Authoring Smil documents by direct manipulations during presentation

Muriel Jourdan, Laurent Tardif and Lionel Villard

OPERA project, INRIA Rhône-Alpes.
655 avenue de l'Europe, 38330 Montbonnot, France.
e-mail: {Firstname.Lastname}@inrialpes.fr
http://www.inrialpes.fr/opera

Abstract - This paper presents Smily an authoring environment to write Smil documents. The main feature of Smily is to strongly integrate the presentation view, in which the document is executed, to the editing process. In this view objects can be selected to perform a wide set of editing actions from attributes setting to direct spatial or temporal editions. This way to edit a multimedia document is close to the well-known WYSIWYG paradigm used by usual word-processors. Moreover in order to help the author to specify the temporal organisation of documents, Smily provides her/him with an execution report displayed through a timeline view. This view also contains information which helps the author to understand why such execution occurs.

1 Introduction

The multimedia documents that we consider in this paper are collections of heterogeneous objects (such as video, audio, text, picture) organised in both the spatial and temporal dimensions. Moreover they support a hypertext structure to provide the reader with interaction capabilities. Several formats exist to model such documents: MHEG [8], HyTime [3], Smil [18] are the most well-known standards. Designing such a document is known to be a complex and error prone task for numerous reasons. Authors must handle large numbers of objects in the document, they have to deal with complex temporal information (synchronisation tasks and objects duration) and they cannot easily reuse parts of existing documents.

Thus, there is no doubt that using authoring tools dedicated to the design of multimedia documents is a better solution than using general programming environments. However, the design of relevant authoring tools raises several open issues and progress still needs to be made to build the "perfect" tool to make writing multimedia documents as easy as writing classical documents.

The World Wide Web Consortium contributes to the debate about which format to model multimedia documents, focusing mainly on their temporal dimensions. They proposed in June'98 [18] the first version of a declarative format, called Smil, which merged the absolute timing and relative positioning of object by mean of a hierarchical structure built upon sequential and parallel operators. Smil is defined as a tag based format and is encoded as an XML DTD (Document Type Declaration).

Although the number of Smil players have increased (such as [9] and [13]) since the birth of Smil, Smil editors have not followed the same evolution. It confirms that the design of a relevant authoring tool for multimedia documents is a non-trivial task. Commercial Smil editors (such as T.A.G [16] and Veon [17]) are mainly based on timeline interfaces and so have disadvantages associated with absolute timing. Smil documents written by using such tools are difficult to maintain. Moreover interactions and continuous media (like video, audio), whose exact duration are not known before their execution, are difficult to integrate. RealNetworks has chosen another approach [12] which is interesting for its simplicity but whose use is very limited. Indeed, it provides the author with a set of predefined templates, in which she/he is obliged to choose one solution in order to instantiate it with objects. As far as we know, only three authoring tools allow the author to really edit Smil structures: SmilComposer (a commercially available tool [11]), GRiNS (a research prototype which is in the process of being industrialised [1] and [2]) and the Smily editor we propose and present in this paper.

Smily is based on multiple views: the **presentation view** in which the document is played, the **timeline view** in which temporal information about the document is visualised and the **hierarchical view** in which the hierarchical structure of the document is displayed. Objects can be directly selected in the presentation view to apply specific editing actions such as changing object attributes or setting temporal operators. This provides authors with an easy way to modify their documents while playing them, without handling two different contexts: one for editing documents, the other for playing them. The selection can also be performed in the two other views if it is more relevant. Moreover, these three views are synchronised: an object selected in one view is highlighted in the other views. In this way the views are strongly linked and it is easy for the author to find an object in one view if this one can be easily selected in another one. Thus one user-friendly way to use the Smily editor consists in playing the document, pausing it, selecting objects with the mouse in one view and performing editing actions through menu selections.

The other interests of Smily are:

- the use of a timeline view to give visual information about the temporal behaviour of the document during the editing process. An execution report is displayed each time a new presentation of the document is played. It shows exactly when objects have been played and why such an execution occurs;
- the way the author designs the spatial layout of the document directly in the presentation view by using high-level spatial constraints. Direct editing in the presentation view allows authors to see both when and where objects appear on the screen.

Smily is the result of research on editing multimedia documents led by the Opera project for 5 years. The Opera project has conceived Madeus [5] a multimedia documents authoring tool based on the use of temporal constraints to describe the temporal organisation of the document. From this prototype, we have extracted a toolkit for building authoring environments of multimedia documents called Mikado [6]. This toolkit has been used to build the Smily editor.

