Modeling and Validation Support for Interactive Networked Multimedia Applications

Luciano Paschoal Gaspary
Maria Janilce B. Almeida

Universidade Federal do Rio Grande do Sul
Instituto de Informática
Curso de Pós-Graduação em Ciência da Computação
Campus do Vale, Bloco IV – Bento Gonçalves, 9500 – Agronomia – 91591-970
Porto Alegre, RS – Brazil
E-mail: {paschoal, janilce}@inf.ufrgs.br

Abstract. This work presents MUSE, a graphical environment for modeling interactive networked multimedia applications. Through an advanced graphic interface and a new high-level authoring model, it is possible to create complex systems in a fast and intuitive way. The authoring model proposed in this work and adopted by the environment deals with media objects distributed in a computer network, allowing the definition of acceptable delay thresholds and alternative media objects. Due to the large expressiveness of the model, however, specifications can be generated with logical and temporary inconsistencies. For this reason, the environment also provides E-LOTOS specifications used with the purpose of analyzing and verifying the applications aiming at validating the temporal requirements defined by the author.

1 Introduction

The 90's have been known by the use of multimedia applications in several fields of the human activity such as education, medicine and entertainment. These applications have become increasingly sophisticated along the time, and nowadays they are executed in distributed environments, operating transparently in heterogeneous platforms. The possibility of having an application with its media objects dispersed over a network influences the creation and modeling of such applications. Users must supply to the authoring tools information like temporary restrictions, defining acceptable delay thresholds to the elements that compose the system and establishing the presentation of alternative media objects.

The definition of these restrictions is accomplished based on a synchronization model, which dictates the rules about how the media objects of an application can be related in time. Several synchronization models have been proposed [1]. Most of them are both flexible and very expressive. That is the reason why the resulting specifications can be source of incoherences where the logical and temporary consistency of the involved media objects can not be assured. An alternative would be to use directly a formal description technique (FDT) to describe the applications, making possible its analysis and so guaranteeing its consistency. The disadvantage of this direct use, however, is the high complexity inherent to FDTs. So, the need of having a structured high-level model to specify interactive networked multimedia applications becomes evident. The resulting specifications shall then be translated to a FDT so that verification and simulation methods can be applied to them.

In this context, an interactive networked multimedia applications authoring model was created. The MUSE environment was developed to support this model, allowing the user to easily define a multimedia presentation according to the MHEG-5 standard [2]. The adoption of MHEG-5 allows multimedia information to be shared without worrying about the platform or operating system used, providing specification and development of portable applications. To make possible the specifications validation process, the environment generates automatically E-LOTOS specifications. The results obtained from the validation are presented to the author in a quite readable way in the own environment. After the elimination of the incoherences, MHEG-5 applications are generated and can be executed by a MHEG engine. This work is part of DAMD (Distributed Multimedia Applications Design) project, sponsored by the Brazilian research council.

Section 2 presents important aspects to be considered in the applications authoring process, relating them to some multimedia synchronization models pointed by the literature. This section also presents the proposed authoring model. In section 3 basic aspects of E-LOTOS FDT as well as a mechanism to represent in this language, specifications generated by the authoring model are presented. Section 4 illustrates the functionality of the environment and in the section 5, one can read the final considerations.

2 Proposed Authoring Model

The specification of multimedia applications is accomplished with base in three fundamental aspects: logical structuring, establishment of temporal relationships and spatial definition among the elements belonging to the application. The logical structuring is concerned to offer abstraction mechanisms, providing a wide and structural view of the application. The specification of the temporal behavior involves the definition of synchronization relations among the involved elements. The spatial synchronization cares about adjusting the placement of the visible elements according to the output devices (video).

The temporal relations are established according to a synchronization model, which imposes rules on how these elements can relate to each other. Several models have been proposed in the literature. One of the most adopted by existent authoring tools is the time-line based one [3]. However it presents many limitations such as the difficulty both to modularize the application and to establish relations among elements with variable or unknown duration like user interaction for example [4]. Models based on restrictions like HTSPN (Hierarchical Time Stream Petri Nets) [5] and object-oriented models do not present these problems. On the other hand, those models have not been widely used in commercial tools because of the difficulty in understanding the specifications resulting from them.

