Integration of SMIL in a Partially-Order/Partially-Reliable Based Multimedia Document Retrieval System

M. Joana Urbano, Antonio J. Mendes, Edmundo H. Monteiro
Centro de Informatica e Sistemas da Universidade de Coimbra
{stmaria, toze, edmundo}@dei.uc.pt

Paul D. Amer
University of Delaware
amer@eecis.udel.edu

ABSTRACT
This document proposes to integrate the SMIL language, a recent W3C standard for multimedia documents, into REMDOR, an architecture for multimedia communication being developed by the University of Delaware's Protocol Engineering group. REMDOR is based on the important concepts of Partial-Order/Partial-Reliability at transport level. This paper refers some differences existing between SMIL and REMDOR temporal models, and shows the limitations of the current SMIL specification in accomplishing the transport layer concepts mentioned above.

1. INTRODUCTION
Existing applications are quite different in structure, behavior and quality of service (QoS) requirements. For some of them, the *occasional* lost, duplication and disorder of packets does not constitute a visible performance degradation issue. Others, such as some multimedia applications, do not require the presentation of their media objects to be in a fixed, predetermined order. For all those applications, the use of an ordered/reliable service, as is offered by TCP, could bring unnecessary longer delays, higher buffer utilization and lower throughput. At the other extreme, the use of a unordered/unreliable service, as UDP, does not offer any QoS guaranties to applications. A good solution would be to have a flexible transport service that takes into account the levels of order and reliability needed by applications. The PEL group [1] is currently developing partially-order/partially-reliability (PO/PR) and partially-order/totally-reliability transport services [1, 2, 3, 4] in response to this need.

To show the benefits of a PO/PR transport service for multimedia document retrieval over the Internet, a prototype system named REMDOR [2] was developed by the PEL group. REMDOR consists of a browser, a server and a prototype language for document specification, PMSL [3][1]. PMSL is a declarative language that allows authors to specify a reliability class to each element of a multimedia document, as well as a partial order over all elements. One further step in the REMDOR project, now underway in cooperation with CISUC[4], consists of integrating a standard multimedia language, more specifically the Synchronized Multimedia Integration Language, SMIL [5]. SMIL became a W3C Recommendation in June 1998 and has been looked as a promising model for the authoring of Web multimedia documents.

The aim of the work that will be described here and that is still being developed is to first make SMIL able to express PO/PR concepts and then to show that there is a gain in using extended

---

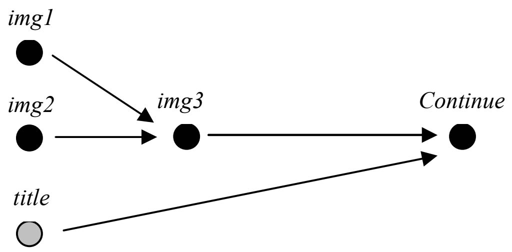
[1] Protocol Engineering Group, University of Delaware, USA.
[2] REmote Multimedia DOcuments Retrieval.
[3] Prototype Multimedia Scripting Language.
[4] Centro de Informática e Sistemas da Universidade de Coimbra, Portugal.

SMIL documents over PO/PR transport networks, compared to networks that do not provide those specific transport services. In this paper, we will exemplify why SMIL 1.0 is currently unable to fully support the important PO/PR concepts, while a proposal for the extension of SMIL is being prepared.

This paper is structured as follows: section 2 describes REMDOR and PO/PR basic concepts, while section 3 takes a brief look over the temporal and synchronization model of SMIL. In section 4, we introduce some scenarios where we illustrate some differences between REMDOR and SMIL temporal models, and show that the current SMIL specification fails to describe some of PO/PR issues. Finally, section 5 presents some conclusions, as well as future and ongoing work.