The paper is organised in two parts. The first part contains both a brief presentation of the Smil format and a synthesis of existing Smil authoring environments. In the second part, the different features of the Smily editor are presented.

2 Smil: What is it and how do we edit it ?

Smil is recommended by the World Wide Web Consortium (W3C) which is currently in version 1.0 (the version that we will consider in this paper [18]). Smil is intended to easily integrate objects of different media types (video, audio, text, picture, ...) in order to provide both spatial and temporal organisations. In addition, mechanisms for linking parts of the document are provided. We only present here the main characteristics of the standard.

2.1 A brief presentation of Smil

A Smil document is composed of a Head and a Body part. The Head part contains the non-time-based information about the document: mainly the set of spatial regions which are used to define its spatial layout. The Body part contains both the set of objects with their attributes, their temporal organisation and the "hypertext" behaviour of the document.

The temporal organisation of a Smil document is based on a hierarchical structure of parallel and sequential synchronisation operators and on temporal attributes set either on objects or nodes. The associated semantics is defined by applying two sets of rules: the first one to define object duration and the second one to define operator behaviours. Rules which define object duration are the following: a "dur" attribute can set the duration of an object to an absolute value. If this attribute is not used, the duration of an object depends on its nature: a continuous object (which has an intrinsic duration) such a video or an audio is played until its end, a discrete object (which has no intrinsic duration) such as a picture or a text is played indefinitely. These first computed duration can be modified by the rules associated with each operator. The sequential (resp. parallel) operator expresses the sequential (resp. simultaneous) play of its operands. By default, the end time of a parallel construction is equal to the end of its longer operand. However, this semantics can be changed by using an "end_sync" attribute on a parallel node, in such a way that the end of the parallel construction will be equal to the end of its shorter operand (end_sync = first) or a designated one (end_sync = id(object), where the id function give the name of the object).

Such a hierarchical structure allows the author to define basic temporal behaviours. For more complex schedules, the author can modify the behaviour computed from the hierarchical structure (as this is explained in the previous paragraph) by using begin and end attributes which can express both absolute or relative offsets. An absolute offset put on a child of a parallel node (resp. sequential node) expresses a shift from the beginning of this node (resp. the end of the previous child). If the author uses a relative offset, she/he obliges the start or end time of an object to be equal to either the start or the end time of another sibling object.

Hypertext behaviour is expressed in Smil as in HTML by using "href" attribute on objects to define the destination of the link.

The last feature of Smil we would like to mention is the "Switch" element which allows the author to express several alternatives, one of which can be selected at runtime by the player. For instance it allows to play a picture instead of a video if the network is overloaded.

At present, Smily supports only a subset of the Smil standard. It allows authors to edit and play Smil documents based on a hierarchical structure built with parallel and sequential node. Moreover end_sync attributes can be set on parallel nodes. dur, begin and end attributes are also supported either on objects or on nodes but begin and end attributes can only express absolute offsets.

2.2 A working example written in Smil

The following example was designed to celebrate the new year in a more animated way than a traditional postcard. It is used in the section 3 to illustrate the presentation of Smily. Temporal attributes have been written in bold. Region attributes are not detailed.

```
1   <?xml version="1.0" encoding="ISO-8859-1"?>
2   <!DOCTYPE smil PUBLIC
3   "-//W3C//DTD SMIL 1.0//EN"
4   "http://www.w3.org/TR/REC-smil/SMIL10.dtd">
5   <smil>
6   <head>
7   <layout>
8   <region id="R" ... />
9   <region id="L1" ... />
10  <region id="L2" .../>
11  <region id="C" ... / >
12  </layout>
13  </head>
14  <body>
15  <par end_sync = "first" id = "par_main">
16    <img src="background.gif"
17         id="background" region="C"/>
18    <audio src="Song.wav" id="Song"
19         begin = "6" />
20    <seq id="seq_pic">
21      <img src="Reindeer.gif" id="Reindeer"
22           dur="18.0" region="R"/>
23      <img src="Name.gif" id="Name"
24           region="R"/>
25    </seq>
26    <seq id = "seq_txt">
27      <text src="Text1.html" id="Merry"
28            dur="9.0" region="L1"/>
29      <text src="Text2.html" id="And"
30            dur= "6.0" region="L1"/>
31      <par id="par_final" end_sync="first">
32        <text src="Text3.html" id="Happy"
33              dur="12.0" region="L1"/>
34        <video src="jumel.gif" id="Video"
35               region="L2"/>
36      </par>
37    </seq>
38  </par>
39  </body>
40  </smil>
```

The temporal behaviour of this document is as follows: a parallel node is used to start simultaneously the background picture, the song, and two sequences of objects called seq_pic and seq_txt. seq_pic is a sequence of pictures (between lines 20 and 25) and seq_txt is a sequence of three texts plus one video (between 26 lines 37). The two parallel nodes have the attribute *end_sync=first* which means that they end when their shorter operand ends. One begin attribute is used on the Song object (line 18) to shift its beginning 6 seconds after the beginning of the main parallel node.

