Formal Approach for the Coherence Control of SMIL Documents

S. Mazouz, D. Dahmani and L. Kaddouri

LSI, Computer Science department, USTHB University
P. O. Box 32, El Alia Bab-Ezzouar, Algiers, Algeria
e-mails: mazouz@lsi-usthb.dz ddahmani2000@yahoo.com,
kaddouri_lies@hotmail.com

Abstract

This paper presents a formal approach based on Time Petri net (TPN) for the coherence control of SMIL documents. TPN model has been widely used to specify real time systems. The question addressed by this paper is whether SMIL documents can be easily modelled by TPN model. For this purpose, we show how to translate a specification of a given document SMIL into a TPN. We have extended the TPN model with component concept which is the building block in the translation procedure. The reachability analysis of TPN allows to verify the consistency proprieties of a document. Furthermore, our model can be used to define scheduling policies.

Key words: Formal methods, Time Petri net, Multimedia and SMIL

1. INTRODUCTION

The Interactive Multimedia Documents (IMDs) have been largely used in several fields and are accessed over the web. The specification of the temporal structure of IMDs has been reported in several publications by the proposal of models, languages and authoring tools, for instance, IMAP [15], MADEUS[8] and SMIL[12]. The language SMIL has been proposed by the W3C as a standard solution for the presentation of IMDs over the web. The temporal model of SMIL allows an author to describe temporal synchronisation relationships. These relations constitute a set of synchronisation constraints which must be satisfied at the presentation of the document. Unfortunately, an incoherence situation may occur as ("deadlock", "media never played", "reference to a media that does not exist", etc). For these reasons, an approach that detects and corrects these inconsistencies is needed. This can be done with *Simple Temporal Constraints problems* as used in *Madeus* [8] (it is an authoring environment for multimedia documents). In this approach, a multimedia document is translated into a graph where nodes correspond to events such as start or end of a media object and arcs to relations between events. These relations take account of metric information (for instance, duration of task). Algorithms such as cycle detection are applied to these

graphs to control coherence and other proprieties. Unfortunately, this framework does not suit some important characteristics of multimedia authoring process such as the presence of links in objects and objects whose duration is not controllable (typically videos).

In [13], another methodology for the design of SMIL documents based on the formal description technique RT-LOTOS is proposed. This approach presents the advantage of providing techniques for modelling the **dynamic** behaviour of the document and its temporal non-determinism. Furthermore, a RT-LOTOS specification [7] [5] is based on composition of processes; this concept allows to describe a multimedia document with simple media objects which can be composed. RT-LOTOS allows also to give a formal semantic to logical and temporal behaviour of a document. Temporal automata derived from RT-LOTOS specification are used for temporal consistency checking (at this level, simulation techniques of RTL tool are used). In [1], a similar approach based on ATN (Algebraic Temporal Net) has been proposed. ATN allows to specify the behaviour of real time system by using an algebraic expression which combines algebraic operators on time Petri nets instantiations.

Our objective is to explore time Petri net's [10] capabilities to model and analyse multimedia documents. This paper presents the on-going work of our approach.

This paper contains four sections. The first one is consecrated to describe the main aspects of SMIL language. In the second section, we present the time Petri net model. Our approach is described in section three. Section four gives some aspects of the reachability analysis.

2. SMIL LANGUAGE

First, we describe the main temporal aspects of SMIL considered in this paper. Then, we introduce, through examples, two kinds of incoherence in SMIL documents.

2.1 Presentation

SMIL is an acronym of "Synchronised Multimedia Integration Language" [12]. It is a declarative language based on XML (Extensible Markup Language), that allows authors to describe interactive multimedia presentations. Using SMIL, an author can specify the temporal behaviour of a multimedia presentation, associate hyperlinks with media objects and describe the spatial aspect of the presentation on a screen.

A media object can be:

- Basic Object like a video, an audio, an image, a text or a reference (i.e. hyperlink such as <a> element and <area> element). Each basic object E is characterised by its beginning time (noted begin(E)), its ending time (noted end(E)) and its duration (noted dur(E)). begin(E) and end(E) can be known values, synchronisation events (for example, the relation begin(E)= end(C) means that E begins when C terminates) or external events (for example, the relation begin(E) = foo.activateEvent means that E begins when the element foo is activated).
- Composite Object (i.e. defined with synchronisation operators): the main synchronisation operators, considered in this paper, are the *par* operator (which plays child elements as a group) and the *seq* operator (which plays the child elements one after another in a *sequence*). *par* and *seq* operators have the same attributes as the basic objects.

The body part of a given SMIL document contains all the temporal constraints to check. Note that it has the same behaviour as the *seq* operator. In the sequel, the synchronisation operators are also called time containers. Begin and end attributes are relative time. Indeed, each object has a reference time which depends on its parent time container.

In the rest of this section we include a set of examples that illustrate both the usage of the SMIL syntax, as well as the semantic of specific constructs. However, the additional syntax related to layout and other issues specific to individual document types is omitted for simplicity.

In the first example, we illustrate a simple timing within a sequence time container. Each child of the operator *seq* begins by default when the previous element ends. In this case the reference time of an element is the end of the previous element except for the first one which is relative to the start of seq .

```
<seq>
  <img id="i1"  dur="5s" src="img1.jpg" />
  <img id="i2" dur="10s" src="img2.jpg" />
  <img id="i3" begin="1s" dur="5s" src="img3.jpg" />
</seq>
```

The element "i1" begins immediately (the default begin offset is always 0 second), with the start of the *seq* , and ends 5 second later. The second element "i2" begins, by default, 0 second after the previous element "i1" ends, i.e. is 5 seconds into the *seq* . Element "i2" ends 10 seconds later, .i.e. at 15 seconds into the *seq* . The last element, "i3", has a begin offset of 1 second specified, so it begins 1 second after the previous element "i2" ends, and has a duration of 5 seconds, so it ends at 21 seconds into the *seq*.In the second example, a simple timing within a parallel time container is illustrated. Start time of par is the reference time for all its children. By default, each child of par begins when the operator *par* begins.

```
<par>
  <img id="i1" dur="5s"  src="img.jpg" />
  <img id="i2" beging =3s dur="10s" src="img2.jpg" />
  <img id="i3" begin="2s" end="5s" src="img3.jpg" />
</par>
```

Element "i1" begins immediately when the *par* begins, which is the default begin time. However “i2” and “i3” begin at 3 seconds and 2 seconds respectively since both have an explicit begin offset. The duration of "i1" is 5 seconds which means “i1” ends at 5 seconds into the *par* . "i2" ends at 13 seconds into the *par* since it has an explicit duration of 10s. The last element "i3" ends at 5 seconds into the par because it has an explicit end offset.