2. REMDOR AND PO/PR

Current multimedia document models allow the description of logical, temporal, spatial and navigational relations between media objects (text, image, audio, video and animation). REMDOR's document model allows, in addition, the specification of different levels of reliability for each object of a document, as well as the specification of the relative order those objects may appear at presentation time. REMDOR is a multimedia document retrieval system composed of a client, a server and a prototype declarative language, PMSL, that was designed for investigating the benefits of using different transport services for multimedia retrieval over the Internet. The specification of objects' order and reliability in a PMSL document is simple: each media element has a set (possibly empty) of *successors*, i.e., the elements that should follow it in the partial order (PO), and is assigned a level of reliability. Those ideas can be well illustrated with the example shown in figure 1.




```
ELEMENT img1, img3. RELIABLE ...
ELEMENT img2, img3. RELIABLE ...
ELEMENT title, Continue. UNRELIABLE ...
ELEMENT img3, Continue. RELIABLE ...
ELEMENT Continue, TheEnd. RELIABLE ...
```

Figure 1 Example of a PMSL scenario: (a) Precedence graph; (b) simplified PMSL code

Figure 1 shows a precedence graph for the following PMSL scenario: the presentation starts with three image elements, *img1*, *img2* and *title*. Both *img1* and *img2* have as their successor *img3*. In coarse terms, it means that *img3* can not be presented until *img1* and *img2* have been shown. The same way, *Continue* is the successor of both *img3* and *title*, which means it should appear after *title* and *img3* have been presented. Black circles in the precedence graph mean that the correspondent elements are *reliable*, i.e., they have to be presented, no matter how many retransmissions are necessary in case of loss or disruption of the elements. The gray circle means that *title* is unreliable, so, no retransmission will occur if *title* is lost. By the time *Continue* arrives, *title* is declared lost if it has not shown up yet and *Continue* could be presented.

After being authored, a PMSL file is parsed by the **ptpc** parser and translated into another file format, PMFF (figure2). The generated PMFF file contains all order and reliability information of the presentation, in the form of a **service profile**, and also the "ready to go", packaged up data [5]. When a document is requested by a client, the server passes the corresponding PMFF service profile to the transport server and transmits the elements as stated in the PMFF file. In turn, the transport server communicates the server profile to the transport client entity so now the transport layer is able to correctly handle the order and reliability of all elements of the document. At the client side, the client application (browser) simply displays the elements as it receives them from the transport layer [6]. The overall scheme is illustrated in figure 2.

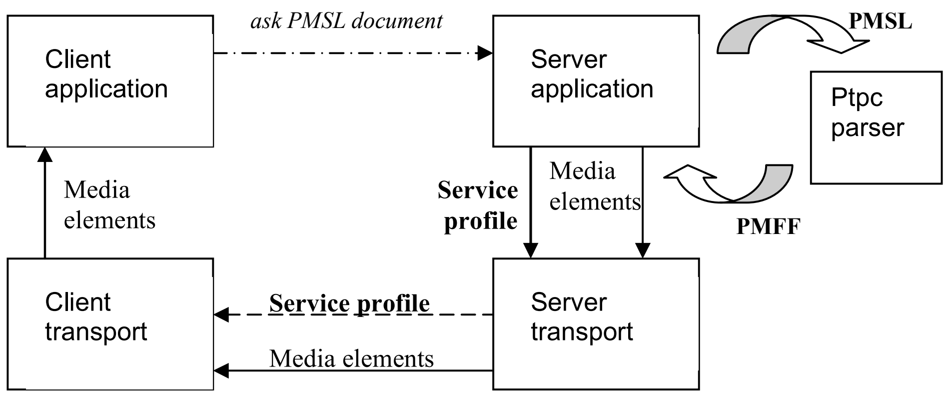


Figure 2 REMDOR architecture with ptpc parser

Our recent research aims to integrate SMIL into the REMDOR architecture, as an alternative language to PMSL. The integration of SMIL in REMDOR is twofold. First, it implies the development of a parser that interprets SMIL 1.0 tags and SMIL 1.0 time model, and translates them into a REMDOR service profile understandable by the transport layer. For SMIL applications to take advantage of REMDOR's expected PO/PR performance gain, they need to be able to easily express order and reliability concepts. We will show in section 4 that this late issue is particularly hard to accomplish, if not impossible, with current SMIL v.1.0 specification. So, an important and already started step on this work is the development of an extension of SMIL, according to [5] recommendations.

