.g. how to coherently integrate data coming from one or several databases. In Section 7 we shall briefly discuss some issues about this problem, we note here that too much generality prevents from satisfactory solutions and is in some way in contrast with the idea of "reporting", an activity based on standardization. Therefore we assume the following scenario for our work:

1. The presentation collects data into groups, like in a text report, such that in each group data of different type exist (e.g., video, audio, text, image), whose instances are related like in a relational table. More precisely, we assume that each group is structurally equivalent to a relational table where columns identify the media types and rows are instances. Some values can be NULL values, denoting that in some instances some media can be missing.

2. Apart from groups, "background" data exists which are associated to the presentation as a whole, or to parts of it identified by a group or a sequence of groups, e.g. a continuous soundtrack, a permanent title, a background image, and so on.

3. The whole presentation consists of the coordinated (e.g. sequential) playback of the groups, taking care of user actions like pause, stop, rewind, and so on.

4. No *a priori* constraint is put on the time properties of continuous data items, but the system should be able to coordinate the execution by synchronizing the beginning and end of the data group components.

Conceptually, defining multimedia reports is not different from building text-only reports: the author must define the structure of the report, i.e. the data layout, and the query to select and retrieve relevant data. In the case of multimedia reporting, data items collected have a temporal behavior, which increases the complexity of the structure definition by adding a new dimension to the task: the author should deal with synchronization problems and temporal sequencing of objects. Moreover, if the spatial layout definition could be trivial, this is not true for the temporal dimension.

As an example, an author could design a news-on-demand service based on a database of articles stored as related multimedia document items: video documentaries, audio and text comments, images, and so on. A multimedia report is built from the selection of the appropriate news, by presenting them as in a synchronized sequence. Each article has a video story, an audio comment, and a text, which must be properly synchronized. The articles are normally played one after the other, but user interaction can modify the sequence, e.g. by skipping forward, or going back, or selecting the articles from an initial menu, and so on.

3. RELATED WORK

The problem of automatic generation of synchronized multimedia presentations as answer to user queries has been approached in recent years from several points of views. In [1] Adali et al. present a process algebra for querying multimedia presentation databases. The algebra can be used to locate presentations with specific properties but also for creating new presentations by retrieving different objects from existing ones. A multimedia document is represented by a tree, whose branches describe all the possible presentation sequences. The authors create a presentation by selecting a path in the tree.

SQL+D [4, 5] is an extension to SQL which allows users to retrieve multimedia documents as result of querying a multimedia database. An SQL+D query specifies all presentations properties, from screen layout to its temporal behavior. In addition to SELECT-FROM clauses the user can define DISPLAY-WITH clauses to describe screen areas (called *panels*), in which groups of retrieved media items are placed with specified relative positions. A SHOW clause defines the temporal behavior in terms of timed sequences of returned instances display.

Different from SQL+D which considers only data stored on multimedia databases, Delaunay^{MM} [8] is a framework for querying and presenting multimedia data stored in distributed data repositories, including the Web. Query answers are organized into multimedia presentations and profiles are used to generate user-defined layout of a document and ad hoc querying capabilities to search each type of media item. While Delaunay^{MM} addresses the specification of spatial layout, it does not deal with the problem of the temporal synchronization of media objects.

Geurts et al. [10] present a formalism to construct multimedia documents by defining high-level semantic relations between media objects. Both spatial layout and temporal dynamics are described through the use of quantitative and qualitative constraints. Quantitative constraints are often not sufficient because they address low level solutions (e.g., the author states that figure A is on the left of another object, but is not interested in specifying the number of pixels between them). The author imposes a set of constraints and the system chooses the solution as the multimedia presentation to present to the user. A prototype is developed called Cuypers [12, 13], which is a transformation environment supporting semi-automated assembling of multimedia documents according to a rich structural and semantic annotation based on XML. The annotation allows for the specification of different processing steps concerning semantic structure, constraints satisfaction and final form presentation, which occur in multimedia authoring, to be integrated in a single execution stream.

