IMPROVING SMIL DOCUMENTS WITH NCM FACILITIES

L.M. RODRIGUES, R.F. RODRIGUES, D.C. MUCHALUAT-SAADE, L.F.G. SOARES
{leandro, rogerio, debora}@telemidia.puc-rio.br; lfgs@inf.puc-rio.br

Laboratório TeleMídia, Depto. de Informática, PUC-Rio
R. Marquês de São Vicente, 225 – 22453-900 – Rio de Janeiro, Brasil

The use of the World-Wide Web to distribute multimedia documents with temporal and spatial relations is a research topic that has recently gained a lot of attention. Following the same proposal of simplicity found in HTML, the WWW consortium has published the SMIL recommendation for declarative authoring of multimedia documents. However, the SMIL simplicity leads to some limitations in the authoring process. In order to overcome these limitations, this work presents in details the integration of the SMIL language with NCM (Nested Context Model), an event-driven and object-oriented hypermedia model. The integration allows one to refine and manipulate SMIL documents in a Web-integrated hypermedia system with version control and high-level authoring tools.

1 Introduction

The use of the World-Wide Web (WWW) to distribute multimedia documents with temporal and spatial relations, besides the usual hypermedia navigation relations, is a research topic that has recently gained a lot of attention. As HTML, the main language used to create Web documents, does not support the definition of such synchronization relations, other proposals have been arisen to achieve this goal.

One of these proposals is a declarative language called SMIL (Synchronized Multimedia Integration Language) [11], adopted as a recommendation by the WWW Consortium. SMIL allows defining multimedia presentations on the Web in a simple way. However, as it happens with HTML, SMIL simplicity leads to some limitations in the authoring process.

The conceptual model of the HyperProp hypermedia system, *Nested Context Model* – NCM [8], is an object-oriented and event-driven model that allows logically structuring a document using compositions. Relationships among components are modeled by compositions and links. Compositions define structural relationships among nodes, while links define synchronous and asynchronous relationships among them.

Aiming to overcome HTML limitations, HyperProp has been integrated with the WWW, offering its features to manipulate Web documents [7]. Following a similar goal, this work presents the integration of SMIL with NCM.

The paper is organized as follows. Section 2 briefly describes the SMIL language. Section 3 presents some SMIL limitations and how they can be overcome by its integration with NCM. Section 4 presents some NCM relevant concepts for

the paper scope. Section 5 describes the mapping between the two models. Section 6 comments related work and Section 7 is reserved to final remarks.

2 SMIL

The SMIL language offers a declarative form to specify the presentation of multimedia documents. A document is divided in two sections: head and body. The *head* contains general information about the document and the specification of the presentation spatial layout. The *body* contains the definition of objects and their temporal relations, as well as the specification of anchors and navigation links.

The spatial layout contains the definition of presentation window regions where objects will be shown. There is a special region that represents the whole window where the SMIL document will be presented. All other regions are defined in relation to this region.

SMIL basic objects, called *media objects*, can be of several types, such as text, image, video and audio. The object does not have the content of the associated data, but a description that includes a reference (URI) to the content itself. Media objects can also have *clip-begin* and *clip-end* attributes, which limit the object content to be presented.

SMIL uses the concept of composition to structure the document presentation. A composition can contain media objects and other compositions, establishing the way its components will be presented. There are two composition types, parallel (*par*) and sequential (*seq*). In a *parallel* composition, components are simultaneously presented, while in a *sequential* composition they are presented in sequence. The document body implicitly defines a sequential composition.

Every media object or composition has a duration that establishes how long its presentation will last. Continuous media objects, such as audio and video, have an implicit duration derived from its content, while discrete media objects, such as text and image, have a null implicit duration. The implicit duration of a composition corresponds to the result of the duration of its components. Besides the implicit duration, every element can have its duration explicitly defined by the author, through an optional attribute called *duration* (*dur*). The explicit duration of an element can be *indefinite*, indicating that the end of its presentation will coincide with the end of the presentation of the composition that contains it.

SMIL elements have two other optional temporal attributes, called *begin* and *end*. If an element with a defined begin/end attribute is contained in a parallel composition or if it is the first component of a sequential composition, it starts/finishes its presentation after the specified time interval since the beginning of the composition. If the element is a component, other than the first, of a sequential composition, the attribute value specifies a time interval to start/finish its presentation, in relation to the end of the previous component. In both parallel and sequential compositions, begin and end attributes can also be specified in relation to the beginning or end of any other element contained in the composition.

Additionally, parallel compositions have an attribute called *endsync*. It specifies the end of the composition in relation to the end of one of its components, which can be the first one to finish, the last one to finish or any specific component.

The same element can be presented several times in sequence. The number of presentations is specified in the *repeat* attribute, whose value can be an integer or *indefinite*. In the latter case, the element presentation will be repeated until the end of the composition that contains it.

SMIL also allows the specification of temporal and spatial sub-regions (anchors) in media objects and the definition of hyperlinks between them. Links can be defined not only between sub-regions of an object, but also among elements (whole media objects or compositions), or associations of these possibilities. Links can also be defined from an element or sub-region to a whole document or a specific element in another document. A SMIL link has an attribute called *show*, which defines the behavior of the link source object when a link is fired. The attribute value can be: *replace*, which specifies that the presentation of the source object will be replaced by the presentation of the target object; *new*, which specifies that the presentation of the target object will happen in another context, without affecting the source object; and *pause*, which specifies that the presentation of the source object will be suspended and resumed after the end of the presentation of the target object, which is done in another context.

Finally, SMIL provides means to define alternatives for a document presentation, depending on factors related to the system or to the user (e.g. language, bandwidth, etc.). In order to define multiple behavior options for a presentation, the language provides the *switch* element. This element allows the author to specify a group of alternative elements, from which at most one will be chosen, according to one or more tests that are made during runtime. These alternative elements are declared in order, and the first one that satisfies its tests is chosen to be presented.

3 Benefits of the integration of the SMIL language with NCM

As previously commented, SMIL simplicity leads to some limitations in the authoring process. Among these limitations, we can mention:

- SMIL offers two types of compound elements to define temporal relations, which are parallel and sequential compositions. The logical structure of a document is necessarily the presentation structure built by a hierarchy of nested compositions. Using this temporal model, changes in a document specification may result in a completely new structure for the document.
- SMIL does not provide data reuse or presentation structure reuse. Structures can only be reused by making copies, which makes maintenance tasks difficult. The language only provides reuse of media object content, identified by a URI, and reuse of presentation layouts inside the same document.