2.3 Authoring environments of Smil documents

The following synthesis is based upon experimentation with some authoring environments (Veon [17], T.A.G [16], SmilComposer [11], Smil Wizard [12]) selected in the list given both at [7] and [15]. No free version of GRiNS [2] exists yet.

We can divide such authoring tools into two categories taking into account only the way they manage the temporal organisation of a document: those which use Smil as an output format but keep their own internal format during authoring processes, and the others (including Smily) that provide the author with a way to edit Smil temporal structures. We will first compare these two solutions, focusing at the end on the two closest authoring tools to Smily, namely SmilComposer and GRiNS. We finish this section with a few words about how the spatial layout is specified in each of these tools.

2.3.1 Two categories of Smil authoring tools

Current authoring tools that use Smil only as an output format are based either on timeline interfaces (Veon [17], TAG [16]) on which authors set objects with absolute timings, or on predefined templates to instantiate (Smil Wizard [12]). Advantages for authors for using such kinds of tools are twofold: firstly, they can go on to use their usual authoring tools without learning a new interface and secondly, they are not obliged to know something about the Smil format. However, in addition to disadvantages already mentioned in the introduction associated with absolute timing paradigm, such environments produce "low level" Smil documents: their hierarchical structures are very poor and objects are set with absolute timings using many begin and end attributes. Thus the output format does not keep the whole semantic of the document. For instance, some temporal synchronisation are not expressed by a sequence or a parallel node but are lost with the intensive use of begin, end, and dur attributes. Consequences for the authors are the following: firstly, the Smil documents they get cannot be edited "by-hand", for instance to add an attribute which is not supported by the authoring environment they used; Secondly, they are obliged to keep two formats of the same document: the Smil one for publishing on the Web and the proprietary format of the tool for editing purpose to be able to modify the document. In this case, the authors are obliged to always keep the same authoring tool. Finally, they cannot use Smil for editing purposes as an exchange format with other authors. The same is true with some HTML authoring environments (FrontPage is a good example of this situation) which produce low level HTML with the same disadvantages for authors.

To cope with these problems, some authoring tools are closer to Smil principles and provide the authors with a way to directly edit Smil structures. This is the case of both SmilComposer [11] and the GRiNS editor [1] (presented below) and also of Smily the Smil editor we present in this paper. Smil documents generated by these tools are more hierarchically structured to capture the whole semantic of the document.

SmilComposer and GRiNS provide the author with a hierarchical view coupled with a dialog box to set attributes on both objects and nodes of the document. The aim of the hierarchical view is to design the main structure of the document: adding parallel and sequence nodes, and objects under those nodes. SmilComposer visualizes this structure by a usual file manager window, whereas GRiNS visualizes in the same view both the structure and some information about the temporal behaviour of the document deduced from this structure. Indeed, objects are displayed by boxes along a vertical axis which represents the time progression (from top to bottom): two objects in sequence (resp. parallel) are shown one over (resp. beside) the other.

SmilComposer is only based on this hierarchical view to specify the temporal organisation of a Smil document. It uses the RealNetworks player G2 [9] to play the generated Smil document, without any link with the editing interface. The major difficulties for an author which uses this tool are to anticipate and to understand the temporal behaviour of her/his document from the hierarchical view only.

This is why, before executing the document, GRiNS provides the author with an additional view: the virtual timeline view which uses horizontal timelines to show when objects are played (only the leaves of the hierarchical structure are visualized) as calculated from the hierarchical view and attributes associated with objects and nodes. Objects which share the same region are set on the same line. The time axis used in this view is virtual: displayed duration are not the exact one due to the indeterminate duration of continuous objects which cannot be known before execution and the presence of switch elements which will be resolved at execution time. The timeline view of GRiNS provides the author with object selection capabilities for instance to change object attributes. Only begin and end attributes are visualized in the GRiNS timeline view (by arrows) to help the author to understand the temporal behaviour displayed. Finally, the document is played in a presentation view which is not integrated to the editing process (no object selection capabilities).