In [3], André presents an alternative approach to the problem of automatic generation of multimedia documents, based on concepts already developed in the context of natural language processing. The author considers the generation of multimedia presentations as a goal-directed activity. The input is a communicative goal with a set of parameters, like target audience and language, resource limitations and so on. The planning component of the system selects a multimedia presentation structure on the base of some communicative rules, and retrieves elementary objects like text, graphics or animations. The temporal behavior is expressed by temporal relations similar to the ones defined by Allen [2] and by metric (in)equalities.

4. DYNAMICS DEFINITION IN MULTIMEDIA PRESENTATIONS

A graph is a visual representation commonly used for describing the temporal behavior of a multimedia presentation. In previous papers [6, 7] we have defined an event-based synchronization model among continuous and non continuous media in a multimedia presentation. The reader is referred to the cited works for a discussion of details and motivations about the model, whose main properties only will be recalled in this paper. Then we have developed a visual authoring tool [9] which allows the author to define the temporal relationships between media by drawing a graph where the nodes are the media objects and the edges the synchronization relationships. Also the template of a multimedia report can be defined by linking object placeholders (the nodes of the graph) with labelled edges (the temporal relations).

Figure 1 illustrates a simple graph showing a (simplified) news-on-demand presentation made of three articles, drawn by our authoring system according to our synchronization model. The symbols $\Leftrightarrow$ and $\Rightarrow$ stand for two synchronization relations which roughly correspond to parallel and sequential play of media objects, and which will be briefly commented in Section 5. They are similar to the `<par>` and `<seq>` tags of SMIL [11], but differences exist which are not detailed here. The graph specifies that, while a background soundtrack is continuously repeated, the three articles are played in sequence, and each article is made of a spoken narration ($news_i$), a video ($video_i$) and a text caption ($caption_i$). The length of each article is controlled by the length of the narration, which is the master medium driving the parallel play of the other two media (the dot at the end of the line denotes the dependent medium). When the last article ends the soundtrack is replaced by a jingle and a credits screen is displayed.

Each media object is associated to a *channel* which defines

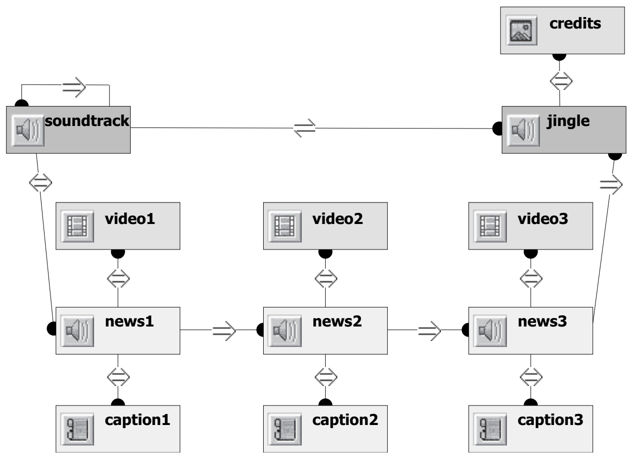

Figure 1: A simplified synchronization graph for a news-on-demand presentation

the place or device used for playback. According to the type of media hosted, a channel is a region of the user screen or an audio device, represented in different colors in the graph.

In the graph of Figure 1 a recurring pattern is immediately perceivable, therefore a schema for a multimedia report which generically displays selected news in sequence can be derived straightforwardly.

The syntax used for drawing a schema graph must make evident which are the items which build up the repeated media groups. The schema does not define the object instances involved in the presentation, since the cardinality of the media set returned by querying the repository is unknown till execution. Therefore each node of the graph is a placeholder for a collection of media with the same characteristics. The schema editor must provide the author a means to draw the structure of a report specifying which part of the structure is a replicated group, the relations inside a group and between different instances of the replication.