- Asynchronous relations in a SMIL document are always *1:1* links triggered by user selection, while all synchronous relations are specified by parallel and sequential compositions and element attributes (begin, end, dur, repeat, etc.). Thus, more complex relations among objects cannot be defined (e.g. "if the presentation of object *A* is in paused state when the presentation of anchor α of object *B* finishes, then restart the presentation of object *A* and increase the presentation rate of object *B* by 5%").
- There is no support for specifying behavior changes that may happen during an object presentation (e.g. "increase the audio volume to *X db*, *t* seconds after the beginning of its presentation").
- SMIL does not offer much flexibility for specifying presentation and preparation duration of objects, for example, allowing the definition of intervals for this duration. This flexibility would allow a presentation control system to adjust time in order to improve the presentation quality and reduce the chance of having temporal inconsistencies [1,4].

The integration of the SMIL language with NCM aims to overcome these limitations. Among the integration benefits, we can highlight:

- Using compositions not only for structuring a document presentation, but also for other logical structuring of its components, for example, context, derivation and task relations [9].
- The possibility of defining temporal synchronization among components using also links, allowing users to change a presentation specification without modifying the document global structure.
- The possibility of reusing data and presentation structure. In NCM, as compositions can be contained in different compositions, it is possible to reuse parts of a document structure (objects and relations) in the same document or in other documents, besides its data items.
- Support for the definition of *m:n* relationships among components of a SMIL document using NCM links, combining user interaction, presentation events and also the manipulation of node attributes.
- The possibility of specifying behavior changes that may happen during an object presentation.
- More flexibility in the specification of temporal behavior of objects, allowing possible adjustments during its presentation, for example, due to network delays.
- Support for version control and cooperative work.
- Using the HyperProp authoring tool, which offers a graphical interface to create and edit structured documents.
- Using the HyperProp formatter (analogous to a SMIL player) to present SMIL documents. The current version of the HyperProp formatter does not

implement, but foresees, support for adjustments during runtime in order to guarantee a better presentation quality according to a user platform.

4 Nested Context Model

NCM is based on the usual concepts of nodes and links. In NCM, a *node* has as main attributes its content, an anchor ordered list and a descriptor. The *content* identifies a set of *information units*. An *anchor* identifies a marked subset of the content. The exact notion of information unit is part of the definition of node class. For example, an information unit of a video node could be a frame, while an information unit of a text node could be a character or a word. Every NCM node has an anchor called λ *anchor*, which identifies the whole content of the node. The *descriptor* contains information determining how the node should be presented in time and space.

NCM distinguishes two basic classes of nodes, called content node and composite node. Intuitively, a *content node* contains data whose internal structure is application dependent, modeling traditional hypermedia nodes. Content nodes may be specialized in other classes (text, graphic, audio, video, etc.) as required by applications. For example, a specialization of the text node class, called HTML node, was defined in [7] to integrate HTML pages with NCM documents. The logical structuring of a document is made by introducing the composition concept. In NCM, the content of a *composite node* is a list of links and nodes, which may be content or composite nodes, recursively. An important restriction is made: a node cannot be recursively contained in itself.

As a node can be contained in different compositions and composite nodes can be nested to any depth, it is necessary to introduce the concept of perspective. Intuitively, the *perspective* of a node identifies through which sequence of nested composite nodes a given node instance is being observed.

The basic synchronization unit in NCM is an event. As stated in [5], an *event* is an occurrence in time that can be instantaneous or can occur over some time period. NCM defines three types of events: *presentation event*, which is defined by the presentation of an anchor of a node in a given perspective; *selection event*, which is defined by the selection of an anchor of a node in a given perspective; and *attribution event*, which is defined by a change in the value of an attribute of a node in a given perspective.

In NCM, an event can be in one of the following states: *sleeping, preparing, prepared, occurring, paused* and *aborted*. Every event has an associated attribute, called *occurrences*, which counts how many times it transits from *occurring* to *prepared* state during a document presentation. Presentation events also have an attribute named *repeat*, which indicates how many times the event should sequentially be presented. As in SMIL, this attribute can contain a finite value or the *indefinite* value. Moreover, presentation events contain five other attributes:

minimum duration, optimum duration, maximum duration, expected duration and elastic cost. The *minimum* and *maximum duration* together define the valid interval for the event presentation duration, while the *optimum duration* specifies the duration that leads to the best quality of presentation. *Elastic cost* is a function that determines the cost if the best quality of presentation is not chosen. The *expected duration* is initialized with the optimum duration. Based on synchronization procedures like the ones described in [1,4], a formatter can change the expected duration to achieve a better presentation quality for the document as a whole, optimizing the global cost. The NCM event state machine is presented in Figure 1.

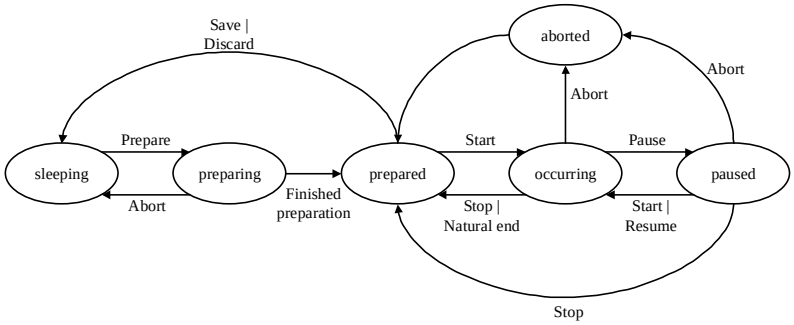

Figure 1 – NCM event state machine

A presentation event can change from *occurring* to *prepared* in two situations: as a consequence of the natural end of the presentation duration, or due to an action that stops the event. As in SMIL, an event duration can be intrinsic or specified by the author. Furthermore, the transition from *aborted* to *prepared* always occurs automatically. The actions that generate state transitions will be explained later in this section.