3. SMIL

SMIL is the W3C standard for Web multimedia documents. SMIL is a simple, declarative language whose syntax is defined by an XML [7] Document Type Definition (DTD).[8] It is similar to HTML [8], though it adds to HTML a new and important dimension: time. With SMIL, authors can define what, where and when to present media objects, such as text, image, audio, video and animation, at the screen, as well as provide ways to precisely define synchronization between those objects.

---

5 Images and streaming elements, such as audio and video, are here further divided into smaller cells that are interleaved according to the presentation scenario.
6 Actually, the client is more complex, using a feature called *explicit release* to acquire graceful degradation [1].
7 eXtensible Markup Language [6].
8 The XML DTD for SMIL 1.0 is given at [7].

Most of the synchronization power of SMIL is concentrated in two SMIL elements: *par*, that groups elements to be played simultaneously (i.e., in parallel), and *seq*, that groups elements to be played in sequence. The conjugation of several, possibly nested, *par* and *seq* elements, in conjunction with temporal attributes, such as *begin*, *end*, *dur* and *endsync*, allows for the description of complex multimedia scenarios. We present next a simple example to show the use of SMIL tags and attributes, as well as parallel and sequential blocks.

```
1. <par endsync="id(aud)">
2.   <img id="logo" fill="freeze"/>
3.   <img id="title" end="10s"/>
4.   <audio id="aud"/>
5.   <seq id="seqBlck" begin="5s">
6.     <img id="img1"/>
7.     <img id="img2" dur="10s"/>
8.   </seq>
9. </par>
```

Lines 1 through 9 define a parallel block. All children of that block must start playing simultaneously, unless some explicit timing information is defined. That way, we can see that three of the children of the par block (*logo*, *title*, and *aud*, lines 2-4) start simultaneously, while the fourth child (*seqBlck*, line 5) starting time is delayed for 5 seconds. The attributes *begin* (line 5), *end* (line 3) and *dur* (line 7) allow for the definition of the explicit beginning, the explicit end and the explicit duration of one or a group of media elements, respectively. The *fill* attribute (line 2) here says that the element *logo* will be visible until the end of the parallel block. Finally, the attribute *endsync* (line 1) of the *par* element tells that the effective end time of the parallel block equals the effective end time of the audio element (*aud*). The timeline corresponding to the SMIL code showed above, assuming an intrinsic duration of 18s for the audio element, is pictured in figure 3.

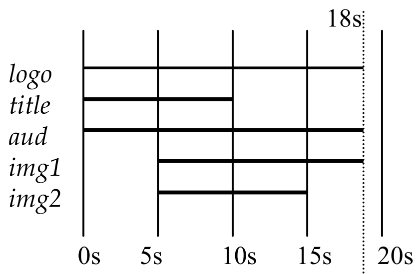


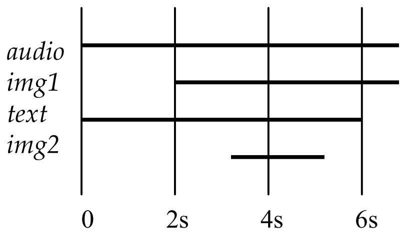
Figure 3 Timeline representation for the temporal and synchronization relations between media elements

4. SMIL AND REMDOR

The major transformation REMDOR requires to support SMIL is the building and inclusion of a parser to translate SMIL 1.0 documents into a PMFF file (service profile plus "ready to go" media elements). This apparently simple task is non-trivial though, due to important differences between the spatial, temporal and synchronization models of SMIL and REMDOR. This section will concentrate basically on temporal differences between SMIL and REMDOR, and also on the major limitations of SMIL when trying to accomplish partially-order and partially-reliability concepts. Some examples will be given to better explain those differences and limitations. Implementation issues related to the incorporation of SMIL in REMDOR system (as, for example, the ones related to the building of the SMIL parser), as well as the spatial and navigational differences between SMIL and REMDOR are out of the scope of this article.