Figure 2 shows pictorially a report schema. The thick rectangle encloses the media placeholders which make up a repeated element, i.e., an *article*, and specifies which events are generated and which synchronization relationships are obeyed. It is then used as a *composite media item* for defining the dynamics of the multimedia report. The template is repeated twice, with the dotted line in between defining the iteration, subject to the following rules:

1. The first instance is executed according to the entering synchronization relationship (in the example, as the soundtrack starts), and the composing media are synchronized as described by the composite details.
2. The instance execution ends according to the synchronization schema described in the composite element, the events are propagated out of the composite according to the relationship which links the two composites. Then the subsequent instance is started according to the same synchronization schema; in the example the second narrations starts, and the video and caption

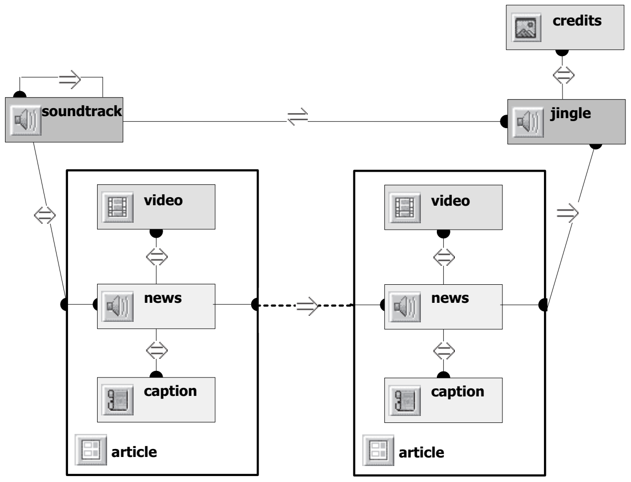

Figure 2: A visual template for a news report

are played in parallel with it.

3. When the last instance ends the events are propagated as described by the links exiting from the second composite, in the example the soundtrack is replaced by a jingle, the credits screen is displayed, and the presentation ends.

The report is thus divided into a number of units some of which are single instances, which define the presentation elements which do not depend on queried data, other units are structural templates to be iterated on actual data instances. In Figure 2 the soundtrack, the jingle and the credits screen are single instances, while the articles are templates.

The graph is a visual representation of the report template, which contains all information about temporal relationships of media objects. Spatial relationships are described by the channels associated to the media items.

This external representation is used by the visual authoring system and is supported by an XML-based language for defining the data structure and relationships in a more suited machine processable representation.

5. AN XML-BASED SCHEMA DEFINITION FOR MULTIMEDIA REPORTS

Hierarchical structure and temporal behavior of a multimedia presentation can be described using an XML schema 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.

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 model such information is often mixed. For example in SMIL[11], 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.

An XML source document describing a hypermedia presentation (and a multimedia report schema) contains three kinds of specifications: the spatial layout of the document, the media involved in the presentation and their temporal behavior. Data is thus organized in three sections: the *layout section*, the *components section* and the *relationships section*. This solution allows for the definition of spatio-temporal relationships among media objects without knowing any information about their location or duration, and makes simpler to draw a report template which can be instantiated with minimal modifications of the report data.

This approach is therefore well suited for describing multimedia report schemas. The author defines the temporal behavior by addressing abstract media objects identifiers (i.e., placeholders of actual data) rather than actual instances. The system will bind object identifiers, defined in the component section, to actual media objects after retrieving the data. The final presentation can be rendered by processing the XML file and accessing media objects which are located elsewhere.

The XML language is based on the same synchronization model underlying the graph representation.

5.1 The Layout section

The `layout` section contains the definition of the spatial layout of media in the presentation window. The presentation layout is organized in channels, which are a combination of a portion of the user screen, hosting some media, and audio devices to play soundtracks or other audio files.