In this work, an authoring model that joins mechanisms for logical structuring the applications to a synchronization model similar to HTSPN is proposed. The logical structuring level is based on the concept of scenes and groups, providing a broad view of the application. The definition of temporal synchronizations is done in each scene where it is possible to relate and order the media objects in time using a simplified graph. The spatial synchronization allows media objects to be geographically organized considering the output device.

2.1 Logical Structuring

The complexity of multimedia applications increase according to the growth of involved media objects and, consequently, to the several temporal relationships established among them. This is the fundamental reason why the specification of these applications in only one plane is inappropriate. To solve this problem, the concept of scenes was incorporated into the model considering the MHEG-5 standard. Multimedia applications can be organized as a group of scenes related by events, which provide the navigation among them. Each of those scenes can be seen as a black box with an internal behavior that, under certain conditions, enables the presentation of other scenes. In figure 1, one can see an application structured in three scenes: Scene1, Scene2 and Scene3.

The use of this concept, however, does not solve the problem of complexity at all, since a specification with many scenes will be hardly understood. Trying to make easier the understanding of so large applications, a hierarchy mechanism was added to the model by means of the concept of group of scenes.

2.2 Temporal Synchronization

The temporal synchronization of an application, as mentioned previously, refers to the ordering of the presentation of its media objects in time. Each media object has a presentation duration that may or may not be foreseen, depending on its nature. The following topics present how the synchronization relationships can be established.

Basic Synchronization. Media objects can be presented sequentially or simultaneously. In the sequential presentation, the playout of a media object depends on the end of another's. Scene2 in figure 1 models the sequential presentation of a recorded interaction (RI) and three images (P1, P2 and P3). The parallelism, in its turn, considers that media objects start to be presented from a same instant. This is modeled in Scene3, where the animation fragment ANI_P1 is presented in parallel with a button (Interaction).

Event Duration and Delay. A minimum and a maximum time of presentation are associated to each media object. In the case of an image or a text, these values are equivalent because they are time-independent media objects. When one deal with media objects like audio and video, however, it is important to determine both a minimum and a maximum presentation time, since these media objects will be hardly presented at the nominal rate due to problems like network traffic (see figure 1). The representation of these times is given by an interval. It is still important to highlight that media objects without a specified time are presented until a scene change happens.

User Interaction and Scene Transition. User interaction corresponds, for instance, to a button click or an object selection. It is represented in this model as a construction whose presentation time is uncertain, varying between the minimum and maximum values associated to it. When the maximum threshold is reached, the scene continues with its presentation. It is still possible to specify a button without maximum time; in this case, its evolution will only happen after the interaction. Scene3 (figure 1) shows a button (Interaction) with duration of [0,20]. It means that the user will have 20 seconds to select it. When this time elapses, P4 will be presented anyway.

The user interference is normally associated to a scene transition. Transition is the constructor that makes the navigation among scenes possible. Its execution involves both the immediate suspension of the presentation of all the media objects belonging to the current scene and the beginning of a new scene presentation. In Scene1 of figure 1, one can see the transition to Scene2. The transitions are also modeled in both Scene2 and Scene3.

Policies on Synchronization Points Triggering. Synchronization points allow the beginning of the presentation of one or more media objects to be associated to different policies related to the end of the presentation of other media objects that converge for these points. To improve the specification power, the model has adopted some widely commented policies in the literature. They allow the association of different behaviors to the synchronization points [6]. The policies supported by the model are the following:

– *Master*: a synchronization point is triggered when the presentation of a master media object is finished, interrupting all the other ones. The master media object is identified by the presence of the character *m* or the word *master* close to it. This is the policy adopted by the synchronization points of Scene1 (figure 1).