By default, a *par* group ends when all elements in the group finish playing. You can modify this behaviour with the *endsync* attribute. The following table lists the endsync values.

Value	Function
all	Ends the group once all clips have finished.
first	Ends the group when the first clip finishes.
ID	Ends the group when a specific clip finishes.
last	Ends the group when the last clip finishes. This is the default.

The two values endsync="last" and endsyc="all" are similar. Both end a *par* group when the last element finishes playing.

In the following example, the parallel group concludes when the video ends, as long as the video plays more than two minutes. If the video has a shorter duration, the group ends when the image clip's two-minute duration expires.

```
<par endsync="last">
   <video id="vid1" src="video1.rm" />
   <img src="banner.png"  dur="2min"/>
</par>
```

The values endsync="first" and endsync="*ID*" can stop a <par> group when a specific element stops playback. For example, the attribute endsync="*ID*" causes the group to conclude when the designated element ends playback. All other elements in the group stop playing at that point, as illustrated in the following example:

```
<par endsync="vid1">
  <video id="vid1" src="video1.rm" />
  <textstream src="moreinfo.rt" />
  <img src="banner.png"  dur="2min"/>
</par>
```

A SMIL document can define links to other media. Two hyperlink tags exist : <*a*> and <*area*> (both found in HTML). The <*a*> tag is the simpler means of creating links, but the <area> tag is more powerful. The <area> tag includes all of the features of <a>, and adds additional ones, such as the ability to define multiple links for each clip, and to create hot spots (image maps) and timed links.

The simplest type of link connects an entire source clip to another clip, as illustrated in the following example.

```
<a href="URL /  video2">
  <video src="video1"  />
</a>
```

In this example the source clip “video1” links to the target clip “video2”. When the viewer clicks “video1” as it plays, “video2” replaces it. The *href* attribute allows to give the URL address of the target clip.

For creating a timed link, we use an <*area*> tag and include temporal attributes (begin and end) that specify when the link is active, relative to the start of clip playback.

The following example creates two temporal links for the clip “video1”. The first link is active for the first 30 seconds of playback. The second link is active for the next 30 seconds because no spatial coordinates are given, the entire video is a link.

```
<video src="video1" >
  <area href="http://www.real.com" begin="0s" end="30s".../>
  <area href="http://www.realnetworks.com" begin="30s" end="60s".../>
</video>
```

2.2 Incoherence in SMIL documents

The presentation of an interactive multimedia document depends on the temporal constraints defined on the objects of the document. Indeed, these relations must be well defined otherwise they can give deadlock situations. Authors of SMIL documents are free to express relationships between SMIL components without describing how they are processed. The author can incrementally modify his specifications by adding, or removing constraints from the current temporal constraints or replacing an object by an another which may have a different duration. These modifications may imply incoherence in the temporal specified constraints. There are mainly two kinds of incoherence:

2.2.1 Quantitative incoherence

Let us suppose that a scenario defines a parallel presentation of two objects namely A and B. The element B begins 4 seconds after the *par* begins. The duration of B is 5 seconds, then it ends at (4s + 5s) 9 seconds. The element A begins 1 seconds after the element B begins (i.e. at 5 seconds) and ends at 3 seconds. This is impossible.

Example:

```
<par>
  <img id="A" begin =begin(B)+1s,  end= 3s   />
  <img id="B" begin=4s, dur="5s"  />
</par>
```

2.2.2 Qualitative incoherence

Let us suppose that a scenario defines a parallel presentation of two objects namely A and B. The beginning of element A depends on the beginning of element B and vice versa. Neither A nor B can begin.

Example:

```
<par>
  <img id="A" begin = begin(B)   />
  <img id="B" begin = begin(A)   />
</par>
```

3. TIME PETRI NETS

Several timed extensions of Petri Nets have been proposed, such as time Petri nets [10] and timed Petri nets. Among these, Time Petri nets (TPNs) are most widely used for

real-time system specification and verification. It has been shown in [3] that TPNs are sufficient to express most useful temporal constraints and are also more expressive than Ramchandis's Timed Petri nets.

First, we recall the definition of Time Petri Nets.

Definition 1 (Time Petri Net)

A Time Petri Net (TPN) is a tuple $N = (P, T, W, M_0, Is)$ where:

1. P and T are nonempty finite sets of *places* and *transitions* respectively with $P \cap T = \emptyset$;
2. W is the *arc weight function*;
3. M_0 is the *initial marking* ;
4. Is: $T \rightarrow Q+ \times (Q+ \cup \{\infty\})$ is the *static interval function*, where Q+ is the set of positive rational numbers.

The Is function associates each transition **t** with a temporal static interval. The left bound and the right bound are called the static earliest firing time (sEFT) and the static latest firing time (sLFT) respectively.

A marking is a function $M: P \rightarrow N$ where M(p) denotes the number of tokens at place p.

The firing of a transition depends on both enabling and timing conditions.

Definition 2 (enabling)

A transition t is enabled in a marking M iff for all p in P, $M(p) \geq W(p,t)$.

We denote by enabled(M) the set of all transitions enabled in marking M. A transition t enabled in marking M is fireable in the associated untimed Petri net but not necessarily in the time Petri net.

Definition 3 (State)

A state of TPNs, called interval state, is be a pair E = (M, Id) in which M is a marking and Id is a dynamic firing interval function. Functions Id associates with each enabled transition the time interval [dEFT, dLFT] in which the transition is allowed to fire.

Some transitions may be enabled by marking M, but not all of them may be allowed to fire due to the firing constraints of transitions.

Definition 4 (Firing Condition)

Firing a transition t, at relative time θ, from state E = (M, Id), is allowed iff both the following conditions hold:

1. The transition is enabled : $\forall p \in P, M(p) \geq W(p, t)$
2. The relative firing time θ, relative to the absolute time at which state E has been reached, is comprised between the dEFT of transition t and the smallest of the dLFTs among of the transitions enabled.

The first condition is the usual one of enabledness for Petri nets and the second results from the necessity of firing transition in their firing interval. It expresses the fact that an enabled transition may not fire before its EFT and must fire before or at its LFT unless another transition fires before and modifies marking M.

Definition 5 (Firing rule)

If a transition t_f is fireable at the time θ from state E = (M, Id), its firing leads to a new state E' = (M', Id'), computed as follows:

1. The new marking M' is defined for any place, as in Petri nets, as:

$M'(p) = M(p) – W(p, t_f) + W(t_f, p)$

2. The new firing intervals Id' for transitions enabled by marking M' are computed as follows:

- For all transitions t enabled by marking M and not in conflict with t_f, then: Id'(t)=Max(0, dEFT(t)- θ)), dLFT (t)- θ
- All other transitions have their interval set to their static firing interval.