4.1 SMIL and REMDOR temporal models

The first example illustrates how different SMIL and REMDOR deals with “fine granularity” synchronization. Consider the timeline scenario below:

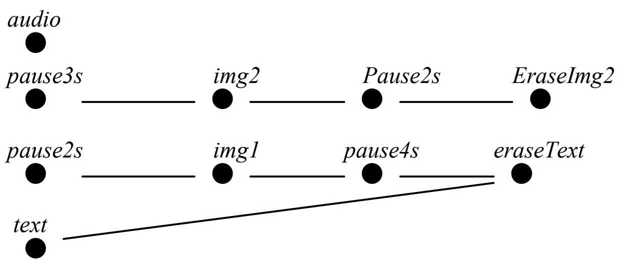


Example 1 Timeline representing explicit temporal relations between media elements

This scenario is easily translated into SMIL 1.0 using the explicit temporal attributes *begin*, *end* and *dur*:

```
<par>
   <audio id="audio"/>
   <img id="img1" begin="id(audio)(2s)"/>
   <img id="text" end="id(img1)(4s)"/>
   <img id="img2" begin="id(audio)(3s)" dur="2s"/>
</par>
```

However, REMDOR deals with explicit synchronization between elements in a rather different manner, making use of the PAUSE element to specify elapsed time between elements starting and ending time. The PMSL precedent graph for the current scenario and the correspondent PMSL (simplified) code is pictured in figure 4. Notice the use of ERASE elements (*eraseImg1* and *eraseText*) to explicitly specify the end time of PMSL elements.



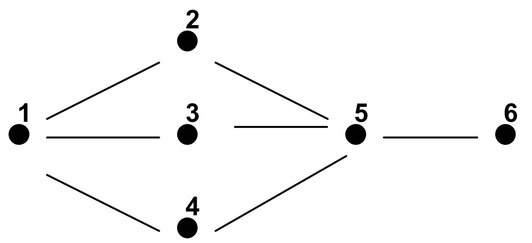

```
ELEMENT audio: theEnd.           RELIABLE
ELEMENT pause3s: img2.           RELIABLE
ELEMENT pause2s: img1.           RELIABLE
ELEMENT text: eraseText.         RELIABLE
ELEMENT img2: Pause2s.           RELIABLE
ELEMENT Pause2s: eraseImg2.      RELIABLE
ELEMENT img1: pause4s.           RELIABLE
ELEMENT pause4: eraseText.       RELIABLE
ELEMENT eraseImg2: theEnd.       RELIABLE
ELEMENT eraseText: theEnd.       RELIABLE
```

Figure 4 REMDOR precedence graph for example 1 and correspondent simplified PMSL code

As can be seen from the precedent graph above, the timeline scenario of example 1 is not fully accomplished with REMDOR temporal model. For instance, we see that *audio*, *pause3s*, *pause2s* and *text* are shown in parallel, which in REMDOR simply means that those elements **can** be displayed at same time; however, nothing forces the elements to be strictly played simultaneously. So, if the *audio* element starts after *pause2s*, for example, *img1* could not be presented 2 seconds after the beginning of *audio*, as required.

4.2 SMIL and partial order concepts

The next example is illustrated by the precedence graph and correspondent PMSL simplified code shown below, and represents a common REMDOR partial order scenario.