– *Earliest*: the synchronization point is triggered when the presentation of the first media object is finished, interrupting all the media objects that are running simultaneously. Graphically, this policy is represented by the presence of the character *e* or the word *earliest* close to the synchronization point.

– *Latest:* the absence of an indication close to the media object or to the synchronization point means that all the media objects that precede this point will be executed (or they will conclude due to the exhaustion of their maximum presentation time) before the synchronization point is triggered.

Synchronization in Time-dependent Media Objects. The synchronization of time-dependent media objects requires its division in segments, each of them representing a distinct interval in its presentation. The granularity of this division is directly associated to the precision degree wished for the synchronization. Scene1 (figure 1) shows such synchronization between a video (VD) and an audio (AU). Both of them were fragmented in three parts.

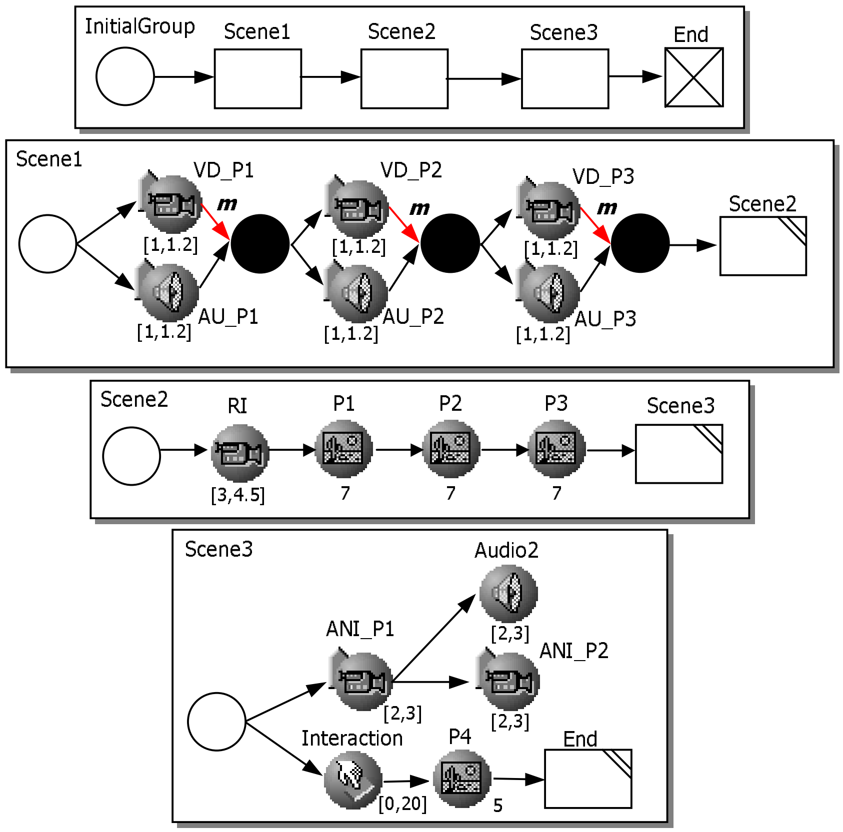


Fig. 1. Representation of the example proposed in [1]

3 Representation of Multimedia Applications in E-LOTOS

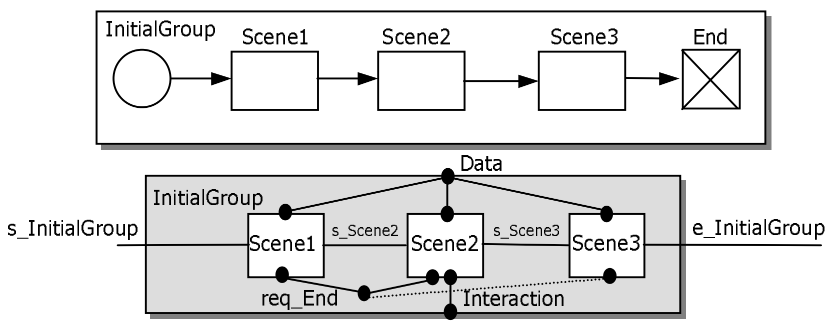
The formalization of specifications is important for the process of their validation. The proposed authoring model, due to its high flexibility and expressiveness, allows both temporally and logically incoherent specifications to be defined. The analysis process detects, for example, conflicts in resources usage and tests if the application end can be reached from all the possible navigation paths. Thus, specifications described by an author according to the model presented in the previous section are translated to a formal representation, analyzed and the obtained results are presented to the user, who will make the necessary adjustments.