Figure 3 shows the XML template of a news-on-demand multimedia report modelled according to Figure 2. The channels `video` and `caption` are rectangular areas of the user screen delimited by the corner coordinates `SupX`, `SupY`, `InfX` and `InfY`. The captions contain textual information about the news report. `Voice` and `sound` are audio channels, therefore they have no layout. Each channel has a unique `name`.

5.2 The Components section

The `components` section contains the description of the media objects involved in the presentation, their types, links to media files, channels used, etc. Media objects are organized into modules: each presentation contains at least one *module* enclosing continuous media objects, called *clips*, and static media objects called *pages*. The synchronization model provides also a hierarchical structure of continuous objects involved in a multimedia presentation; the reader is referred to [6] for details.

Each element has a unique identifier `id`, which is used to reference the object from the other sections of the document, a `type` and a `channel` in which it is played.

If the XML specification describes a presentation, the clips and the pages have an attribute `file` whose value is the

```
<presentation xmlns="report.xsd">
 <layout width="500" height="400" >
  <channel name="video" SupX="19" SupY="13" InfX="471" InfY="321"/>
  <channel name="caption" SupX="19" SupY="324" InfX="471" InfY="391"/>
  <channel name= "voice"/>
  <channel name= "sound"/>
 </layout>

 <components>
  <module id="news">
   <clip id="soundtrack" file="sound.wav" channel="sound" type="audio"/>
   <composite id="article">
     <clip id="video" channel="video" type="video" />
     <clip id="news" channel="voice" type="audio" />
     <page id="caption" channel="caption" type="text" />
   </composite>
   <clip id="jingle" file="jingle.wav" channel="sound" type="audio"/>
   <page id="credits" file="credits.txt" channel="video" type="image"/>
  </module>
 </components>

 <relationships>
  <play>
   <master><cont_object id="soundtrack"/></master>
   <slave><ref_composite id="article" num="first"/></slave>
   <master><ref_composite id="article"/></master>
   <slave><cont_object id="news"/></slave>
   <master><cont_object id="news"/></master>
   <slave><object id="video"/></slave>
   <master><cont_object id="news"/></master>
   <slave><object id="caption"/></slave>
   <master><cont_object id="jingle"/></master>
   <slave><object id="credits"/></slave>
  </play>

  <act>
   <ended><cont_object id="soundtrack"/></ended>
   <actvated><cont_object id="soundtrack"/></activated>
   <ended><ref_composite id="article"/></ended>
   <activated>
        <ref_composite id="article" num="next"/>
   </activated>
   <ended> <ref_composite id="article" num="last"/> </ended>
   <activated> <object id="jingle"/> </activated>
  </act>

  <repl>
     <before> <object id="soundtrack"> </before>
     <after> <object id="jingle"> </after>
  </repl>
 </relationships>
</presentation>
```

Figure 3: XML schema for a news report

path of the media files. If the XML specification describes a report it is a template: pages and clips definitions are placeholders for retrieved items, therefore the attribute `file` will be added at presentation generation phase.

The media can be defined inside a tag `composite`, which represents a composite media item which builds up a report repeated item. In Figure 3, the clips `soundtrack` and `jingle`, and the page `credits` refer to media objects which do not depend on the report instantiation (therefore the attribute `file` is defined), while the `composite` section represents the thick rectangle of Figure 1, which will be instantiated on the results of the query execution.

Each item inside the tag `composite` represents a class of objects; the items are instantiated by querying a data repository. In Figure 3, the clip `video` represents the set of videos returned for the selected news, `news` is the set of voice comments and `caption` is the set of text pages related to the same news. The attribute `file` is missing because it will be added during the schema and data integration phase.

5.3 The Relationships section

The `relationships` section describes the temporal behavior of objects through a list of synchronization primitives needed for the correct playback of the presentation.