A *link* represents *m:n* relationships among objects. It is composed by *m* source end points, *n* target end points and a meeting point. *End points* define events, while the *meeting point* contains the description of the relationship among them. Since a node may pertain to more than one composite node, the source and target end points of a link are identified by tuples of the form $<(N_k,...,N_1), \alpha, type>$, such that: $(N_k,...,N_1)$ is the nested node sequence to reach N_1, in relation to the composition that contains the link; α is the identification of an anchor or an attribute of N_1; and *type* specifies the event associated to α (presentation, selection or attribution). N_1 is called an *anchor node* of the link.

The meeting point contains a condition and an action. Conditions are related to source end points, while actions are operations that should be applied to target end points. When a condition is satisfied, its associated action is fired.

Meeting point *condition* evaluates boolean values and can be binary simple or compound. Every *binary simple condition* is expressed by two unary conditions: a *previous condition*, to be satisfied immediately before the condition evaluation instant, and a *current condition*, to be satisfied at the condition evaluation instant. A

binary simple condition is satisfied when both unary conditions are satisfied. A unary condition is a comparison concerning event states, event attributes or node attributes. It can also receive the *true* value, if it is not relevant in the binary simple condition evaluation. Comparison operators used in the evaluation of unary conditions are: $=$, $\neq$, $<$, $\leq$, $>$ and $\geq$. A *compound condition* consists of a logical expression involving other conditions, simple or compound, and based on the operators $\wedge$ (and), $\vee$ (or), $\neg$ (not) and $\oplus$ (delay). The logical operators $\wedge$, $\vee$ and $\neg$ have the usual meanings of the propositional logic. An expression $(C \oplus d)$ is satisfied at a time t if condition C was satisfied at time $[t - d]$, where $d \in \Re$. When d is non-negative, the meeting point defines a traditional causal relationship. When d is negative, satisfied conditions imply actions that should be fired in the past. In these cases, the meeting point semantically specifies constraint relations among events. Like the event duration, d is specified with minimum, optimum, expected and maximum values and an elastic cost.

Meeting point *actions* can also be simple or compound. A *simple action* is executed over an event contained in the destination end point set. Simple actions applied on end points that define presentation events are described in Table 1. The model also defines simple actions over attribution events, but they are not relevant in this paper.

A compound action is defined by an expression based on the operators | (parallel) and $\rightarrow$ (sequential), specifying the execution order of each action of the expression. Moreover, every simple or compound action has an optional attribute called *wait*. This attribute determines the time that must be waited before executing the action. Like the event duration and the condition delay operator, the *wait* attribute is specified as a tuple <minimum, optimum, maximum, expected, elastic cost>.

A hypermedia document model should permit the definition of the expected behavior of each node when it is presented, specifying its presentation layout and how the presentation should be modified in time. NCM defines this information separately from the associated node, which allows a better reuse of objects. For example, using distinct layouts, one can define different presentations for the same media object. A string can be presented as text, or it can be synthesized as audio, depending on the layout specification. Note that as different layouts can lead to different event duration intervals, different qualities of presentation and different platform requirements, the definition of all these issues should also be part of the layout and not part of the media object (node).

In NCM, spatial and temporal specification of a node and behavior changes during its presentation are defined in the descriptor object. The presentation of a node to the final user (document reader) is done by the association of this node with a descriptor. The model also allows a descriptor to be specified by the reader during the presentation. If it does not happen, the system will use a descriptor previously defined by the author. In this case, the descriptor can be specified in the link (action parameter), in the composition that contains the node or as an attribute of the node.

Action	Description
Prepare(E, d)	If event *E* is in the sleeping state, it goes to the preparing state; otherwise, there is no transition. The action can optionally specify a descriptor to be associated to the anchor node of the target end point that defines *E*. At the end of the procedure, the repeat attribute of *E* is initialized with the value specified in the associated descriptor. If there is no specified value, the attribute is set to one (default value).
Start(E, d, n)	If event *E* is in the sleeping, preparing, prepared or paused state, it goes to the occurring state through the state machine (see Figure 1), starting its presentation from the beginning. Otherwise, nothing happens. As in the prepare action, the start action can optionally specify a descriptor to be associated to the anchor node of the target end point that defines *E*. This action also has another optional parameter *n* to initialize the *repeat* attribute of *E*. If this parameter is not specified, the *repeat* attribute is initialized with the value declared in the associated descriptor.
Stop(E)	If event *E* is in the occurring or suspended state, it goes to the prepared state; otherwise, there is no transition. This action increases by one unit the *occurrences* attribute of *E* and decreases by one unit the *repeat* attribute of *E*. If the value of *repeat* is still greater than zero after being decreased, a new presentation of *E* is automatically started.
Pause(E)	If event *E* is in the occurring state, it goes to the paused state, otherwise, nothing happens.
Resume(E)	If event *E* is in the paused state, it goes back to the occurring state, otherwise, nothing happens. The presentation of *E* is resumed from the last information unit presented before the event has been paused.
Abort(E)	If event *E* is in the occurring or paused state, it goes to the aborted state and immediately after it goes to the prepared state. The *occurrences* attribute of *E* is not increased and the *repeat* attribute is set to zero. If *E* is in the preparing state it goes to the sleeping state. If *E* is in any other state, nothing happens.

Table 1 – NCM simple actions applied on presentation events

This is the respective priority order used to choose the descriptor. If there is no descriptor specified, a default descriptor of the node class is used.

A *descriptor* has as main attributes the node exhibition/edition method and a collection of event descriptions. NCM allows descriptor attributes to be extended with other attributes depending on the node class.

The *node exhibition/edition method* can identify any program or can be a method implemented in the descriptor itself. Each *event description* consists in a tuple <α, *type*, *dur*, *rep*, *oper*>. The first parameter (α) identifies an anchor of the node to which the descriptor will be associated. *Type* specifies the event type associated to the anchor (presentation, selection or attribution). *Dur* is a tuple <d_{min}, d_{opt}, d_{max}, f_c>, which defines values to initialize, respectively, the minimum duration, optimum duration, maximum duration and elastic cost attributes of the event. *Rep* specifies the number of repeated exhibitions for the event. This parameter initializes

the *repeat* attribute value of the event, if it is not initialized by a link action. *Dur* and *Rep* are only used for presentation events. *Oper* specifies an ordered sequence of operations, which are executed whenever the event state changes. Each operation is composed by a condition and an action, which are similar to link meeting point conditions and actions. The only difference is that their entry scope are limited to events and attributes of the node associated to the descriptor, or to attributes defined in the descriptor itself.