The formal description technique E-LOTOS (Enhancements to LOTOS) [7] is an enhanced version of LOTOS and is in standardization process. The main innovation of the language is the incorporation of quantitative time notion, allowing the definition of instants in which actions or events can happen. This is a fundamental feature for representing multimedia applications and for this reason, E-LOTOS was chosen to formally represent them.

The representation of multimedia applications is hierarchical and considers the four essential elements of the authoring model: application, group, scene and media object. All these elements are modeled as processes that evolve according to previously established synchronization relationships. The way of formally represent multimedia applications commented in this section is based on the approach presented in [8]. Further details are presented in the following topics.

3.1 Data Representation and Root Process Instantiation

Data representation is done by means of a library called *classes*, which define data types for all possible media objects. There are types like BitmapClass, StreamClass and SwitchButtonClass, whose definition is based on their respective MHEG-5 classes. For example, the fields of BitmapClass are the media object, its position in the output device and its dimensions. The application is started from the instantiation of the root group process. After that, its evolution is indeed able to occur.



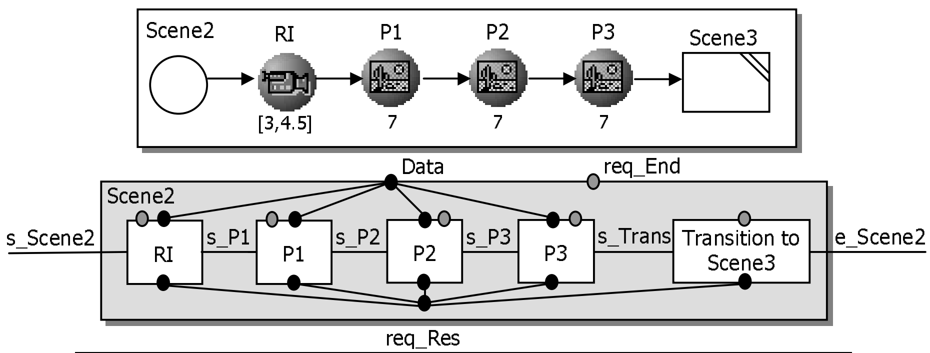

```
process InitialGroupl[s_InitialGroup,e_InitialGroup,Interaction,Data]
  (...,RI:StreamClass,d1:Time,d2:Time,P1:BitmapClass,
   dP1:Time, P2:BitmapClass,dP2:Time,P3:BitmapClass,
   dP3:Time,...):exit is
  hide s_Scene,s_Scene2,s_Scene3, req_End in                              (1)
     s_InitialGroup;
     par s_Scene2#2,s_Scene3#2                                            (2)
       [s_Scene2]→Scene1[s_Scene,s_Scene2,Data,req_End](...)              (3)
       [s_Scene2,s_Scene3]→Scene2[s_Scene2,s_Scene3,Interaction,Data,     (4)
                           req_End] (RI,d1RI,d2RI,P1,dP1,P2,dP2,P3,dP3)
       [s_Scene3]→Scene3[s_Scene3,e_InitialGroup,Data,req_End](...)       (5)
     endpar
     [>req_End;exit                                                       (6)
   endhide
endproc
```

Fig. 2. InitialGroup modeling in E-LOTOS

3.2 Group Representation

In the representation of groups, the hiding operator is used. Taking the example of figure 2, one can see that some internal events like the beginning of both Scene2 (s_Scene2) and Scene3 (s_Scene3) are not visible outside the process (1). These events are used to synchronize the presentation of the scenes belonging to InitialGroup. The synchronization is modeled with the *par* operator (2). For instance, the beginning of Scene2 is associated with the end of Scene1 (s_Scene2) (3 and 4). The same occurs with Scene2 and Scene3: the beginning of the later is synchronized with the end of Scene2 (s_Scene3) (4 and 5).