Only enabled transitions not in conflict with t_f, have their intervals shifted by the value of θ toward the time origin; the remaining enabled transitions have their interval set to their static firing interval.

Note that there is another characterisation of state, called clock state. In this characterisation, a valuation clock is associated to each enabled transition. When a transition t becomes enabled its clock is set to zero. The value of this clock increases with time until the transition is fired or disabled by the firing of another transition. When the clock reaches sLFT(t), transition t must be fired immediately, without any delay.

The firing rules above define a reachability relation among states of a time Petri net. However, using this technique of states for analysis purpose is not possible in general since this set may be infinite. An enumerative technique is proposed in [3], based on the concept of state class.

Informally, states classes are defined as the union of all firing values which are possible from a given marking.

Definition 6 (state class)

A state class is a pair C=(M, D) where:

1. M is a marking M
2. D is a firing domain, i.e. a set of constraints on the values of firing time for enabled transitions by current marking M. D represents the infinite number of time possible from marking M; it is defined as the union of firing domain of all the states in the class.

Berthomieu, Diaz and Menashe have defined a domain D as the solution of a set of a system of inequalities in which variables are one to one associated with the transitions enabled by marking M: $D=\{\theta / A.\theta \geq b\}$

Where A is a matrix, b a vector of constants, and θ a vector of variables, depending on the enabled transitions.

The initial state class C_0 is defined by the initial marking M_0 and the system of inequalities $\{sEFT(t) \leq t \leq sLFT(t) / t \in enabled(M)\}$.

Definition 7 (Firing Condition in State Class)

Let C=(M, D) be state class and t_f a transition.

The transition t_f is fireable from state class C iff both the following conditions holds:

1. t_f is enabled in marking M
2. The augmented system of inequalities is consistent: $D \cup \{ t_f \leq t / t \in enabled(M)\}$.

Definition 8 (Firing rule in State Class)
Let t_f be a fireable transition from state class C=(M, D). Firing t_f yields a new state class C'=(M', D') computed as follows:

1. $\forall p \in P$, $M'(p) = M(p) - W(p, t) + W(t, p)$
2. The domain D' is calculated by four steps:
   a. Add to the system D, the fireability conditions for transition t_f
   b. Make a change of variables: each variable t ($t \neq tf$) is substituted by $t+t_f$.
   c. Eliminate by substitution the variable t_f and all those associated to transitions in conflict with t_f for marking M.
   d. Add for every newly enabled transition t, the inequality: $sEFL(t) \leq t \leq sLFT(t)$.

The number of state classes is bounded if and only if the TPN is bounded in the sense of ordinary Petri nets theory.

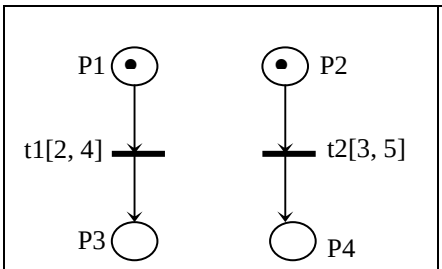

Fig. 1.a Simple TPN model

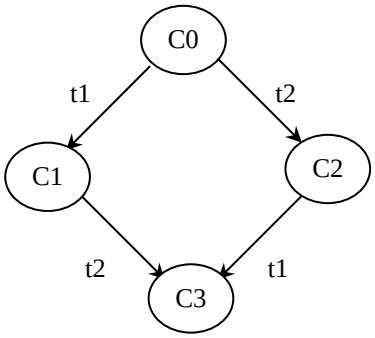

Fig 1.b State class graph

Example
Consider a simple TPN model, shown in Fig. 1.a.
The initial state class C0=(M0, D0) where

M0=(1 1 0 0)
D0={ $2 \leq t1 \leq 4$, $3 \leq t2 \leq 5$}

Firing t1 will result in state class C1=(M1, D1). Then it follows from definition 8 that:

M1=(0 1 1 0)
D1 is computed in four steps:

- Augmented system: $2 \leq t1 \leq 4$, $3 \leq t2 \leq 5$ plus $t1 \leq t2$.
- Change of variables: $2 \leq t1 \leq 4$, $3 \leq t2+t1 \leq 5$, $t1 \leq t2+t1$.
- Elimination of t1: $0 \leq t2 \leq 3$
- There is no new enabled transition at M1: $0 \leq t2 \leq 3$

Similarly at C0, firing t2 will results in state class C2=(M2, D2) where:

M2=(1 0 0 1)
D2= $0 \leq t1 \leq 1$

Both C1 and C2 lead to state class C3 = ((00 11), ∅) by firing t2 and t1 respectively. Note that C3 is a terminal state. Thus, the state class graph is depicted in Figure 1.b.

4. OVERVIEW OF THE TRANSLATION APPROACH

Our objective is to give a formal approach to verify the coherence of SMIL documents by using Time Petri Nets. Here we define the main translation patterns. A SMIL document is modelled by TPN (see figure 2) that contains two places (initial and final place) and a start transition ts-smil constrained with a time interval equals to [0, 0]. The initial place p_{in} is associated to the <smil> tag and corresponds to the start of the presentation. The final place p_f is associated to the </smil> tag and corresponds to the end of the presentation. In the initial marking M_0, the place p_{in} is marked with one token.

A SMIL document describes the composition of several media objects by means of synchronisation operators. To facilitate the derivation of a TPN from a SMIL

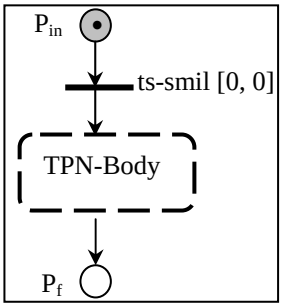

Fig 2 SMIL document

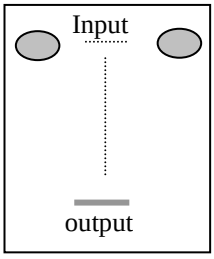

Fig 3 Component

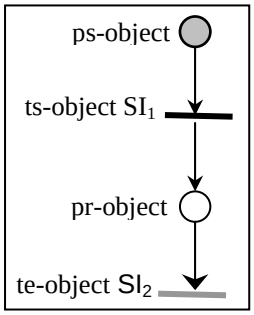

Fig 4 Basic Media object

document, we extend the TPN model with a 'component' concept (figure 3). A component is the basic building block in the translation procedure. It encapsulates a TPN which describes the temporal behaviour of a media object. Its input and output interfaces are respectively a set of input places and an output transition.