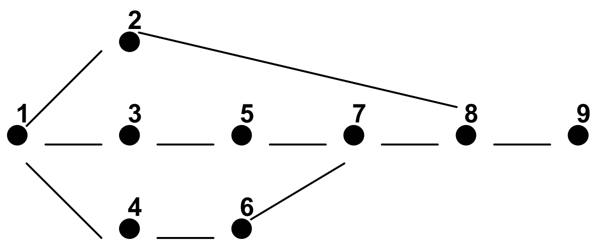
ELEMENT 1: 2, 3, 4.	RELIABLE
ELEMENT 2: 5.	RELIABLE
ELEMENT 3: 5.	RELIABLE
ELEMENT 4: 5.	RELIABLE
ELEMENT 5: 6.	RELIABLE
ELEMENT 6: END.	RELIABLE

Example 2 Limitation of SMIL synchronization model in order to specify REMDOR's successor-based partial order

Although simple, this scenario is not easily specified with SMIL 1.0. In fact, SMIL's synchronization model is based on the combination of *par* and *seq* elements, which means that each element is inserted directly in either a sequential or a parallel block. Consider elements 2, 3 and 4. They have to be presented after element 1 and before element 5, however, there is no explicit indication whether they might be presented in sequence or in parallel. In reality, neither of these two options is suitable to describe the partial order of these objects. On one hand, presenting them in sequence is forced: which one of the 6 possible sequences should be chosen? Also, if the transport layer delivers the elements in an order different to the one chosen, some elements must be buffered and the potential partial order benefits of the scenario would be lost. On the other hand, grouping elements 2, 3 and 4 in a parallel block would tell the player to start playing the elements *simultaneously*, even though it would probably imply extra delay or even stopping the presentation.

One possible solution we coarsely presents here is the redefinition of *par* element, that would make it possible to authors to choose between a strict or a "loose" *par* element. This way, the **strict** *par* element would be similar to SMIL 1.0 *par* element, while the **loose** *par* element would allow the specification of a more flexible parallel group of elements. Consider the scenario of example 2. The definition of a loose parallel group with elements 2, 3 and 4 would tell the player that the elements could be presented simultaneously, if they would arrive at the same time, **or** in a more flexible, undetermined order, so far they are presented after element 1 and before element 5. The proposed redefinition of *par* element can be done by defining two new elements (e.g., *spar* and *lpar*, for strict and loose parallel elements) instead of current *par* element, or simply by defining a new attribute to the *par* element. This attribute (e.g., the *start* attribute) would have "strict" and "loose" values, being "strict" the default value.

Example 3 shows another partial order scenario. It is followed by 3 proposals to the translation of the scenario into SMIL (simplified) code:



Example 3 Another scenario of partial order

Proposal 1

```
1. <seq>
2.  <ref id="1"/>
3.  <par>
4.    <ref id="2"/>
5.    <seq>
6.     <ref id="3"/>
7.     <ref id="5"/>
8.     <ref id="7"/>
9.     <ref id="8"/>
10.    </seq>
11.    <seq>
12.     <ref id="4"/>
13.     <ref id="6"/>
14.    </seq>
15.  </par>
16.  <ref id="9"/>
17. </seq>
```

Proposal 2

```
1.  <seq>
2.   <ref id="1"/>
3.   <par id="par1">
4.     <ref id="2"/>
5.     <ref id="3"/>
6.     <ref id="4"/>
7.   </par>
8.   <par>
9.     <ref id="5"/>
10.    <ref id="6"/>
11.  </par>
12.  <ref id="7"/>
13.  <ref id="8"/>
14.  <ref id="9"/>
15. </seq>
```