The disabling operator must also be mentioned (6). As one can observe, the req_End event reaches all the processes of the group; it is used to model the end of the application. When generated (by a transition to end), groups and scenes are terminated with exit.




```
process Scene2 [s_Scene2,e_Scene2,Data,req_End]
  (RI:StreamClass,d1:Time,d2:Time,P1:BitmapClass,dP1:Time,
   P2:BitmapClass,dP2:Time,P3:BitmapClass,dP3:Time)
  hide s_Scene,s_P1,s_P2,s_P3,req_Res in
    loop forever in                                        (1)
      s_Scene2;
        par s_P1#2,s_P2#2,s_P3#2, s_Trans#2
        [s_P1]→RI [s_Scene, s_P1,Data,req_End,req_Res] (RI,d1,d2)
        [s_P1,s_P2]→P1[s_P1,s_P2,Data,req_End,req_Res](P1,dP1))
        [s_P2,s_P3]→P2[s_P2, s_P3,Data,req_End,req_Res](P2,dP2))
        [s_P3,s_Trans]→P3[s_P3,s_Trans ,Data,req_End,req_Res](P3,dP3))
        [s_Trans]→Transition[s_Trans,e_Scene2,req_End,req_Res]()
      endpar
    endloop
    [> req_End;exit
  endhide
endproc
```

Fig. 3. Representation of Scene2

3.3 Scene Representation.

Scene modeling differs in many aspects from group representation. One of the differences is that scene processes instantiate media objects and restrictions instead of groups and scenes. The presence of the *loop* operator in the representation is another important difference (1). It is used to allow a scene to be presented more than once (net structure). Figure 3 shows Scene2 previously instantiated in figure 2.

The event req_Res is responsible for restarting the state of the media objects of the current scene when a transition to another scene happens. Synchronization among media objects is also established with the *par* operator.

3.4 Basic Objects and Temporal Restrictions.

Basic or monolithic objects were defined by Karmouch [2] and model the presentation of simple media objects. These media objects are defined by the occurrence of synchronous (beginning, end) and asynchronous (user interaction) events. Several combinations of these events can be formulated, but only eight are pointed as important in the definition of interactive multimedia scenes. This work presents three of these combinations (see table 1). The fourth object presented in this table (pSsSe - Synchronous start Synchronous end) doesn't appear in [2]. It allows time-dependent media objects like video and audio to be modeled, which have minimum and maximum duration. In the definition of the processes, the Data event was used to represent the presentation of the media object.

E-LOTOS Code	Description
Synchronous start Synchronous end	
process pSsSe[start,end,Data:class] (media:class,d:time):exit is start;Data(!media);wait(d);end@t[t=0];exit endproc	Used to model time-independent media objects like image and text with a presentation known duration.
Synchronous start Asynchronous maximum end	
process pSsAme[start,end,user,Data:class] (media:class,d1,d2:time):exit is start; Data(!media);wait(d1) ; (user@t[t<=d2] [] wait(d2);exit);end@t[t=0] ;exit endproc	Used to model user interaction; if the interaction doesn't occur during the interval [d1,d2] the process is finished when the maximum time (d2) is reached.
Synchronous start Asynchronous end	
process pSsAe[start,end,user,Data:class] (media:class,d:time) : exit is start;Data(!media);wait(d);user;end@t[t=0];exit endproc	Modeling of user interaction without a maximum time to wait defined. The process finishes only once the interaction occurs.
Synchronous start Synchronous maximum end	
process pSsSme[start,end,Data:class] (media:class,d1,d2:time) : exit is start;Data(!media);wait(d);end@t[t≤d2];exit endproc	Used to model time-dependent media objects like audio and video, which have a minimum and a maximum duration defined.