4.1 Basic object

A basic object E is translated into a TPN that contains a start place $p_{s\text{-}object}$, two transitions (a start transition $t_{s\text{-}object}$ and an end transition $t_{e\text{-}object}$) and a region place $p_{r\text{-}object}$. The presence of a token in the region place signifies that the object is "on execution" and can be labelled by the region's name if it's a visual object. The transitions $t_{s\text{-}object}$ and $t_{e\text{-}object}$ correspond respectively to the beginning and to the end of the object. The static temporal intervals associated to these two transitions are mainly depending on the attributes begin, end and duration.

The pattern of a basic media object is given in figure 4 where the grey place is the input interface and the grey transition is the output interface.

The static temporal intervals SI_1 and SI_2 are defined as follows:

1. The static interval SI_1 of $t_{s\text{-}object}$ depends on begin attribute of the object:

- The begin attribute is not specified (not given in the SMIL document), this means that the object must start when its reference time is activated. So, $SI_1 = [0, 0]$.
- Begin(E) is a known value: the object must start begin(E) time units after the activation of its reference time. Hence, SI_1 is equal to [begin(E), begin(E)].

- Begin(E) is a synchronisation event such as begin(E) = begin(A) + t, (where A is any media object having the same reference time as E and t is a natural number): object E must start t time units after the beginning of object A. This case is illustrated by figure 5.a. $t_{s\text{-}E}$ must occur t time units after $t_{s\text{-}A}$. $t_{s\text{-}base}$ is a transition which activates the reference time of A and E. For this reason, a synchronisation place is added between $t_{s\text{-}A}$ and $t_{s\text{-}E}$. The synchronisation of a media E with another media gives us a new component Es which encapsulates the component E.
- Begin(E) is a synchronisation event such as begin(E)=end(A)+t, (where A is any media object having the same reference time as E and t is a natural number). Object E must start t time units after the end of object A. This case is illustrated by figure 5.b. $t_{s\text{-}E}$ must occur t time units after $t_{e\text{-}A}$.

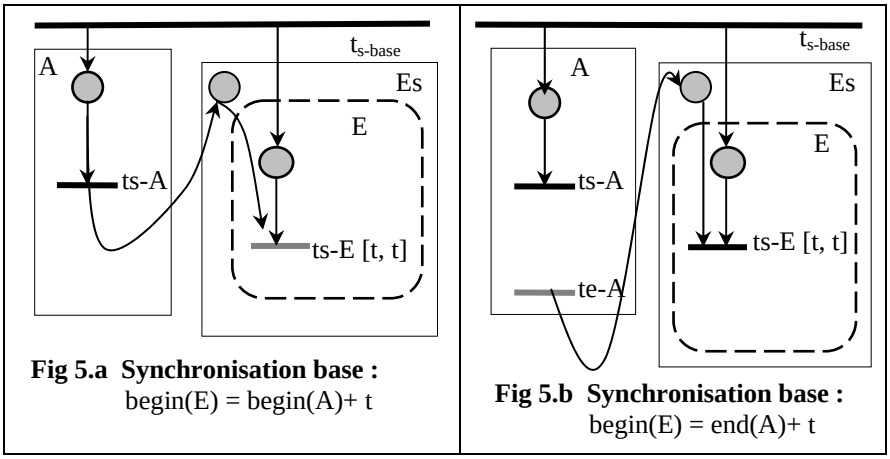

Fig 5.a Synchronisation base : begin(E) = begin(A)+ t

Fig 5.b Synchronisation base : begin(E) = end(A)+ t

2. The static interval SI_2 of $t_{e\text{-}object}$ depends on begin, end and duration attributes of the object:
- dur(E) and end(E) are not specified: the object can finish at any time. So, SI_2 =[0, ∞[
- end(E) is not specified and dur(E) is a known value: object E must finish dur(E) time units after its beginning. So, SI_2 = [dur(E), dur(E)],
- begin(E) and end(E) are known values and dur(E) is not specified: the end of object E must occur end(E)-begin(E) time units after E's start. Hence, SI_2 = [d, d] where d = end(E)-begin(E),
- begin(E), dur(E) and end(E) are known values: object E has been specified with two durations dur(E) and end(E)-begin(E). The smallest duration is taken into account (SMIL semantic). Hence, SI_2 = [d, d] where d = min(end(E)-begin(E), dur(E)),
- end(E) is a known value, dur(E) is not specified and begin(E) is a synchronisation event such as (begin(E) = begin(A)+ t) : this case is depicted in figure 6.a. The time interval SI_2 of E's end transition is equal to [0, ∞[because dur(E) is not specified. We add a new shared place named "disrupt" and a timer transition constrained by the time interval [end(E), end(E)]. The disrupt place is an "input/output" place for all the E's transitions. When the time allowed is passed, te-Es transition stops or inhibits the media E by 'stealing' the token from associated 'disrupt' place.

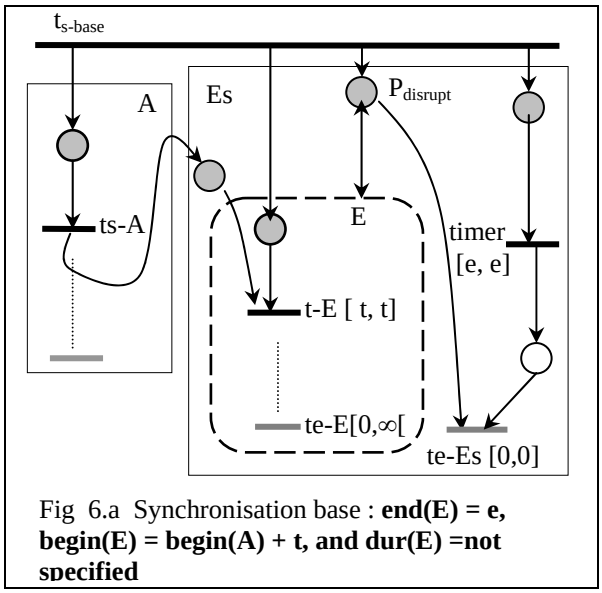

Fig 6.a Synchronisation base : **end(E) = e, begin(E) = begin(A) + t, and dur(E) =not specified**

- Figure 6.b presents the same synchronisation case as previously except that dur(E) is a known value. Hence, the time interval SI_2 of E's end transition is equal to [dur(E), dur(E)]. The firing of the first transition among the pair ($t_{e\text{-}E}$, timer) implies the firing of $t_{e\text{-}Es}$.

Other synchronisation events such as end(E) = end(A) + t are treated in similar manner.