2.3.2 Spatial layout specification

The layout of the document is specified in each previously mentioned tool (Veon, T.A.G., SmilComposer, …) by using approximately the same mechanism: a layout window allows the author to draw boxes and to set between them usual spatial relations (such as centring or aligning) by grouping objects as in graphical editors. Then, the author associates each object with a region. The Smil format does not contain any kind of spatial relations to define the spatial layout of the document. However, since such mechanisms really ease the authors task, authoring tools of Smil documents handle this kind of information as extra editing tags (see below the solution applied in Smily).

This simple way to edit the spatial layout of a document is a direct consequence of the clear separation made in the Smil format between spatial and temporal information. However, this strong separation has the disadvantage that when the author draws regions in the layout window, she/he does not know if they will appear at the same time on the screen during the document presentation, while this information is important to decide their relative positions. Moreover for long documents, the whole set of regions drawn on the layout view at the same time is too complicated to manage for authors.

3 The main principles of editing with Smily

The highest number of Smil editors of the first previously mentioned category reflect some wait-and-see policy of commercial companies with regards to Smil. They prefer low-cost solutions rather than really investigate other solutions. This is time now to reach a new step to fulfil author needs completely. The Smil standard is enough mature to design editors which are strongly linked with its structures. This is the aim of Smily.

3.1 Smily objectives

The first objective of the Smily editor is to provide author with a "WYSIWYG" interface. This paradigm makes the success of word-processors. It has to be adapted to multimedia documents as explained in [4]. This implies to strongly integrate the presentation of the document to the authoring process. The spatial layout specification will strongly benefit by such an integration.

The second objective of Smily is to really ease the author in the most difficult task of multimedia documents design: the temporal synchronization specification. As in GRiNS, we promote the use of a timeline view to show temporal positions of objects deduced from the hierarchical specification. However, this first level of information must be completed by some explanations about the reasons of such positions.

Finally, we are convinced by the necessity of providing the authors of Smil documents with temporal checking capabilities during the editing process. This is a guarantee of the compliance with the Smil specification which defined some cases of temporal errors. Detecting such errors during the editing process and not at the end of an editing session (or before the presentation of the document) helps the author to find the source of the error. As far as we know, existing authoring tools do not address this issue.

3.2 General Overview

Figure 1 shows a screen dump of the Smily interface during an editing session of the previous example. Different areas (A, B, …) are displayed to ease the presentation:

- (A) is the presentation view which shows the presentation of the document with basic time control functions (C) such as play, pause and resume.