Proposal 3

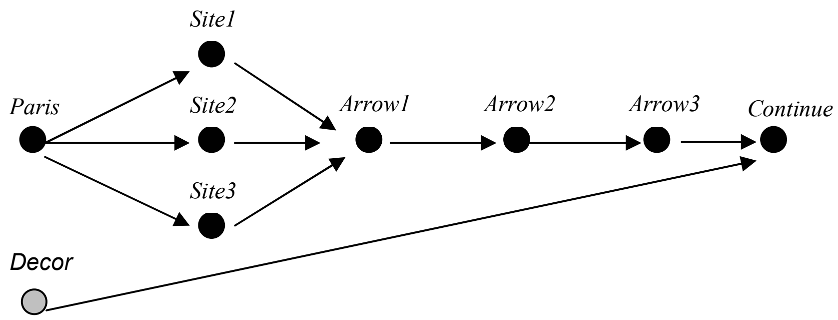
```
1.  <par>
2.   <ref id="1"/>
3.   <ref id="2" begin="id(1)(end)"/>
4.   <ref id="3" begin="id(1)(end)"/>
5.   <ref id="4" begin="id(1)(end)"/>
6.   <ref id="5" begin="id(3)(end)"/>
7.   <ref id="6" begin="id(4)(end)"/>
8.   <ref id="7" begin="id(5)(end)"/>
9.   <ref id="8" begin="id(7)(end)"/>
10.  <ref id="9" begin="id(8)(end)"/>
11. </par>
```

Proposal 1 fails to unequivocally specify that element 7 must follow elements 5 and 6. In fact, the partial order over those elements would be violated if the effective duration of element 4 extends the effective duration of the sum of elements 3 and 5. Proposal 2 correctly specifies the order of elements but presents some limitations: it unnecessarily groups elements 2, 3 and 4 in a parallel group (as discussed in example 2), and also limits the presentation of element 6 to be after the end of *par1*, even if element 4 has already reached its end. Finally, proposal 3 presents a neat and condensed code, although it fails to specify that element 8 should be presented after element 7 **and** element 2. Moreover, the use of the attribute *begin* only allows to specify the beginning of an element relating it to a well defined temporal moment of another element (as its begin, end or an explicit value of its playback time). Consider, for example, line 4 of proposal 3. We would like to specify that element 3 must follow element 1, but without referring a *determined* event-value. Indeed, in some partial order scenarios, it would be very useful to specify indeterminate time events with SMIL. One possible solution to this specific problem would be to consider in SMIL specification another element-event value as, for example, the *afterbegin* value. This way, line 4 of proposal 3 could be rewritten as illustrated below, meaning that element 3 should follow element 1 anytime after the starting time of element 1.

```
<ref id="3" begin="id(1)(afterbegin)"/>
```

4.3 SMIL and partial reliability concepts

The last example includes some issues already referred in precedent scenarios and adds a new component: *reliability*. It consists of a presentation and starts with a full screen picture, *Paris*. Next, three images, *Site1*, *Site2* and *Site3*, will be displayed over *Paris*; the order of presentation of those three images is intentionally **not** pre-defined, in order to take profit of partial order benefits. *Arrow1*, *Arrow2*, *Arrow3* follow in sequence. *Continue* is a button that, when pressed by users, will erase all elements from the screen. In addition, there is a merely decorative element, *Decor*, that is partially reliable; that means that it is useful until the user presses *Continue* but is considered lost if it hasn't arrived until that moment, enabling the presentation to proceed without delay.



Example 4 Limitation of SMIL synchronization model in order to specify REMDOR's reliability classes; partially-reliable element *Decor* is filled in gray

One possible implementation of this scenario follows:

```
<par>
   <img id="paris" region="globe" fill="freeze"/>
   <par id="sites" begin="id(paris)(begin)">
     <img id="Site1" fill="freeze"/>
     <img id="Site2" fill="freeze"/>
     <img id="Site3" fill="freeze"/>
   </par>
   <img id="Arrow1" begin="id(paris)(2300ms)" dur="2s"/>
   <img id="Arrow2" begin="id(Arrow1)(end)" dur="2s"/>
   <img id="Arrow3" begin="id(Arrow2)(end)" dur="2s"/>
   <ref id="Continue" begin="id(Arrow3)(end) " fill="freeze"/>
   <ref id="Decor" fill="freeze"/>
</par>
```

This solution presents some limitations: similar to what happened in precedent examples, the partial order over *Site1*, *Site2* and *Site3* cannot be fully achieved. Also, the solution fails to specify an *undetermined* time between the beginning times of *Sites1* and *Arrow1*, so the concept of flexible successors is not achievable. Finally, the code above doesn't express the partially reliable nature of *Decor*. Whatever combination of *par* and/or *seq* SMIL elements we do, there is

no possibility to specify that the effective begin of *Decor* could be anytime until *Continue* is pressed.
Our proposal for the assignment of reliability classes with SMIL consists of the addition of the attribute *reliability* for media elements. This attribute would have a "reliable" value as default. However, the way reliability classes can be defined in the SMIL DTD is object of further study.