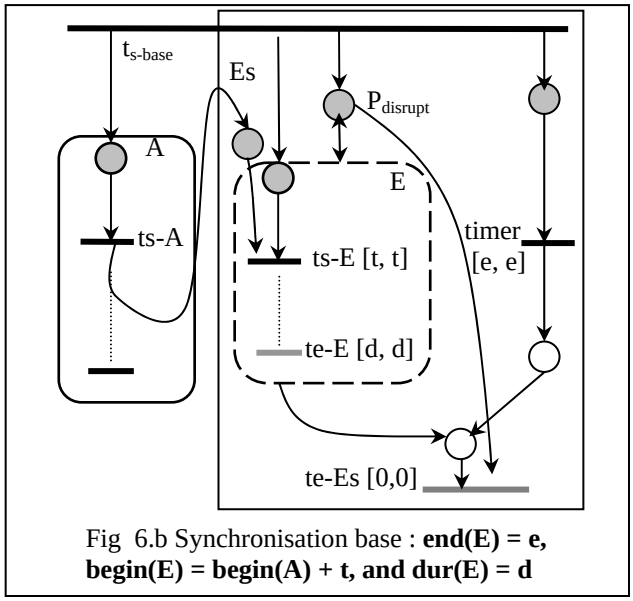

Fig 6.b Synchronisation base : **end(E) = e, begin(E) = begin(A) + t, and dur(E) = d**

4.2. Composite object

Besides input and output interfaces and the start transition, a component of a composite object contains its children's TPNs and a meet place. The pattern given in figure 7 enables to model the seq operator. $t_{s\text{-}seq}$ is linked to the input interface of the first child. The output interface of each child (except the last one) is linked to the input interface of its successor. The output interface of the last child is linked to the meet place. $t_{e\text{-}seq}$ is constrained by default to the interval [0, 0] because the seq must finish when its last child terminates.

The modelling of a par operator depends mainly on the "endsync" attribute. The pattern of figure 8 enables to model the fact that the *Par* waits the ends of all its children (i.e. *endsync ='last'*). $t_{s\text{-}par}$ is linked to all the input interface of its children.

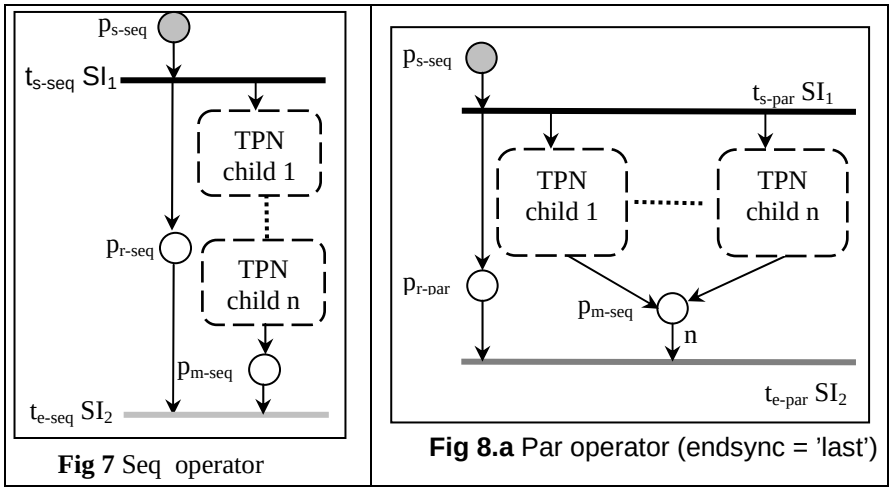

Fig 7 Seq operator

Fig 8.a Par operator (endsync = 'last')

The output interface of each child is linked to the meet place. $t_{e\text{-}par}$ is constrained by default to the time interval [0, 0] because the par must finish when all its children terminate.

The pattern of figure 8.b treats the case endsync='first'. When the first child terminates, all others children must also be stopped. For this reason, we add a marked disrupt place for each child. As soon as the first child terminates, transition $t_{E\text{-}par}$ must occur stealing the tokens in all the disrupt places. Thus, all other children are stopped.

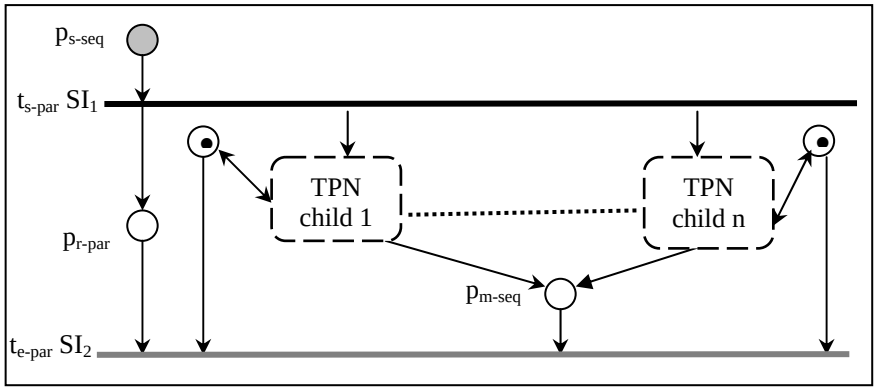

Fig 8.b Par operator (endsync = 'first')

The pattern of figure 8.c treats the case endsync='name-object' where name-object is the identifier of one child of par. When this child terminates, all other children (which have not terminated) must be stopped. We add a marked disrupt place for each child except name-object child. As soon as, the specified child terminates, transition $t_{e\text{-}par}$ must occur stealing the tokens in all the disrupt places.

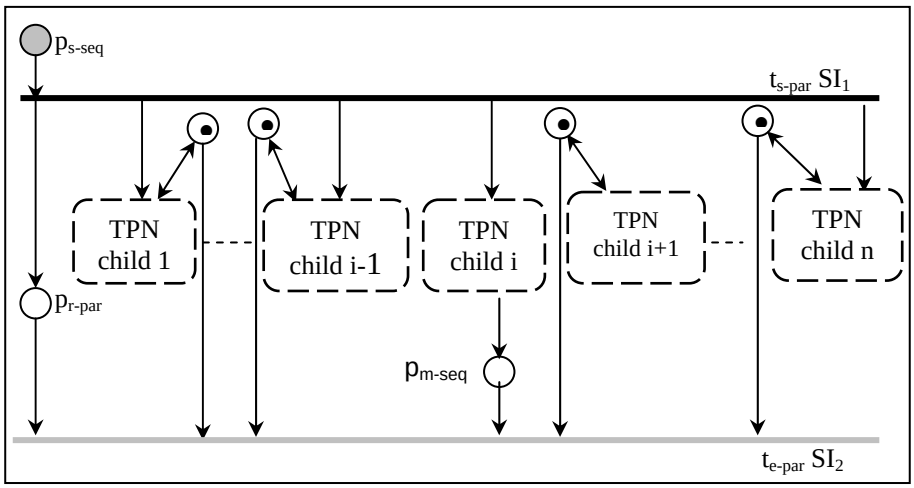

Fig 8.c Par operator (endsync = 'name-object') where name-object is the identifier of child i.