5 Conversion of SMIL documents into NCM documents

5.1 Media objects and anchors

SMIL media objects are converted into content nodes and descriptors in NCM. A content node is created from the *type* and *src* attributes declared in the media object. Its class corresponds to the media object type (text, image, etc.), and the value of its *content* attribute is assigned with the reference (URI) stored in the *src* attribute of the media object. Two or more distinct media objects with the same values for their *type* and *src* correspond to the same NCM content node, inserted in different perspectives and with distinct descriptors. The information stored in descriptors is discussed in Section 5.2.

For each media object anchor, an equivalent NCM anchor is created and inserted in the anchor list of the node. If the media object has *clip-begin* and *clip-end* attributes, an additional anchor is created and inserted in the anchor list of the node, defining this sub-region of the node content. In this case, all synchronization relationships associated to this node will make reference to this anchor, instead of the λ anchor of the node (see Section 4). In this paper, we will not consider media objects with clip attributes, without generality loss.

5.2 Spatial layout, object duration and repetitions

Given the region attribute of a media object, the information about spatial localization is extracted from the SMIL document layout and stored in the descriptor. In order to store the object duration, an entry is created in the event description list associated to the λ anchor of the node (see Section 4). The event type is defined as presentation. The minimum, optimal and maximum duration are all set to the same value (the object duration), and the cost function is set to a null value, since SMIL does not provide flexibility in the duration declaration, as mentioned in Section 3. The *rep* parameter is set to the value of the *repeat* attribute specified in the media object, if it is specified.

Descriptors are also created for composite nodes. NCM compositions will represent SMIL parallel and sequential compositions, as will be seen in Sections 5.3 and 5.4. In this case, descriptors will be important to store the duration, the number of repetitions and additional node attributes.

5.3 Parallel Compositions

A SMIL parallel composition is represented in NCM as a composite node containing other content or composite nodes corresponding to the components of the SMIL composition. The NCM composition also has links to express synchronization among its components. The generic structure of an NCM composite node that represents a SMIL parallel composition is presented in Figure 2, while its links are described in Table 2.

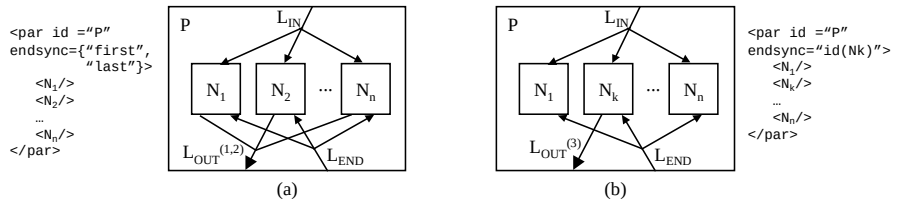

Figure 2 – Representation of SMIL parallel compositions in NCM. (a) Compositions ending by the first or the last component. (b) Compositions ending by a specific component.

Link L_{IN} is responsible for the beginning of the parallel presentation of the nodes contained in composition *P* as soon as the presentation of *P* starts.

Links $L_{OUT}^{(1,2,3)}$ are responsible for the end of the presentation of *P*. The component of *P* responsible for the end of its presentation depends on the *endsync* attribute of the SMIL parallel composition. If the composition is finished by the last component (link $L_{OUT}^{(1)}$), each time the presentation of a node contained in *P* finishes, the link meeting point verifies if all nodes have already finished their presentations (including possible repetitions). If the composition is finished by the first component (link $L_{OUT}^{(2)}$), the first component that finishes its presentation fires the link. Finally, if *P* is finished by a specific component (link $L_{OUT}^{(3)}$), the link only tests the component presentation event.

When *P* leaves the *occurring* state, it must abort the presentation of each node it contains, if it is still occurring. This action is performed by link L_{END}, which is fired when the presentation event state of node *P* changes from *occurring* to *prepared*, or from *occurring* to *aborted*. The former transition occurs as a consequence of the *stop* action of a link (e.g., $L_{OUT}^{(1,2,3)}$) or the expiration of the duration specified in node *P* descriptor, if an explicit duration is defined for this node. The latter transition occurs as a consequence of the *abort* action over the presentation event of *P*, caused by a link contained in another composition in which *P* is contained.

The beginning of a component may be defined explicitly in relation to the parallel composition. Supposing that a component N_k of the parallel composition *P* presented in Figure 2 has an explicit begin of *t* units of time in relation to the parallel composition itself, a new link $L_{IN,k}$ is created in order to warrant the specified delay to start N_k presentation. This link has a source end point defined in *P* and a target end point defined in N_k. Besides, link L_{IN} is modified so that it does not

Link	Source Events	Target Events	Meeting Point
L_{IN}	E_P: <(P), λ, *pres*>	E_1: <(N_1), λ, *pres*> E_2: <(N_2), λ, *pres*> … E_n: <(N_n), λ, *pres*>	***Condition***: <st(E_P)=prepared, st(E_P)=occurring> ***Action***: start(E_1) | start(E_2) | … | start(E_n)
$L_{OUT}^{(1)}$	E_1: <(N_1), λ, *pres*> E_2: <(N_2), λ, *pres*> … E_n: <(N_n), λ, *pres*>	E_P: <(P), λ, *pres*>	***Condition***: (<st(E_1)=occurring, st(E_1)=prepared> ∨ <st(E_2)=occurring, st(E_2)=prepared> ∨ … ∨ <st(E_n)=occurring, st(E_n)=prepared>) ∧ (<true, rep(E_1)=0> ∧ <true, rep(E_2)=0> ∧ … ∧ <true, rep(E_n)=0>) ***Action***: stop(E_P)
$L_{OUT}^{(2)}$	E_1: <(N_1), λ, *pres*> E_2: <(N_2), λ, *pres*> … E_n: <(N_n), λ, *pres*>	E_P: <(P), λ, *pres*>	***Condition***: (<st(E_1)=occurring, st(E_1)=prepared> ∧ <true, rep(E_1)=0>) ∨ (<st(E_2)=occurring, st(E_2)=prepared> ∧ <true, rep(E_2)=0>) ∨ … ∨ (<st(E_n)=occurring, st(E_n)=prepared> ∧ <true, rep(E_n)=0>) ***Action***: stop(E_P)
$L_{OUT}^{(3)}$	E_k: <(N_k), λ, *pres*>	E_P: <(P), λ, *pres*>	***Condition***: <st(E_k)=occurring, st(E_k)=prepared> ∧ <true, rep(E_k)=0> ***Action***: stop(E_P)
L_{END}	E_P: <(P), λ, *pres*>	E_1: <(N_1), λ, *pres*> E_2: <(N_2), λ, *pres*> … E_n: <(N_n), λ, *pres*>	***Condition***: <st(E_P)=occurring, st(E_P)=prepared> ∨ <st(E_P)=occurring, st(E_P)=aborted> ***Action***: abort(E_1) | abort(E_2) | … | abort(E_n)

pres = presentation
$L_{OUT}^{(1)}$ - compositions finished by the last component (endsync = "last" – default)
$L_{OUT}^{(2)}$ - compositions finished by the first component (endsync = "first")
$L_{OUT}^{(3)}$ - compositions finished by a specific component N_k (endsync = "id(N_k)")