5. CONCLUSION AND FUTURE WORK

In this paper we referred REMDOR, a multimedia document retrieval PO/PR-based system. Our assumption is that if authors are able to specify in their multimedia documents a level of reliability for each media element, as well as a partial order over the elements of the document, then those documents could benefit of PO/PR transport services when transmitted over an unreliable and/or congested network. We then took a brief look over temporal issues of SMIL, the new W3C standard for Web multimedia documents, and showed the major problems that will arise with the adoption of SMIL in REMDOR, due to the different time and synchronization models presented by them. In particular, we identified three different problems: first, there is a compromise between the gain that could be obtained with the flexibility provided by partial order and the degree of synchronization that could exist between elements. As a consequence, strict clock-value synchronization between elements is hard to specify in REMDOR without loosing the benefits of partial order. Second, the fact that SMIL 1.0 synchronization model relies in basically two tags, *seq* and *par*, implies that in some cases SMIL is unable to correctly specify a partial order over the elements. Third, the current specification of SMIL doesn't allow for an easy and effective assignment of classes of reliability to elements of a document. Finally, we suggested some solutions for the limitations mentioned above. These solutions will be further developed and will be incorporated in a general proposal for the extension of SMIL that we intend to submit to the Synchronized Multimedia Working Group of the World Wide Web Consortium.

The next steps in this work include the building and inclusion of the SMIL parser at REMDOR, as well as the development of the proposal to the extension of SMIL 1.0 mentioned above. We also preview the execution of an extensive phase of experiments with the extended system upon several link scenarios.

6. ACKNOWLEDGMENTS

This work is supported by The Advanced Telecommunications/Information Distributed Research Program (ATIRP) Consortium sponsored by the U.S. Army Research Laboratory under the FEDLAB Program (DAAL 01-96-2-0002) and by Fundação para a Ciência e Tecnologia (FCT), Programa PRAXIS XXI.

7. REFERENCES

[1] P. Conrad, E. Golden, P. Amer, R. Marasli, "A Multimedia Document Retrieval System Using Partially-Ordered/Partially-Reliable Transport Service", Proceedings of Multimedia and Computing Networking 96, San Jose, January 1996.

[2] P.Conrad, P.Amer, E.Golden, S.Iren, R.Marasli, A.Caro, "Transport QoS over Unreliable Networks: No Guarantees, No Free Lunch!", 5th Workshop on QoS, Columbia University, May 1997, 315-318

[3] R.Marasli, P.Amer, P.Conrad, "Partially Reliable Transport Service", Proceedings of IEEE ISCC '97 - 2nd Symposium on Computers and Communications, Alexandria, Egypt, July 1997

[4] P. Conrad, P. Amer, M. Taube, G. Sezen, S. Iren, A. Caro, “Testing Environment for Innovative Transport Protocols”, Proceedings of MILCOM '98, Boston, October 1998
[5] Synchronized Multimedia Integration Language (SMIL) 1.0 Specification, W3C Recommendation 15-June-1998, http://www.w3.org/TR/REC-smil/
[6] Extensible Markup Language (XML) 1.0, W3C Recommendation 10-February-1998, http://www.w3.org/TR/1998/REC-xml-19980210
[7] J. Ossenbruggen, S. Mullender, XML document type definition (DTD) for SMIL 1.0, http://www.w3.org/TR/REC-smil/SMIL10.dtd
[8] HTML 4.0 Specification, W3C Recommendation, revised on 24-Apr-1998, http://www.w3.org/TR/REC-html40/