Example 1

Let us consider the scenario defined by the following SMIL document which consists of a sequence of a video clip (vid) followed by an image (img2). This sequence must be presented simultaneously with another image (Img1) followed by an audio (aud). Assume that the duration of img1, aud and vid are respectively 5, 20 and 10 seconds. Object img2 does not have an explicit duration. The end of img2's presentation is determined by the end of aud.

The TPN associated to this scenario, is given in figure 9. The disrupt place is used to stop the media seq2 as soon as the media seq1 has finished.

```
<par id='par1'>
   <seq id= 'seq1' >
         <img id='img1' dur='5s'…./>
         <audio id='aud' dur='20s'.../>
   </seq>
   <seq  id='seq2' end =end(seq1) >
         <video  id='vid'  dur='10s'..../>
          <img id='img2'.....:>
   </seq>
</par>
```

Example 2

```
<par id='par1'>
   <img id='img'  begin='2s' dur= "4s"  />
   <video  d='vid'  begin=begin(img)+"2s"  end="3s"
/>
    <text id='txt'   begin=begin(vid)  dur="2s" />
```

The TPN of the above SMIL document is depicted in figure 10.

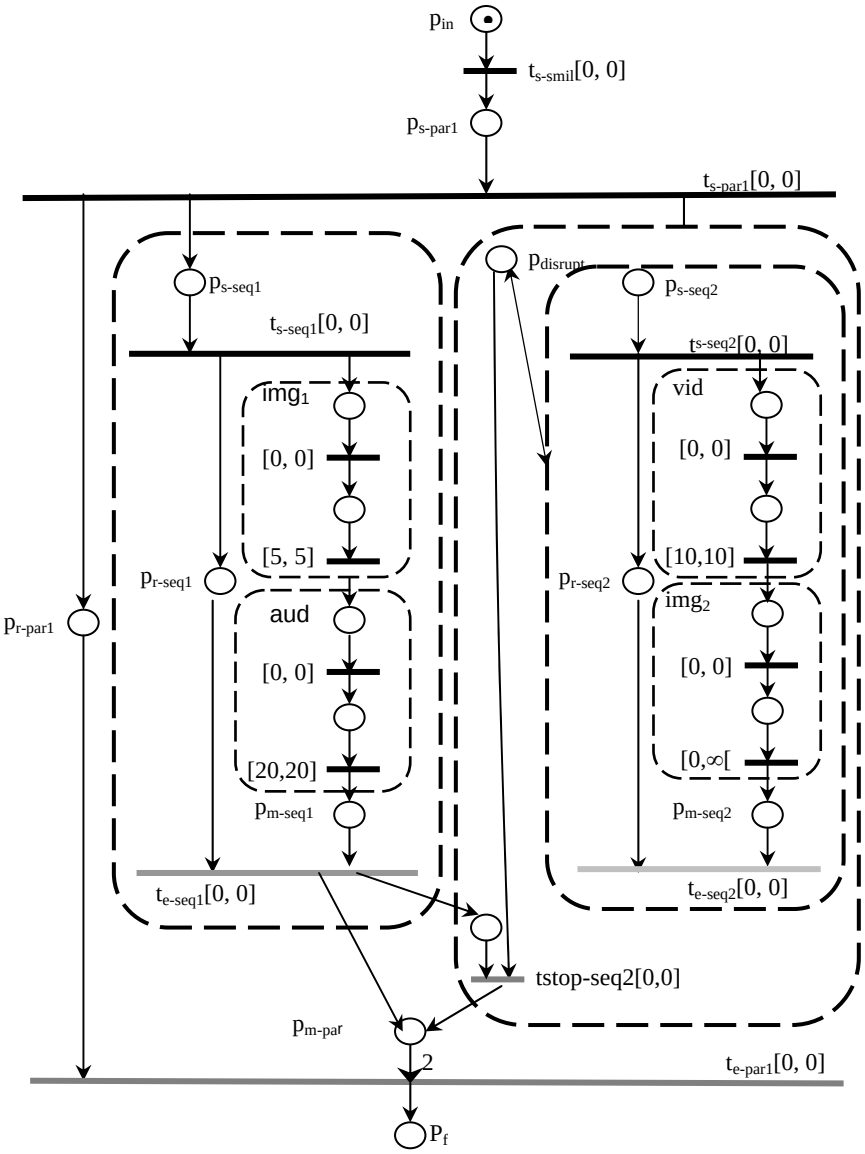

Fig 9 Time Petri net N_1

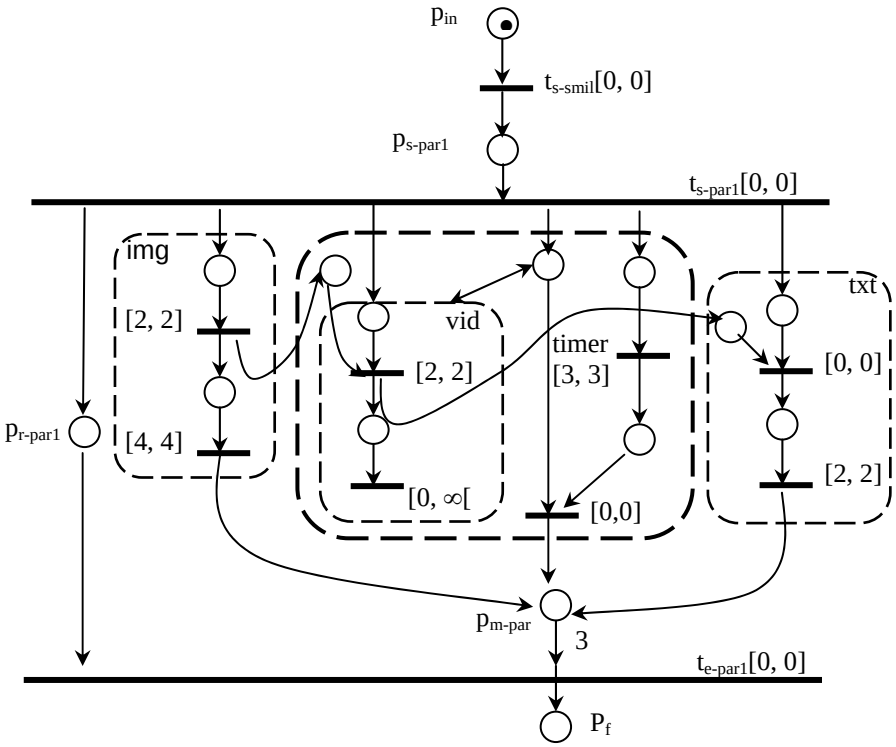

Fig 10 Time Petri net N_2

4.3. Link object

The area operator is translated into a TPN that contains a link place p_{link} and two transitions (a link start transition $t_{s\text{-}link}$ and a duration link transition $t_{dur\text{-}link}$). If the link can be activated in the time interval [d, d'] then the time interval of start link transition SI_1 is equal to [d, ∝] and the time interval of the duration link transition SI_2 is [d', d']. These two transitions have the link place as shared input place.