Table 2 – Specification of links in parallel compositions. In meeting point conditions, we adopted the following representation conventions: st(*E*) evaluates the state of event *E*, and rep(*E*) evaluates the *repeat* attribute of event *E*.

start N_k presentation, but just prepares it. The reason to prepare N_k presentation is due to the need to initiate the *repeat* event attribute, in order to avoid firing link $L_{OUT}^{(1)}$ before N_k presentation. Table 3 presents the descriptions of the modified link (L_{IN}) and the new link ($L_{IN,k}$).

Link $L_{IN,k}$ has a condition that verifies if the presentation event of composition *P* remains in the *occurring* state after the specified delay. The state can have changed for three reasons: i) the end of the composition was determined by a node, other than the last, that has finished its presentation before the specified delay; ii) composition *P* has finished its presentation as a consequence of the definition of an

explicit duration for the composition; iii) an *abort*/*stop* action has been performed over the presentation event of *P*.

The explicit begin of a component can also be given in relation to another node of the composition. In this case, the new link created will have this node as its source end point. Table 3 describes the new link ($L_{m,k}$) created to express the explicit begin of a node N_k in relation to another node N_m of the same composition.

Link	Source Events	Target Events	Meeting Point
$\mathbf{L_{IN}}$	$\mathbf{E_P}$: <(P), λ, *pres*>	$\mathbf{E_1}$: <(N_1), λ, *pres*> … $\mathbf{E_k}$: <(N_k), λ, *pres*> … $\mathbf{E_n}$: <(N_n), λ, *pres*>	***Condition***: <st(E_P)=prepared, st(E_P)=occurring> ***Action***: start(E_1) | start (E_2) | … | prepare(E_k) | … | start(E_n)
$\mathbf{L_{IN,k}}$	$\mathbf{E_P}$: <(P), λ, *pres*>	$\mathbf{E_k}$: <(N_k), λ, *pres*>	***Condition***: (<st(E_P)=prepared, st(E_P)=occurring> ⊕ t) ∧ <true, st(E_P)=occurring> ***Action***: start(E_k)
$\mathbf{L_{m,k}}$	$\mathbf{E_m}$: <(N_m), λ, *pres*> $\mathbf{E_P}$: <(P), λ, *pres*>	$\mathbf{E_k}$: <(N_k), λ, *pres*>	***Condition***: (<st(E_m)=prepared, st(E_m)=occurring> ⊕ t) ∧ <true, st(E_P)=occurring> ***Action***: start(E_k)

Table 3 – Specification of links for explicit begin in parallel compositions

5.4 Sequential Compositions

Like a parallel composition, a SMIL sequential composition is represented in NCM as a composite node containing other content or composite nodes associated with the components of the SMIL composition, and links to express synchronization among its components. The SMIL document body is represented in the same way. The generic structure of an NCM composite node that represents a SMIL sequential composition is presented in Figure 3, while its links are described in Table 4.

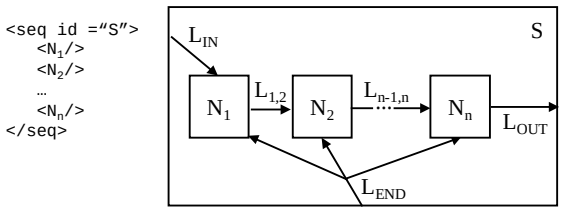

Figure 3 – Representation of SMIL sequential compositions in NCM

Link L_{IN} starts the presentation of the node corresponding to the first component of the SMIL sequential composition, as soon as the presentation of composition *S* starts. The beginning of the presentation of each subsequent node is

caused by the end of the presentation of the previous node (considering possible repetitions). These relations are established by links $L_{i-1,i}$, with i ∈ [2,n]. Finally, link L_{OUT} finishes the presentation of *S* after the last node presentation.

When the presentation event of composition *S* leaves the *occurring* state, the presentation of all its component nodes must be aborted. These actions are performed by link L_{END}, and typically occur when the presentation event of *S* is aborted, or when it finishes as a consequence of the definition of an explicit duration.

Link	Source Events	Target Events	Meeting Point
$\mathbf{L_{IN}}$	$\mathbf{E_S}$: <(S), λ, *pres*>	$\mathbf{E_1}$: <(N_1), λ, *pres*>	***Condition***: <st(E_S)=prepared, st(E_S)=occurring> ***Action***: start(E_1)
$\mathbf{L_{i-1,i}}$ (i∈[2,n])	$\mathbf{E_{i-1}}$: <(N_{i-1}), λ, *pres*>	$\mathbf{E_i}$: <(N_i), λ, *pres*>	***Condition***: <st(E_{i-1})=occurring, st(E_{i-1})=prepared> ∧ <true, rep(E_{i-1})=0> ***Action***: start(E_i)
$\mathbf{L_{OUT}}$	$\mathbf{E_n}$: <(N_n), λ, *pres*>	$\mathbf{E_S}$: <(S), λ, *pres*>	***Condition***: <st(E_n)=occurring, st(E_n)=prepared> ∧ <true, rep(E_n)=0> ***Action***: stop(E_S)
$\mathbf{L_{END}}$	$\mathbf{E_S}$: <(S), λ, *pres*>	$\mathbf{E_1}$: <(N_1), λ, *pres*> $\mathbf{E_2}$: <(N_2), λ, *pres*> … $\mathbf{E_n}$: <(N_n), λ, *pres*>	***Condition***: <st(E_S)=occurring, st(E_S)=prepared> ∨ <st(E_S)=occurring, st(E_S)=aborted> ***Action***: abort(E_1) | abort(E_2) | … | abort(E_n)

Table 4 – Specification of links in sequential compositions

The beginning of a component presentation may be defined explicitly in relation to the end of the previous component, specifying a delay. When the explicit specification is made in the first component, link L_{IN} is slightly modified. If the specification is made in component *k*, link $L_{k-1,k}$ is modified. Table 5 presents the description of the modified links for the specified cases.