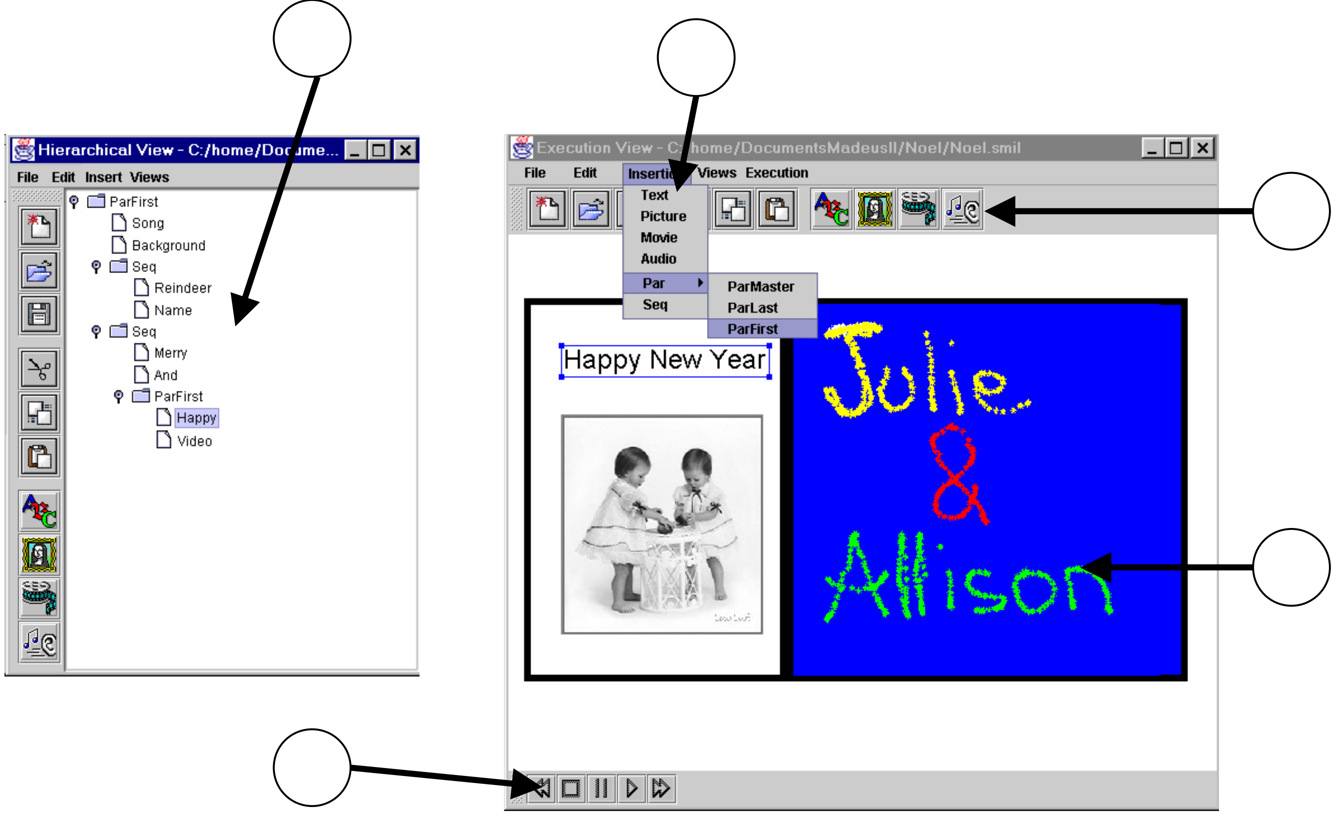

Figure 1 – *Smily Interface : hierarchical view on the left and presentation view on the right*

- (B) is the hierarchical view which shows both the set of objects involved in the document and their hierarchical organisation in terms of sequence and parallel nodes. "end_sync" attributes are also shown in this view. It is built like a "file manager" with close and open facilities on each node of the hierarchical structure.

- (D) is the control menu accessible through each view. It allows the author:

  - to perform usual action on a hierarchical structure: to add/remove/move an object at each level of the hierarchy and to add/remove a temporal node (E).

  - to set attributes on objects, regions or nodes with a dialog box.

  - to set spatial relations between region as explained in section 3.4.

The timeline view is presented in figure 2. It uses the timeline metaphors to visualize the temporal structure of the document. This view will be presented in more details in section 3.3.

To apply these actions, the author has to select objects or nodes in the most appropriate view. The three views are synchronised, that means that an object selected in one view is highlighted in the other. This is the case of the Happy object in figure 1. Thus, the author has no difficulty to make the link between the three views and more particularly between the presentation view and the others. With other words, she/he does not loose time to look for an object if this object can be easily selected in one view. Such a situation often occurs. This is the case, for instance when the author looks at the presentation of the document and wants to know which temporal operators affect an object currently displayed. The most appropriate view to find this kind of information is the timeline view but the object selection is easier in the presentation view, since the author directly sees the object on the screen.

One way to build a Smil document from scratch consists in adding new objects under the root node (present by default when asking for a new Smil document), thus associating with these objects temporal operators. At this point, the author can play her/his document and modify its layout (see section 3.4). If she/he wants to change something, she/he can pause the execution to perform the changes (for instance, adding a new temporal node in the hierarchy). Then, she/he can save and go on by replaying the document.