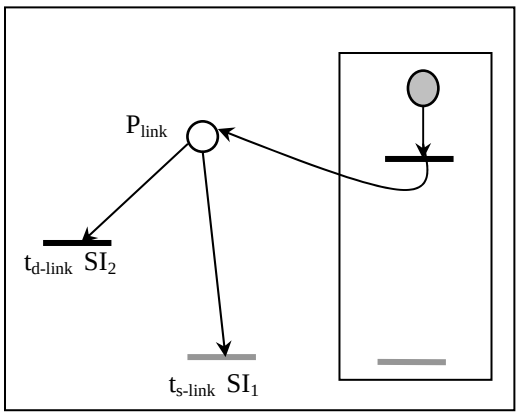

Fig. 11. area operator (show ='new')

Example

Let us consider the following SMIL document:

```
<par id=”par1”>
      <img id=”img”, begin=”2s”,  end=”10s” />
    <video  id=”vid”,  dur=”6s”>
        <area  href=”U1” begin=”0s” end=”3s”  show=”new” / >
       <area href=”U2”, begin=”3s” end=”6s” show=”new” / >
</audio>
</par>
```

Its TPN is depicted in figure 13.

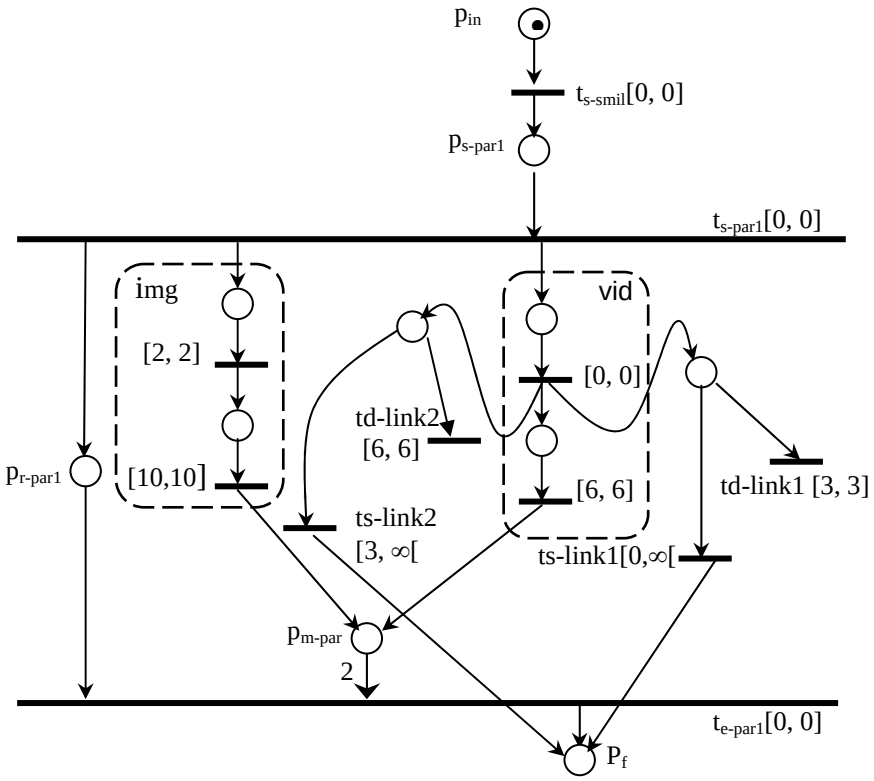

Fig. 12. Time Petri net N_3

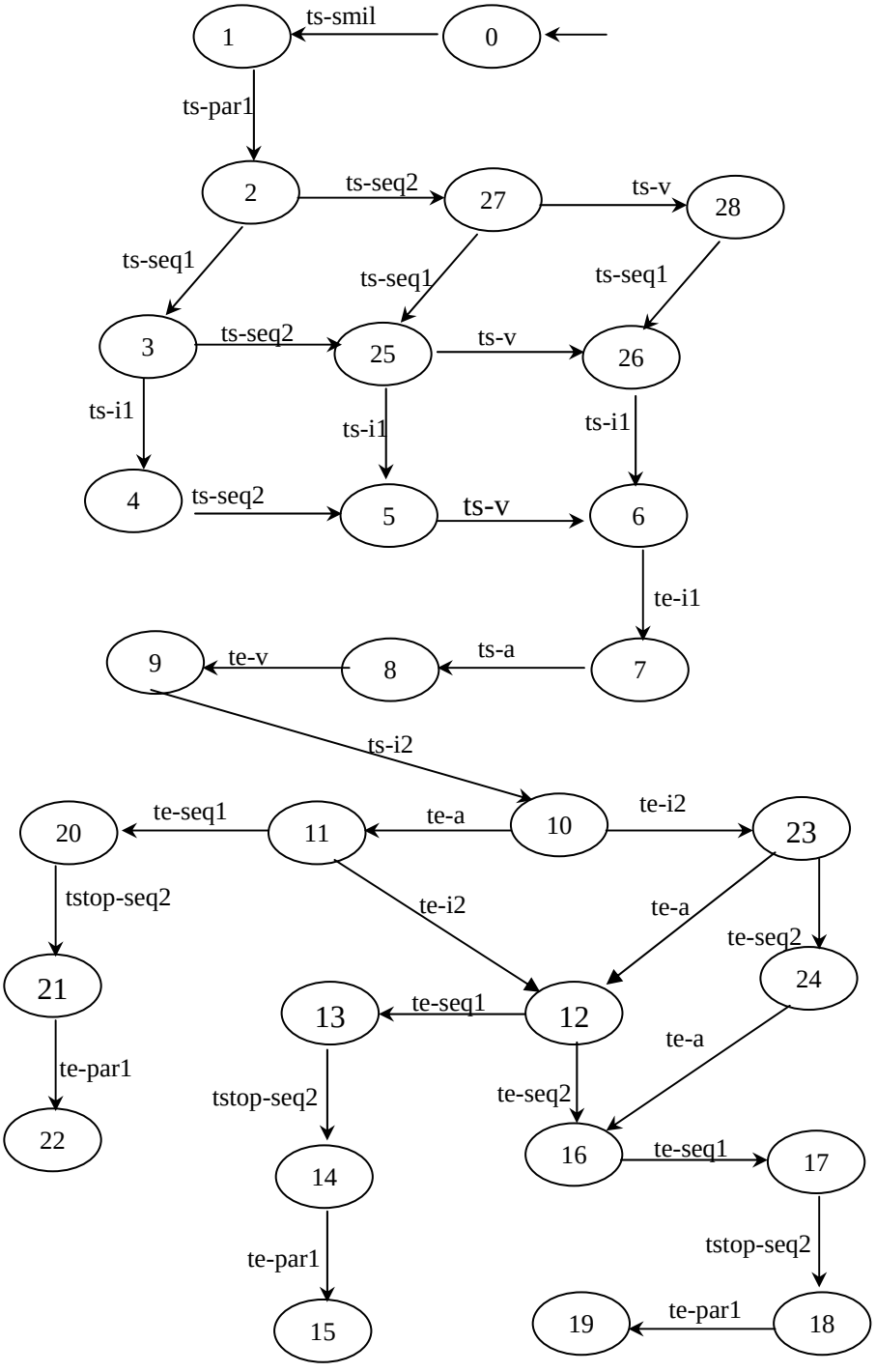

Fig. 13 State Class graph of N_1

5. ANALYSIS

As in [6], we consider that a document is consistent if the action characterising the start of the document is necessarily followed (some time later) by an action characterising the end of the presentation. Our analysis approach uses the state class space of TPN [3] in order to verify the consistency of a SMIL document. This latter is consistent if in every terminal state class (these classes are derived from the initial class), the final place p_f is marked. We have used the tool TINA [4] to compute state class space.

The state class space of TPN N_1 is depicted in figure 13. In the above example, in all terminal states classes reachable from the initial state class C_0, the place p_f is marked. We can conclude that the associated document is consistent.

But the state space class of N_2 (see figure 14) contains one terminal state C_5 in which the place p_f is not marked. So the associated SMIL document is not consistent. This is due to the fact that media img2 never begins.

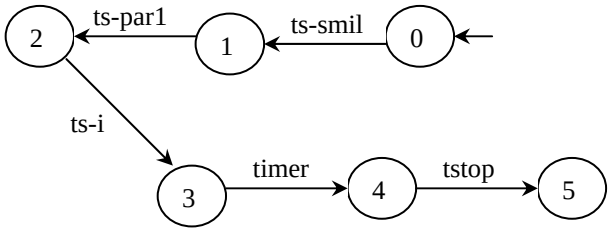

Fig 14 State class graph of N_2

6. CONCLUSION

In this work, we have defined a formal approach based on TPNs for coherence control of SMIL documents. Our model analyses only but efficiently a subset of SMIL documents. The state class graph derived from TPN allows to verify if a given presentation can be played until the end. Other information (such as object never played or object started and never terminated) can be extracted from the state class graph by using model checking technique. Such information is useful to correct errors in SMIL documents. A TPN associated to a given SMIL document can be used to compute minimal and maximal duration of every path of the TPN. These bounds can be used efficiently to apply an optimal scheduling policy to handle multimedia flow. It obvious that the number of classes may increase exponentially with the system size. State space analysis is a useful approach for the automatic verification. Unfortunately, it suffers from the so called explosion problem: the number of classes may increase exponentially with the system size. So, in our current works, we are interested to confine this problem by using reduction methods [9] as well as modular analysis [14].