5.5 Navigation links

SMIL allows the definition of traditional hypermedia links activated by user interaction. These links are always *1:1*, that is, with a source anchor and a target anchor. The anchors are defined in the link itself.

SMIL links are converted into NCM links. If the *show* attribute of the SMIL link is *replace*, the NCM link action stops the source document presentation before starting the target object presentation. If it is *pause*, the action pauses the source object presentation, which is resumed by another link after the end of the target

Link	Source Events	Target Events	Meeting Point
$\mathbf{L_{IN}}$	$\mathbf{E_S}$: <(S), λ, *pres*>	$\mathbf{E_1}$: <(N_1), λ, *pres*>	***Condition***: (<st(E_S)=prepared, st(E_S)=occurring> ⊕ t) ∧ <true, st(E_S)=occurring> ***Action***: start(E_1)
$\mathbf{L_{k-1,k}}$	$\mathbf{E_{k-1}}$: <(N_{k-1}), λ, *pres*> $\mathbf{E_S}$: <(S), λ, *pres*>	$\mathbf{E_k}$: <(N_k), λ, *pres*>	***Condition***: ((<st(E_{k-1})=occurring, st(E_{k-1})=prepared> ∧ <true, rep(E_{k-1})=0>) ⊕ t) ∧ <true, st(E_S)=occurring> ***Action***: start(E_k)

Table 5 – Specification of links for explicit begin in sequential compositions

object presentation. If it is *new*, the action simply starts the target object presentation, without affecting the source document presentation.

For example, consider a SMIL link from an element of the document "origin.smil" to an element of the document "destination.smil", with the *show* attribute equal to *replace*. Let S_1 and S_2 be the NCM compositions that represent the body of these documents, respectively, and let N_1 and N_2 be the NCM nodes corresponding to the source and the target SMIL elements, respectively. Link *L*, created in NCM to represent this SMIL link, is specified in Table 6. Its source end point corresponds to the selection event of N_1, and its target end points correspond to the presentation events of S_1 and N_2. The link action stops the presentation of S_1, and then starts the presentation of N_2. The node lists (S_1,…, N_1) and (S_2,…, N_2) specify the perspectives of N_1 and N_2 in the context given by the conversion of the documents "origin.smil" and "destination.smil", respectively. The link is inserted into a third composition that also contains nodes S_1 and S_2.

Link	Source Events	Target Events	Meeting Point
L	$\mathbf{E_S}$: <(S_1,…, N_1), λ, *sel*>	$\mathbf{E_1}$: <(S_1), λ, *pres*> $\mathbf{E_2}$: <(S_2,…, N_2), λ, *pres*>	***Condition***: <st(E_S)=prepared, st(E_S)=occurring> ***Action***: stop(E_1) → start (E_2)

sel = selection

Table 6 – Specification of a traditional hypermedia navigation link

5.6 Alternative presentations

As previously mentioned, SMIL provides the *switch* element, which allows the definition of alternatives for media object and composition presentation, according to system characteristics or user preferences.

NCM does not directly offer this kind of feature. Instead, it proposes that users create structures and virtual links with queries that define the presentation behavior according to the context. However, virtual entities and query languages are not implemented in the system yet, and are left for future work. Nevertheless, it is

possible to simulate the specification of alternative presentations through composite nodes, links and operation lists in event descriptions.

In order to convert a *switch* element to an NCM object, a composite node is created containing nodes that correspond to the components of the *switch* element. Associated with this composite node, there is a descriptor with an attribute *attr*, over which the test must be done. Figure 4 describes the generic structure of a composition that represents alternative components. Table 7 presents the description of the links shown in the figure. The attribute *attr* is declared in attribution event A_S.

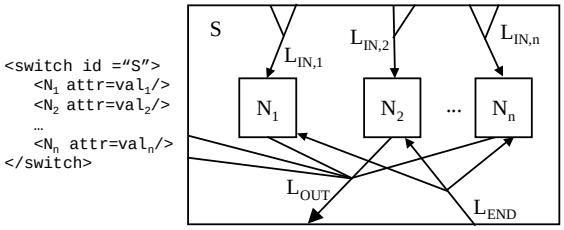

Figure 4 – Generic structure of a switch element as an NCM composition

Link	Source Events	Target Events	Meeting Point
$\mathbf{L_{IN,i}}$ (i∈ [1,n])	$\mathbf{E_S}$: <(S), λ, *pres*> $\mathbf{A_S}$: <(S), *attr*, *attrib*>	$\mathbf{E_i}$: <(N_i), λ, *pres*>	***Condition***: <st(E_S)=prepared, st(E_S)=occurring> ∧ <true, at(A_S)=val_i> ***Action***: start(E_i)
$\mathbf{L_{OUT}}$	$\mathbf{E_1}$: <(N_1), λ, *pres*> $\mathbf{E_2}$: <(N_2), λ, *pres*> … $\mathbf{E_n}$: <(N_n), λ, *pres*> $\mathbf{E_S}$: <(S), λ, *pres*> $\mathbf{A_S}$: <(S), *attr*, *attrib*>	$\mathbf{E_S}$: <(S), λ, *pres*>	***Condition***: (<st(E_1)=occurring, st(E_1)=prepared> ∧ <true, rep(E_1)=0>) ∨ (<st(E_2)=occurring, st(E_2)=prepared> ∧ <true, rep(E_2)=0>) ∨ … ∨ (<st(E_n)=occurring, st(E_n)=prepared> ∧ <true, rep(E_n)=0>) ∨ (<st(E_S)=prepared, st(E_S)=occurring> ∧ (¬<true, at(A_S)=val_1> ∧ ¬<true, at(A_S)=val_2> ∧ … ∧ ¬<true, at(A_S)=val_n>)) ***Action***: stop(E_S)
$\mathbf{L_{END}}$	$\mathbf{E_S}$: <(S), λ, *pres*>	$\mathbf{E_1}$: <(N_1), λ, *pres*> $\mathbf{E_2}$: <(N_2), λ, *pres*> … $\mathbf{E_n}$: <(N_n), λ, *pres*>	***Condition***: <st(E_S)=occurring, st(E_S)=aborted> ∨ <st(E_S)=occurring, st(E_S)=prepared> ***Action***: abort(E_1) | abort(E_2) |…| abort(E_n)

attrib = attribution

Table 7 – Specification of links in switch compositions

Links $L_{IN,i}$, with $i \in [1,n]$, start the presentation of the node N_i whose value val_i is equal to the value of the *attr* attribute of the composition descriptor. Link L_{OUT}

stops the presentation of *S* when the presentation of the chosen node is finished, or if no component satisfies the condition. Link L_{END} warrants that the end of the composition presentation aborts the chosen node presentation.