Table1. Representation of basic objects

Figure 4 shows the representation of P2, present in the definition of the scene modeled in figure 3. The event req_End may again be observed, because media objects are also always being executed. In some cases, they may be executed more than once during the presentation of the scene.

The authoring model and consequently the tool, to be described in the next section, provide the definition of three distinct temporal restrictions: WaitMaster, WaitLatest and WaitEarliest. Their E-LOTOS representation controls the end of the media objects that converge to the synchronization point.

```
process P2 [i_P2,f_P2, Data,req_End,req_Res]:exit is
  (P2:BitmapClass,dP2:Time)
  loop forever in                                   (1)
    pSsSe[i_P2, f_P2, Data] (P2,dP2)
    [>req_Res                                       (2)
  endloop
  [> req_End;exit
endproc
```

Fig. 4. Representation of simple multimedia objects

4 The Authoring Environment

The creation environment is divided in two units: media repository and specification area. At any moment, the user can insert media objects into the repository. This is done by browsing a local media object or referencing a remote one. The specification area is composed of the scenes and groups of an application. Each scene is represented by two different views: the temporal and the spatial one. The temporal view allows the user to insert icons and synchronization points and to establish their relationship using arcs. The visible elements, used in the temporal synchronization, can be adjusted in the spatial view.

Figure 5 shows the basic functionality of the authoring environment. The toolbar has shortcuts to its main functions (1). Two support windows can be observed: Specification Hierarchy (2) and Icons Palette (3). The former provides the user a general view of the application, presenting all the scenes and groups in a tree and providing a structured view of the relationships among them. In this case, the modeled application was the example presented in section 2 and is composed, therefore, of three scenes: Scene1, Scene2 and Scene3 (4). The later, in its turn, provides mechanisms to visualize and edit the icons properties. In the same figure, the bitmap icon (P1) of Scene2 is selected and its specific properties are presented in the mentioned window. Icons that have an associated media object (audio, text, image, and video) present a special property called “media”. This property must be filled with a media object existing in the repository. In this example, icon P1 is associated to the media object *Rio Bonito.*

Still in figure 5, the specification of Scene2 scene can be observed (5). It is composed of video (RI) followed sequentially by the presentation of three images (P1, P2 and P3). By the end of the presentation of the last media object, a transition to Scene3 happens. These information are presented by the temporal view. At the same time, there is the spatial view of these elements, which allows the user to organize geographically the visible constitutive elements of the scene (6). It is possible to move or resize them, and their properties related to coordinates and dimensions are automatically updated.