The tags `play` and `act` define the basic relationship of parallel and sequential synchronization of media objects that in the visual representation are denoted by the symbols $\Leftrightarrow$ (*plays with*) and $\Rightarrow$ (*activates*). The tag `play` defines the synchronous starting of a `master` and a `slave` objects. As soon as the master object ends playing, the slave is terminated too. In the example, the video and the caption play in parallel with the voice comment, and remain active as long as the audio file plays. The tag `act` defines the sequential composition of two objects. When the `ended` object ends, the `activated` object begins to play. In the example, the news are played one after the other. Both the `master` and the `ended` objects of the relationship `play` and `act` must be continuous media, static media have no defined duration.

The tag `repl` defines sharing of a same channel between two objects which are active at different times in cases different from simple sequencing. The tags `before` and `after` define respectively the replaced and the replacing objects in the channel usage. In the example the jingle replaces the soundtrack in the same audio channel.

Two other tags, which do not appear in the example, explicitly model user interaction: the tag `stop` associates two media which play together (by direct relationship or a consequence of a series of relationships) and models the synchronous stop of one of the two objects when the other object is forced to end. The tag `link` defines the behavior of the original presentation once a hyperlink is followed. The attribute `behavior` defines whether the presentation is paused or stopped.

Objects linked by the relationships are referred by two tags: `object`, which addresses both continuous and static objects, i.e. modules, clips and pages, and `cont_object`, which addresses only continuous media, i.e., modules and clips.

In an XML template, relationships can be establish between objects and composite items. In this case, the tag `ref_composite` is used. Relationships between composite items must be careful evaluated during the schema and data integration phase. The attribute `num` identifies which instance of the composite is referred: the `previous` instance, the `next` one, or the `first` or `last` instance.

A specification facility related to some aspects of the presentation structure is defined, which allows the system to automatically infer some relationships by examining the hierarchical structure of modules and clips. However we do not discuss this facility here, simply noting that an optional attribute `from_struct` denotes such inferences.

6. SCHEMA AND DATA INTEGRATION

Once media objects are collected from query results, the template is instantiated on the objects retrieved in order to generate the actual multimedia report. Figure 4 shows the XML description of the presentation described in Figure 1.

The layout section of the XML document is not affected by report instantiation, which involves the objects and their relationships, but does not modify the layout. Only the reference to the `namespace` needs to be modified, since the XML schema refers to the namespace defined for presentation templates, while the XML presentation addresses the one defined for final multimedia presentations.

The `components` section must be extended to address the objects retrieved. The query returns an ordered sequence of tuples, whose elements belong to the media classes defined in the report schema by the objects inside the composites. In our example the returned set is:

$$RS = \{(video_i, news_i, cap_i) \mid 0 < i \leq |RS|\}$$

The `components` section can be completed by replacing the composite elements with the description of the objects defined inside them. The composites are replicated $|RS|$ times, and the instances are distinguished by systematically changing the object name placeholders in the template. The attribute `id` is instantiated by appending a sequence number i, and the attribute `file` is added to each object, referring the actual media locations. Media outside the composite elements remain unchanged, while the composite is replaced by the media of the resulting set. In the same way, relationships between objects outside the composite remain unchanged while relationships between objects inside a composite element are replicated. The result is shown in figure 4.

The management of relationships which involve composite media items and other objects is a bit more complex. With reference to Figure 2, a composite item can be both the origin and the end of a dynamic synchronization relation. The attribute `num` of the template definition points out which tuple of the resulting set is affected by the relationship. If it is not present, the relationship must be replicated for any tuple of the set, otherwise it involves only the selected tuple.

In the example the soundtrack play starts the execution of the first composite element. Since the attribute `num` is present with value `first`, the relationship must be evaluated only once, and refers to the first instance of the composite. Each composite element starts playing the audio file instantiated for the `news` item, since the attribute `num` is not defined in the relation `play` between `article` and `news`. The relationship `play` ($\Leftrightarrow$) is propagated from the composite to the voice comment. The object `news` plays the role of a master, since it starts the video and the caption, and stops their playback when ending. Its ending coincides also with the ending of the composite component.