It is worth to note that some attributes defined in SMIL cannot be tested in the proposed solution. For example, this is the case of tests related to the available network bandwidth. For this kind of attribute, the node being tested must be chosen not only when its attribute value is equal to the value defined by the system, but also when it is smaller than this value. Moreover, in this case, more than one link $L_{IN,i}$ could be satisfied. Simply changing the comparison operator in the meeting point conditions of the links solves the first point. The second point is overcome by the computation of the first node that satisfies its tests, following the declaration order. This computation is done in the operation list specified in the presentation event description.

5.7 Example of conversion

This section presents an example of conversion of a SMIL document into an NCM document. The SMIL document is presented in Figure 5. The document body is composed of a sequential element with two parallel elements, each one with two media object elements. The first parallel element is finished by the last component, and the second one is finished by a specific component (the video element).

```
<smil>
  <head>
    <meta name="title" content="matise"/>
    <layout>
      <root-layout id="matise" background-color="#fcedc4" width="721" height="587"/>
      <region id="m_title" left="4%" top="4%" width="47%" height="22%"/>
      <region id="V-Main" left="52%" top="5%" width="45%" height="42%"/>
      <region id="music"/>
    </layout>
  </head>
  <body>
      <seq id="News-at">
        <par>
          <text id="Leader-Title" region="m_title" src="leader_title.html"/>
          <audio id="Leader-Music" region="music" src="logo1.aiff"/>
        </par>
        <par endsync="id(DCAB-Video)">
          <text id="DCAB-Intro" region="m_title" src="intro.html" dur="8s"/>
          <video id="DCAB-Video" region="V-Main" src="zoomin.mpv"/>
        </par>
      </seq>
  </body>
</smil>
```

Figure 5 – Example of a SMIL document

The NCM document generated by the conversion is shown in Figure 6. The body element has been converted into a composite node (“example.smil”), containing another composite node that corresponds to the sequential element

("News-at"). Both nodes "example.smil" and "News-at" contain links to represent the sequential behavior, as described in Section 5.4. The parallel elements have been converted into composite nodes ("$par1" and "$par2") containing terminal nodes that correspond to the media object elements, and links to represent the parallel behavior, as described in Section 5.3.

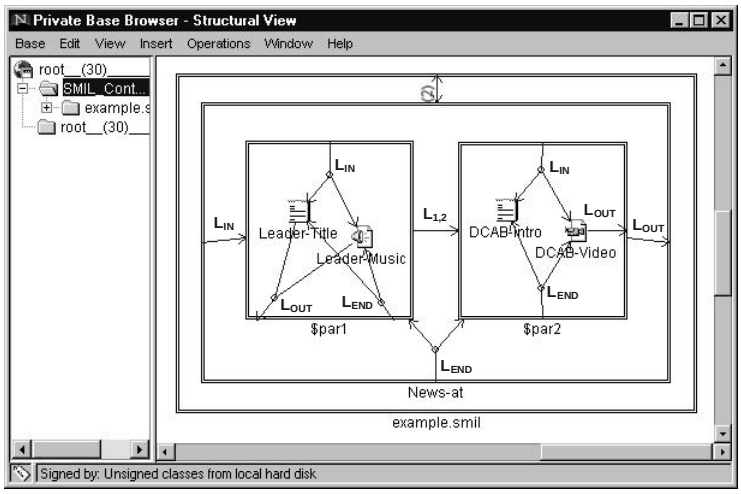

Figure 6 – The generated NCM document

5.8 Implementation

The screen presented in Figure 6 is the client browser of the HyperProp system. This browser provides a graphical structural view of NCM documents, allowing both authoring and navigation operations. The conversion of SMIL documents into NCM documents is called by a menu item. The translator automatically imports a set of SMIL documents and adds them to a special composite node, called "SMIL Context", which also contains NCM links corresponding to external links between the SMIL documents.

The process of conversion is divided in two parts. The first one is a parser that reads the SMIL document, verifies its syntactic correctness and builds a data structure that represents the SMIL elements in an object-oriented basis. The second one translates this data structure to the NCM data structure, which represents NCM documents. The reason for dividing the conversion in these two parts is that the SMIL document is read only once, and the translation is started only after verifying that the document is correctly specified.

6 Related Work

There are some proposals to allow the presentation of multimedia documents on the Web. One of them is HTML+TIME (*Timed Interactive Multimedia Extensions for HTML*) [12]. This proposal aims to incorporate synchronization facilities, creating temporal extensions to HTML, and not using a new substitute language. As a result, any HTML element could be presented at a specific moment with a given duration. Besides the new attributes, HTML+TIME also incorporates some SMIL elements, such as *par*, *seq* and *switch* compositions, which can be used to specify more complex relations. However, the proposed model inherits many HTML limitations, such as the fact that links are unidirectional and embedded in the node content (HTML pages). This limitation hinders, among other things, node content reuse without inheriting relations, link creation in pages where users do not have permission to write, and knowing which links make reference to a given page.

Another proposal following the SMIL approach of defining a declarative language to specify multimedia presentations on the Web is presented in [6]. This proposal is based on three concepts: *hypertime links* for temporal compositions, *time bases* for close lip-sync synchronization between media objects, and *dynamic layout* for positioning and movement. Hypertime links are similar to traditional hyperlinks, containing only one source anchor and only one target anchor. The difference from hyperlinks is that, similarly to NCM presentation events, source anchors on hypertime links do not need user interaction to be activated. The time base concept defines how an object should adjust its presentation rate. One time base can be shared by several objects, establishing a master-slave way to specify fine synchronization among them. Therefore, similarly to NCM, the model presents the elastic time concept, that is, an object duration can be calculated depending on the environment and the author specification. The layout contains spatial information to present the objects. In order to make no distinction between temporal and spatial synchronization, layouts are also considered objects. The authors also defined an execution architecture and implemented a prototype using Java and Dynamic HTML. This prototype allows presenting documents defined in the proposed extended language using commercial browsers. This proposal, like SMIL and NCM, separates link from data content. However, the proposed model does not present any composition concept, making difficult or even impossible a structured authoring process. Defining relationships among more than two objects is not allowed either. Behavior changes in the presentation of objects are previewed as future work.