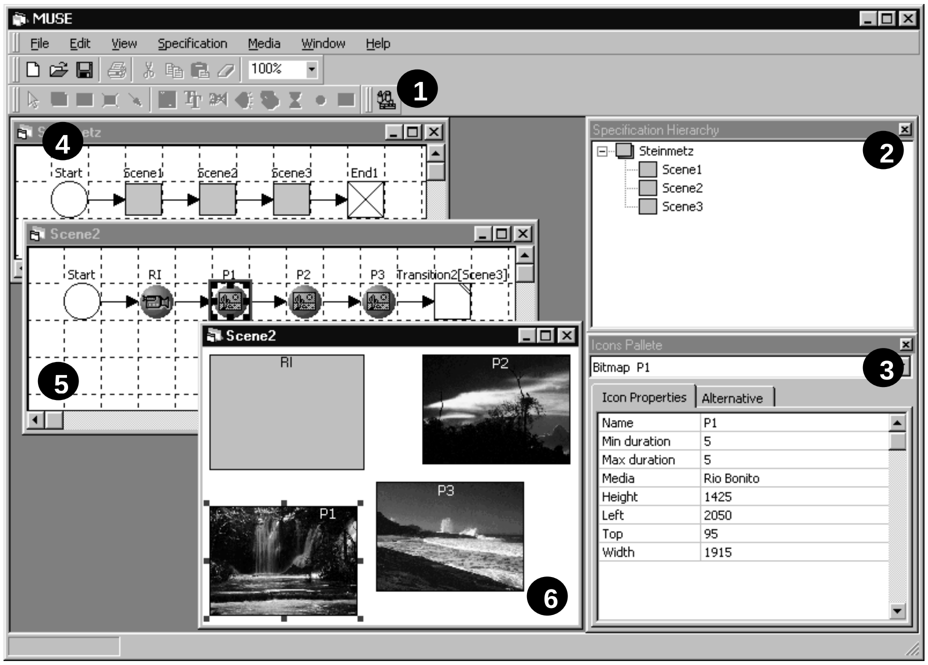


Fig. 5. Graphic interface of the authoring environment

Time-dependent media objects, like video, can be divided in smaller segments, allowing the synchronization of other elements with specific points of them. The environment provides mechanisms that make the process of fragmentation of these media objects easy. MUSE also provides means to reuse scenes and groups. It can be done by browsing the group or scene to be retrieved. It is necessary just to redefine the transitions, defining where the application should evolve to after its presentation. Finally, it is also important to highlight the functionality of E-LOTOS code generation. This is obtained through the special saving option Save as E-LOTOS.

5 Conclusions and Future Work

This work initially proposed a new model for specifying interactive networked multimedia application. Besides, mechanisms for mapping this model to the E-LOTOS language were presented. Finally, the developed environment was described. The main contribution of this work is, therefore, the construction of an environment turned on both ease of use and good expressiveness. At the same time, means to provide the formal representation of applications aiming at its analysis is also a great contribution.

The model proposed distinguishes intentionally the concepts of logical structuring and temporal synchronization. The logical structure of the applications facilitates its organization in chapters, sections or in any other unit. For this reason, the application becomes modular and the temporal behavior of each module is defined. Naturally, the complexity of the parts is much smaller than the whole. Thus, the use of a graph based model based is appropriate, since small graphs will be handled.

The future works include the creation of mechanisms that allow the user to define in the own environment parameters of quality of service, which will be used during the execution of the application. The possibility to define alternative media objects is also an important future task.

It's important to highlight that the use of this environment integrated to the other tools being developed in the project provides a complete framework covering all the steps involved in distributed multimedia applications design: specification, verification and presentation. The ease of the authoring model presented to the user and the use of a formal description technique to validate the applications created by him turn the environment attractive and easy to use without restricting its expressiveness of the environment.

References

1. G. Blakowski and R. Steinmetz. A Media Synchronization Survey: Reference Model, Specification, and Case Studies. IEEE Journal on Selected Areas in Communications, 14(1): 5-35, January 1996.
2. ISO/IEC DIS 13522-5. Information Technology - Coding of Multimedia and Hypermedia Information, Part 5: Support for Base-Level Interactive Applications, 1995.
3. N. Hirzalla, B. Falchuk and A. Karmouch. A Temporal Model for Interactive Multimedia Scenarios. IEEE Multimedia, 24-31, fall 1995.
4. L. Soares e R. Rodrigues. Autoria e Formatação Estruturada de Documentos Hipermídia com Restrições Temporais. In Proc. of the 3rd Workshop on Multimedia and Hypermedia Systems, São Carlos, Brazil, May 1997.
5. P. Sénac, R. Willrich, P. de Saqui-Sannes. Hierarchical Time Stream Petri Nets: A Model for Hypermedia Systems. In Application and Theory of Petri Nets, 1995.
6. P. Sénac, M. Diaz and P. de Saqui-Sannes. Toward a formal specification of multimedia synchronization scenarios. Ann. Telécommun. No. 49, pp 297-314.
7. ISO/IEC JTC1/SC21/WG7. Enhancements to LOTOS. Revised Working Drafts on Enhancements to LOTOS (V4), Project WI 1.21.20.2.3, January 1997.
8. J. P. Courtiat and R.C. de Oliveira. Proving Temporal Consistency in a New Multimedia Synchronization Model. ACM Multimedia, Boston, 1996.