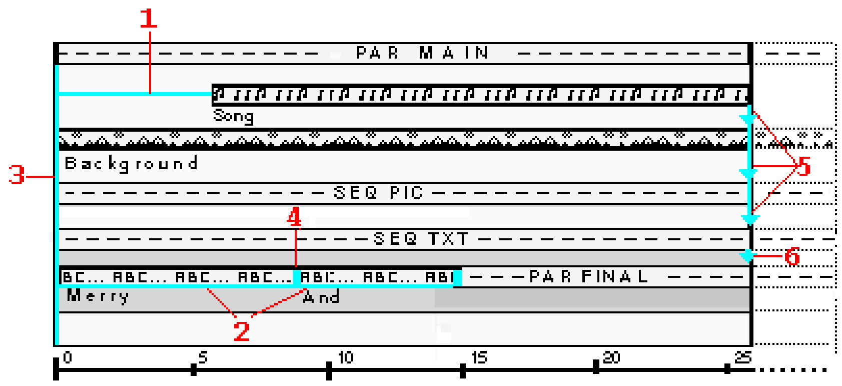

Figure 2 - *The Smily timeline view with some closed boxes*

3.3 Smily timeline view

Because our environment is based on a closed connection between edition and presentation, it was straightforward to build the content of a timeline view from an execution. As an **execution report**, it provides the author with a reliable perception of the temporal behaviour of the document, together with a way to understand this execution thanks to additional information. We choose to show on this view both basic objects (the leaves of the hierarchy) and nodes for two reasons. First, node visualizations are mandatory to display temporal consequences of some Smil elements. For instance, a begin attribute set on a child of a parallel node links the start time of this child and the start time of the parallel node. Second, nodes are a simple structure to use as the base of a zoom/unzoom mechanism The last feature of the Smily timeline view is that it also gives information about which objects have been interrupted or even have been not presented at all due to end_sync attributes. Both kind of information are useful to check if such end_sync attributes have been well set on the hierarchical structure of the document. This information are shown using dotted lines just after the execution report..

Therefore, this view is characterised by:

- objects displayed by boxes along the time axis, **exactly as they have been played.** Their start time can be read on the graduate axis and the length of their box gives their duration. Children of a sequence node (resp. parallel node) are drawn on the same horizontal line (resp. on successive horizontal lines).

- macro-boxes associated with sequential and parallel nodes which encapsulate the boxes of their children. These macro-boxes can be opened and closed by the author by using the mouse. The different level of the hierarchy are distinguished by backgrounds with different levels of grey colours.

- graphical (blue or red) marks associated with each temporal placement resulting from a Smil element (temporal operator or attribute):

  - a vertical line between two points (start or end times) indicates that an explicit Smil element implies that these points are simultaneous in every execution of the document.

  - a horizontal line is used to visualize either a begin, a dur or a end attribute. It links the two time points involved by the attribute.

- a vertical arrow links the end points of every operands of a parallel node which has an end_sync attribute equal to "first". This arrow goes from the operand which causes the end of the node to the other objects.

Moreover the timeline view is synchronised with the hierarchical view, through these marks: by selecting a vertical mark (resp. horizontal mark) the author can see which Smil nodes (resp. objects) is involved in the corresponding synchronization effect.

Figure 2 shows the timeline view computed after one execution of our complete working example. This execution is such that it is the Song object (due to its effective duration) which is the shortest operand of the <par end_sync=first> node line . In this figure, the seq_pic and par_final nodes are closed. Numbered red arrows are added to the timeline view to ease its presentation.

A blue horizontal line (arrow 1) links the start time of the main parallel node and the start time of the Song object due to the begin attribute set on this audio object. If this mark is selected, the Song object is highlighted in the hierarchical view, since the begin attribute is set on this object. The dur attributes on Merry and And are also displayed by such a blue horizontal lines (arrows 2).

A blue vertical line (arrow 3) is drawn between the start times of the "Par main" node children to show these time points are simultaneous due to the main parallel node. The same idea is used to visualize the effects of a sequential node: a blue vertical line (arrow 4) is drawn on the end time of Merry and the start time of And.

Blue arrows (arrows 5) link the end of the Song object to the end of the other operands of the par_main node, since in this execution, this is the Song object that causes the end of those objects. Moreover, the end of the seq_txt node interrupts the par_final node in its turn. This last effect is also visualized by a blue arrow (arrow 6). If this blue arrow is selected, the hierarchical view highlights the seq_txt node and the par_main node, since both elements explain the interruption.

Figure 3 gives the same timeline view than figure 2 but each node of the hierarchy are opened. In this view, we choose to colour in red the two horizontal lines which are the graphical vizualisations of the dur attributes set on Name and Happy. Indeed, the reported execution is such that the effective duration of these objects are shorter than the one specified by their dur attribute. We think that this is a relevant information to show to the author

On our example, one relevant information given by the second part of the timeline view (displayed using dotted lines just after the execution report) is that in the reported execution (which is a possible one) the video is interrupted. This behaviour is may be not wished by the author. One solution to preserve the content of the video, consists in removing the end_sync=first attribute on the par_main node and setting a dur attribute on the background object.