Madeus [3], an authoring and presentation system for multimedia documents based on a constraint model, developed a work similar to the one presented in this paper, permitting importing SMIL documents into its environment [2]. Madeus conceptual model also offers compositions for logical structuring of documents, and temporal and spatial relationships among components. However, it also has some limitations, as it does not offer *m:n* relations and does not allow the specification of behavior changes during an object presentation. The work presented in [2] is

concentrated on the SMIL temporal model, and does not consider all SMIL elements. For example, SMIL *switch* compositions are not mentioned.

7 Final Remarks

This paper presented how to overcome the limitations of the SMIL language through its integration with the NCM event-driven model. The exercise of representing SMIL documents has led to refinements in some aspects of NCM, as the definition of the *repeat* attribute of events and how its value is set. It was shown that all SMIL functionality could be represented in NCM. However, the way parallel and sequential compositions were modeled caused a little difference in relation to the SMIL recommendation. According to this recommendation, when a link is fired by a user selection, the destination document is presented from the target element until the end of the document. In the conversion to NCM, when a hyperlink has as target end point the presentation of a node *N* contained in a parallel or switch composition, only this node will be presented. When *N* is contained in a sequential composition, only the nodes in the composition (node *N* and subsequent nodes) will be presented. NCM could be easily extended to embed SMIL behavior. However, we think that we should try a general solution, since this SMIL behavior is not always desirable.

NCM is the conceptual data model of the HyperProp system. HyperProp is implemented in Java and is integrated with the Web. This integration, presented in details in [7], has allowed providing support for structured authoring of documents, version control and definition of relationships for temporal and spatial synchronization. The integration also provided a graphical structural view of Web sites using fisheye techniques. As a result of the work presented in this paper, a SMIL-NCM translator was added to the system, to automatically import SMIL documents to NCM. It is now possible to incorporate the facilities offered by NCM and implemented in HyperProp to SMIL documents.

In spite of NCM complex syntax for the definition of links, we are working on the HyperProp system in order to offer a high-level graphical interface for document edition. The aim of the system is to provide an interface with templates of the main types of links, for authors that are not familiar with the model. For advanced authors, the system provides methods to directly edit the meeting point of links, allowing the specification of more complex relations.

As a future work, it is intended to define NCM extensions to allow a direct specification of the main facilities of the SMIL language. An example of extension is the definition of NCM *parallel* and *sequential composition* classes, which would have the same semantics of the homonym SMIL classes. This extension would allow generating SMIL documents from NCM documents, with the restriction that only a subset of the NCM documents would be appropriate to be converted, due to SMIL limitations. Another future work is the definition of an XML DTD to support a declarative specification of documents that complies with these extensions of

NCM. The aim is that this new language incorporates facilities that overcome SMIL limitations, while keeping its present simplicity for the definition of presentations.

References

1. Buchanan, M.C.; Zellweger, P.T. "Automatically Generating Consistent Schedules for Multimedia Documents", *Multimedia Systems*, Springer-Verlag, April 1993.
2. Collomb, P. "De Smil à Madeus: deux langages de synchronisation de documents multimédia", *Rapport de Maîtrise Informatique de l'Université Joseph Fourier (Grenoble 1)*, September 1998.
3. Jourdan, M.; Layaïda, N.; Roisin, C.; Sabry-Ismail, L.; Tardif, L. "Madeus, an Authoring Environment for Interactive Multimedia Documents", *Proceedings of the ACM Multimedia Conference 98*, pp. 267-272, ACM, Bristol, September 1998.
4. Kim, M.Y.; Song J. "Multimedia Documents with Elastic Time", *Proceedings of the ACM Multimedia Conference 95*, San Francisco, November 1995.
5. Pérez-Luque, M.J.; Little, T.D.C. "A Temporal Reference Framework for Multimedia Synchronization". *IEEE Journal on Selected Areas in Communications (Special Issue: Synchronization Issues in Multimedia Communication)*, Vol. 14, No. 1, January 1996. pp. 36-51.
6. Rousseau, F.; Duda, A. "Synchronized Multimedia for the WWW". *Proceedings of the Seventh International World-Wide Web Conference*, Brisbane, April 1998.
7. Rodrigues, R.F.; Muchaluat-Saade, D.C.; Soares, L.F.G. "Composite Nodes, Contextual Links and Graphical Structural Views on the WWW". *Special Issue on World Wide Web of the Journal of the Brazilian Computer Society (JBCS)*, Vol. 5, No. 2, November 1998. pp. 31-44.
8. Soares, L.F.G.; Casanova, M.A.; Rodriguez, N.L.R. "Nested Composite Nodes and Version Control in an Open Hypermedia System". *International Journal on Information Systems (Special Issue on Multimedia Information Systems)*, Vol. 20, No. 6. Elsevier Science Ltd, 1995. pp. 501-519.
9. Soares, L.F.G.; Muchaluat-Saade, D.C.; Rodrigues, R.F. "Authoring and Formatting of Hypermedia Documents", *Workshop on Multimedia Authoring Systems in the IEEE International Conference on Multimedia Computing and Systems*, Austin, June 1998.
10. Soares, L.F.G.; Souza, G.L.; Rodrigues, R.F.; Muchaluat-Saade D.C. "Versioning Support in the HyperProp System". *Multimedia Tools and Applications - An international Journal*, Kluwer Academic Publishers, Vol. 8, No. 3, May 1999.
11. "Synchronized Multimedia Integration Language (SMIL) 1.0 Specification". *W3C Recommendation*. Available in http://www.w3.org/TR/REC-smil, June 1998.
12. "Timed Interactive Multimedia Extensions for HTML (HTML+TIME)". *W3C Note*. Available in http://www.w3.org/TR/NOTE-HTMLplusTIME, September 1998.