In the generated presentation therefore a relation is established between the soundtrack and the item associated to the first instance of the audio files containing the news narrations, as depicted in Figure 2. For the other instances of the composite, the relationships between the objects inside them are replicated.

At the end of the last composite, the relation `act` between `article` and `jingle` starts the play of a jingle. According to the `num` attribute of the composite element, this relation must be translated only for the `last` tuple, thus the relationship `act` is instantiated between `news3` and `jingle`.

An `act` relationship exists between the two composites, specifying at both ends an item of type *news*. The relation must be replicated for all the tuples of the resulting set, since the attribute `num` is not defined in the first element of the presentation, but assumes the value `next` in the second element. The relations instantiated are therefore

$$news_i \Rightarrow news_{i+1}, 1 \leq i \leq 2$$

making the generated presentation to play sequentially the articles one after the other.

```
<presentation xmlns="model.xsd">
  <layout width="500" height="400" >
   <channel name="video" SupX="19" SupY="13" InfX="471" InfY="321"/>
   <channel name="caption" SupX="19" SupY="324" InfX="471" InfY="391"/>
   <channel name= "voice"/>
   <channel name= "sound"/>
  </layout>

  <components>
   <module id="news">
     <clip id="soundtrack" file="sound.wav" channel="sound" type="audio"/>
     <clip id="video1" file="video1.mpeg" channel="video" type="video"/>
     <clip id="news1" file="news1.wav" channel="voice" type="audio"/>
     <page id="caption1" file="caption1.txt" channel="caption" type="text"/>
     <clip id="video2" file="video2.mpeg" channel="video" type="video"/>
     <clip id="news2" file="news2.wav" channel="voice" type="audio"/>
     <page id="caption2" file="caption2.txt" channel="caption" type="text"/>
     <clip id="video3" file="video3.mpeg" channel="video" type="video"/>
     <clip id="news3" file="news3.wav" channel="voice" type="audio"/>
     <page id="caption3" file="caption3.txt" channel="caption" type="text"/>
     <clip id="jingle" file="jingle.wav" channel="sound" type="audio"/>
     <page id="credits" file="credits.txt" channel="video" type="image"/>
   </module>
  </components>

  <relationships>
   <play>
     <master><cont_object id="soundtrack"/></master>
     <slave><cont_object id="news1"/></slave>
     <master><cont_object id="news1"/></master>
     <slave><object id="video1"/></slave>
     <master><cont_object id="news1"/></master>
     <slave><object id="caption1"/></slave>
     <master><cont_object id="news2"/></master>
     <slave><object id="video2"/></slave>
     <master><cont_object id="news2"/></master>
     <slave><object id="caption2"/></slave>
     <master><cont_object id="news3"/></master>
     <slave><object id="video3"/></slave>
     <master><cont_object id="news3"/></master>
     <slave><object id="caption3"/></slave>
     <master><cont_object id="jingle"/></master>
     <slave><object id="credits"/></slave>
   </play>

   <act>
     <ended><cont_object id="soundtrack"/></ended>
     <activated><cont_object id="soundtrack"/></activated>
     <ended><cont_object id="news1"/></ended>
     <actvated><cont_object id="news2"/></activated>
     <ended><cont_object id="news2"/></ended>
     <activated><cont_object id="news3"/></activated>
     <ended><cont_object id="news3"/></ended>
     <activated><object id="jingle"/> </activated>
   </act>

   <repl>
      <before> <object id="soundtrack"> </before>
      <after> <object id="jingle"> </after>
   </repl>
  </relationships>
 </presentation>
```

Figure 4: The generated presentation in XML

7. CONCLUSION

We have not discussed in this paper issues related to the query formulation and execution. This is of course a problem of crucial importance, and we do not claim it is easy to formulate formally and to solve. A number of questions must be answered, which however do not interfere with the schema model we have discussed here. We have assumed that data comes from one multimedia database, therefore media instances are naturally related to each other much as in a relational table. What if several data repositories are accessed? This situation seems desirable due to the large number of available media sources. However the problem may become intractable for several reasons:

1. Different data repositories can be different and hold data items which are semantically close but very far in their physical properties, e.g., different in video size, or in image resolution, or in audio fidelity. Therefore a coherent presentation including elements from all the repositories can be very hard or even impossible.
2. Different repositories can require different query languages, or queries with different parameters, due to the differences in DB schemas. What about result integration?
3. We assume that different kinds of media be returned. How are different elements related if they come from different databases, so that in principle only the interpretation of content can relate them? How can we"link", say, a video instance to an instance of a text related to the same article but coming from a different repository?

In approaching automatic generation of presentations therefore we are bound to a set of constraints which make our initial assumptions realistic and effective.

1. We must be able to select coherent data, i.e. data that is semantically related and that can be put in a presentation which is recognizable by the reader as a meaningful document. This problem is present in all the automatic presentation construction systems, and is assumed implicitly.
2. Data can be linked by external keys or equivalent cross-reference information which assure that we can identify related data by testing such information.
3. Data is coherent with respect to physical playback properties.

These requirements are satisfied if we have only one multimedia database. They can be guaranteed to some extent by filtering data coming from different databases, even if in this case the amount of homogeneity in the resulting presentation can be lower. We should in this case assume that automatic generation leads to a presentation prototype that has to be refined by hand.

Applications that can be satisfied with these requirements are wide: news-on-demand, that we have used as a scenario in a very simplified view, is a good case since the assumption that the same database holds news video with associated texts and audio is realistic. Advertisement is another good case, since it is plausible that a set of advertised items can be described each by one picture, or a video, a spoken text, a jingle, and so on, related by well identifiable keys.

In all cases the multimedia report can be completed with purely aesthetic media such as a background soundtrack, decorative frames, contour images, and so on, which can be described in the report schema or added in a subsequent editing phase.

8. ACKNOWLEDGEMENTS

This work has been supported by Italian Ministry of Education, University and Research (MIUR) in the framework of the National Project *DeVisu* and of grants for young researchers support.

9. 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] J. F. Allen. Maintaining knowledge about temporal intervals. *Comm. ACM*, 26(11):832–843, November 1983.

[3] E. Andrè. *A Handbook of Natural Language Processing: Techniques and Applications for the Processing of Language as Text*, chapter The Generation of Multimedia Documents, pages 305–327. Marcel Dekker Inc., 2000.

[4] 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.

[5] 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.

[6] 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.

[7] A. Celentano and O. Gaggi. Multimedia reporting: building multimedia presentations with query answers. In *Workshop on Multimedia Information Systems (MIS 2001)*, pages 61–71, Capri, Italy, November 2001.

[8] 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.

[9] O. Gaggi and A. Celentano. A Visual Authoring Environment for Multimedia Presentations on the World Wide Web. Technical report, Department of Computer Science, Università Ca' Foscari di Venezia,

Mestre (VE), Italy, February 2002, *Submitted for publication.*

[10] J. Geurts, J. van Ossenbruggen, and L. Hardman. Application-Specific Constraints for Multimedia Presentation Generation. In *International Conference on Multimedia Modeling 2001 (MMM01)*, pages 247–266, CWI, Amsterdam, The Netherlands, November 5-7 2001.

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

[12] L. Rutledge, B. Bailey, J. van Ossenbruggen, L. Hardman, and J. Geurts. Generating Presentation Constraints from Rethorical Structure. In *11th ACM Conference on Hypertext and Hypermedia*, San Antonio, Texas, USA, May 30–June 3 2000.

[13] J. van Ossenbruggen, J. Geurts, F. Cornelissen, L. Hardman, and L. Rutledge. Towards Second and Third Generation Web-based Multimedia. In *The Tenth International World Wide Web Conference*, pages 479–488, Toulouse, France, May 1–5 2001.