3.4 Editing Spatial Layout

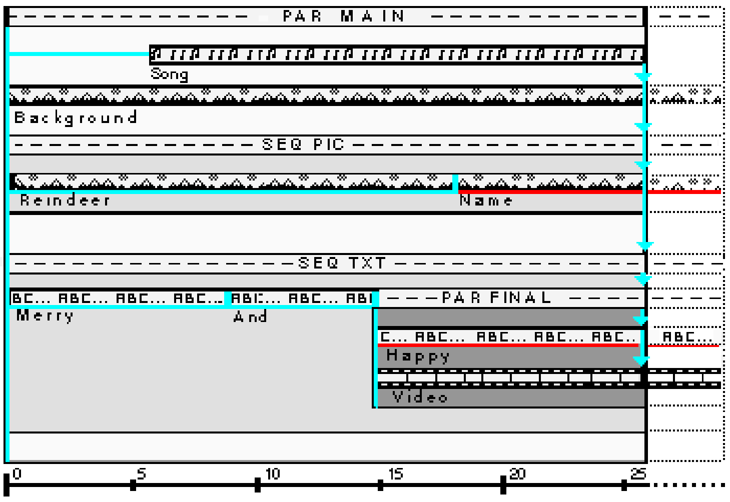

Figure 3 - *the Smily timeline view with each node opened*

The main feature of the spatial layout specification in Smily is to use the **presentation view** to move/resize objects directly on the screen. This is a straightforward consequence of the strong connection between the editing and presentation interfaces in Smily. Therefore, the spatial layout specification is not completely disconnected from the temporal organisation of the document as in the other editors we have tried. This means that authors see both when and where objects appear on the screen. This is very useful for authors to decide the relative positions of regions. Moreover, the time progression is used to reduce the number of regions shown at the same time, thus avoiding the second drawback (see section 2.3.2) attached with the use of a layout view: every regions are drawn at the same time.

For each object inserted in a Smil document (except audio elements), an associated new region is automatically created.

Region attributes (left, top, width, ...) can be modified by the author in several ways:

- directly through the general attributes dialog box, or by direct manipulations on the presentation view (moving, resizing objects).
- indirectly by setting spatial relations (centring, aligning,...) between objects through the presentation view. These relations are maintained while objects are moved on the screen by the author.

Likewise, the region attribute of an object, which specifies in which region this object will appear, can be modified either by using the attributes dialog box or by setting the specific "same region" spatial relation between two selected objects. It implies that the second selected object takes the same region as the first.

Spatial relations are handled in Smily by using constraint algorithms [10]. This has the following advantage: it is possible for an object to have spatial relations with different objects, each relation will be maintained while these objects are moved on the screen by the author. The same is not true with usual mechanisms used in graphical editors to handle such spatial relations: they are only maintained through an active group of objects and one object cannot be in more than one group at the same time.

Spatial relations between regions do not exist in the Smil format. It implies that during the saving of document, spatial relations are kept like extra editing information by adding new tags. In order to keep Smil document readable by other players, Smily uses namespace [18] as it is recommended in the Smil specification. It is also possible to save a "pure" Smil document (without namespace element), for instance to allow document validation.

3.5 Checking temporal consistency of the document

From the subset of Smil handled in Smily, there is no way to introduce a temporal inconsistency in a document. Indeed, the main source of temporal errors are relative offsets which are not yet supported. Smily does not support relative offsets because the semantics associated with such values of begin and end attributes are not well defined in the current version of the Smil specification[1]. Thus, we prefer to wait for a clearer definition of such values to integrate them in our editor. However, the set of rules given in this specification can be applied to some small Smil examples without ambiguities. Some of these examples break temporal properties express in the specification. We take them to illustrate the interests of using Smily as far as temporal consistency checking is concerned.

The two temporal rules which are expressed in the smil specification and that we consider here are the following: (rule 1) a begin attribute used with a relative offset can only delayed the start time of an object computed without this attribute; (rule 2) A relative offset defined as the sum of the start time of another object and an absolute value must be such that the absolute value is lower than the duration of the designated object.

The first Smil example that we consider has the following body part. It breaks the rule 1.

```
<seq>
  <picture id="A" dur="4s" .../>
  <picture id="B" begin="id(A)(begin)" .../>
</seq>
```

[1] Improvement on this point is provided for in the second version of Smil in addition with the extension of timing and synchronization functionalities (see [14])

The second example is obtained by replacing the value of the begin attribute of B by "id(A)(7s)" which means that the begin of B is equal to the begin of A plus 7 seconds. It breaks the rule 2.

Smily checks the two rules each time an editing action is performed on the document in order to warn the author as fast as possible. If an error is detected, this incremental detection helps her/him to find the source of the error. Smily is able to perform such verification thanks to its internal graph structure [6] which is maintained during the whole editing session. Algorithms used to detect such inconsistencies are inherited from the Mikado toolkit, since the same needs of temporal checking appear in the Madeus authoring tool [5].