Even though, TPN model has been successfully used in modelling a consequent subset of SMIL documents. Some cases can not be modelled easily. For instance, synchronisation between multimedia objects that haven't the same time reference. This is due to the fact TPN model can't be easily used to model some particular cases of timeout.

7. REFERENCES

[1] Abdelli**,** A.**,** Towards SMIL document Analysis using an algebraic Time Net, PCM 2004, Tokyo Japan.

[2] Allen, J.F. "Maintaining Knowledge About Temporal Intervals". In: Communications of ACM, v26, n11, 1983, pp832-843.

[3] Berthomieu, B., Diaz, M., Modeling and verification of time-dependent systems using time Petri nets. In *IEEE Transactions on Software Engineering*, volume 17. *n°*3, 1991.

[4] Berthomieu, B., Ribet, P.O., Vernadat, F., "The TINA Tool: Construction of Abstract State Space for Petri Nets and Time Petri Nets", International Journal of Production Research, Vol.42, N°14, pp.2741-2756, 2004

[5] Bolognesi, T. and Brinksma, E., Introduction to the ISO specification language LOTOS. In *Computer Networks and ISDN Systems*, volume 14. *n°*1, 1987.

[6] Courtiat, J.P, Oliveira, R.C, "Proving Temporal Consistency in a New Multimedia Synchronisation model", In Proceedings of ACM Multimedia'96, Boston, USA, Nov. 1996. pp 141-152.

[7] ISO, "LOTOS – A Formal Description Technique Based on the Temporal Ordering of Observational Behavior", ISO Information Processing Systems – Open Systems Interconnection IS 8807, September 1988

[8] Jourdan M., Layaida, N., Roisin, C., Sabry-Ismail, L. and Tardif L, Madeus: an Authoring Environment for Interactive Multimedia Documents". In 6th ACM Multimedia'98 Conference, P. 267-272, Bristol (UK), 1998

[9] Kristensen, L.M., Valmari, A., *Improved Question-Guided Stubborn Set Methods for State Properties*. Application and Theory of Petri Nets 2000, 21st International Conference, Aarhus, Denmark, June 26-30, 2000, Proceedings, Lecture Notes in Computer Science 1825, Springer-Verlag 2000, pp. 282-302. Also Department of Computer Science, University of Aarhus, DAIMI PB-543, January 2000, 21 p.

[10] Merlin, P. M., Farber, D. J., Recoverability of Communication Protocols. IEEE Transactions of Communications, 24(9):36–103, September 1976.

[11] RT-Lotos, Real-Time LOTOS Laboratory. http://www.laas.fr/RT-LOTOS.

[12] SMIL, Recommandation de SMIL2.0 du W3C http://www.w3c.org/TR/smil20

[13] Sampaio, P.N.M., Conception formelle de documents multimedia : une approche s'appuyant sur RT-LOTOS, Univ Paul Sabatier thesis, Toulouse, France, april 2003.

[14] S. Christensen and L. Petrucci, Modular analysis of {Petri} nets, Computer Journal, vol 43, N° 3, pages 224--242", 2000.

[15] M. Vazirgiannis, S. Boll, Events in Interactive Multimedia Applications: Modeling and Implementation design . In Proc of IEEE International Conference on Multimedia Computing and Systems (ICMCS'97), Ottawa-Canada, June 1997.