4 Conclusion

In this paper we present Smily a Smil editor by comparing it with other commercial and prototype tools. Smily is an authoring tool which allows an author to edit Smil structures by direct manipulation in the presentation view. As far as we know, Smily is the only tool which integrates this view in the editing process. In this view, objects can be selected to perform a wide set of editing actions from simple attributes setting to direct spatial or temporal editions. This way to edit a multimedia document is close to the well-known WYSIWYG paradigm used by editors of textual documents. It reduces the length of the action/perception cycle and the authors do not need to manage an editing and a presentation contexts.

Moreover in order to help the author to specify the temporal organisation of documents, Smily provides her/him with an execution report displayed through the timeline view. This execution report has the particularity to be associated with additional information directly linked with the hierarchical view which helps the author to understand why such execution occurs.

Smily is built using Mikado: a scalable toolkit for building multimedia authoring environment. This allows us a fast development of the application and adaptability capabilities, particularly to follow the advancements of the second version of Smil which is being developed. Moreover, thanks to Mikado we will be able to provide the authors with interesting temporal checking capabilities as far as the semantics of relative offsets in Smil will be clearly defined.

The current state of Smily allows us to write Smil documents as explained in this paper, only the timeline view is not yet completely operational but should be usable before Summer'99.

Future works in Smily will evolve in the two following directions:

- Integrating a larger subset of the Smil standard. In addition with relative offsets, we are very interested by the switch element which allows the document to adapt itself during its presentation.

- Extending the timeline view in such a way that it will be also an editing view. Our idea is different from the one experimented in GRiNS which tries to show every possible execution at the same time. We prefer first to display one possible execution and second to give authors access to the others by direct manipulations on the current display. Some conclusive experiments have yet been made in this direction in the Madeus authoring tool [4].

References

[1] Bulterman D.C.A., Hardman L., Jansen J., Mullender K.S., Rutledge L., "*A GRaphical INterface for creating and playing SMIL documents*", proc. WWW-7*,* pp. 519-529, Computer Networks and ISDN Systems 30, Brisbane, Australia, April 1998.

[2] GRiNS, on line : http://www.cwi.nl/GRiNS/.

[3] ISO/IEC JTC1/SC18/WG8 N1920, *Information Technology: Hypermedia/Time-based Structuring Language (HyTime), Second edition,* ISO/IEC, août 1997. http://www.ornl.gov/sgml/wg8/docs/n1920/html/n1920.html.

[4] Jourdan M., Roisin C., Tardif L., "*Multiviews Interfaces for Multimedia Authoring Environments*"", Proc. of the 5th Conference on Multimedia Modelling, Lausanne, 12-15 October 1998.

[5] Jourdan M., Layaïda N., Roisin C., Sabry-ismail L., Tardif L., "*Madeus, an Authoring Environment for Interactive Multimedia Documents*", 6th ACM Multimedia'98, Bristol, 12-16 septembre 1998.

[6] Jourdan M , Roisin C, Tardif L., "*A Scalable Toolkit for Designing Multimedia Authoring Environments*", submitted to the special issue on multimedia authoring and presentation techniques, for the ACM Multimedia Systems journal, vol. , 1999.

[7] JustSMIL, "http://www.justsmil.com/".

[8] Price R., "*MHEG: An Introduction to the Future International Standard for Hypermedia Object Interchange*", Proceedings of the First ACM Conference on Multimedia, pp. 121-128, ACM Press, Anaheim, Californie, August 1993.

[9] Real Netwoks G2, on line : http://www.real.com/g2/index.html.

[10] Sannella M., Malorney J., Freeman-Benson B., Borning A, "*MultiWay versus OneWay Constraints in User Interfaces : Experience with the DeltaBlue Algorithm, Software practice and Experience*", vol. Vol. 32, num. 5, pp. 529-566, May 1993.

[11] Smil Composer, on line : http://www.sausage.com/supertoolz/toolz/st smil.html.

[12] Smil Wizard in G2 Authoring Kit, http://www.real.com/products/tools/authkit/ index.html.

[13] Soja Barbizon, on line : http://www.helio. org/.

[14] Synchronized Multimedia Working Group Charter, *http://www.w3.org/AudioVideo/ Group/symm-wg-charter#1*.

[15] Synchronized Multimedia, http://www.w3. org/AudioVideo.

[16] T.A.G, on line : http://tag.digital-ren.com.

[17] Veon , on line : http://www.veon.com/.

[18] W3C Recommendation, "*Synchronized Multimedia Integration Language (SMIL) 1.0 Specification*", http://www.w3.org/TR/ REC-smil, 15-June 1998.