Navigating with Inheritance in Hypermedia Presentations
Virginie Sans and Dominique Laurent
ETIS UMR CNRS 8051
University of Cergy Pontoise
2 avenue Adolphe Chauvin
95302 Cergy Pontoise Cedex - France
+(33)1 34 25 66 19
{virginie.sans, dominique.laurent}@dept-info.u-cergy.fr

ABSTRACT
Due to significant improvements in networking, providing multimedia presentations has become a real commercial issue and a technical challenge for Web TV, e-learning, commercial presentations. Hyperlink in HTML has been designed for hyperdocuments navigation, but the actual use of hyperlinks lacks of relevant features to play continuous media such as videos. The main two issues in using hyperlink and multimedia in a web context are:

(a) Navigation and continuity of streamed hypermedia. In a multimedia presentation, when switching from one point to another, if the continuity of streamed media is required, then a video should not stop.

(b) Adaptability of the presentation in response to the user’s reaction. In our approach, we consider spatial regions whose positions can change according to the user’s wishes, as well as spatial regions that can appear in the presentation and preserve the continuity of streamed media.

This paper addresses these two issues by considering an object oriented vision of a multimedia presentation structure. We propose a new XML based language for multimedia presentations, called ARM. In this multimedia document format, we introduce the notions of spatial and temporal inheritance, and of overloading. Spatial inheritance together with overloading allows to keep the same spatial structure between two or more documents, or to change this spatial shape in response to the user’s activity. Additionally, the temporal inheritance allows the continuity of streamed media even in the case of multiple hyperlinks (i.e., hyperlinks with more than one target).

Categories and Subject Descriptors
H.4.3 [Communications Applications]

General Terms
Multimedia presentation, synchronization, hyperlink, hypermedia, WWW, Flash communication server

Keywords
ARM, Advanced Rich Media, Inheritance, Overloading, Overdefining, Spatial Inheritance, Temporal Inheritance

1. INTRODUCTION
Due to significant improvements in networking, providing multimedia presentations has become a real commercial issue and a technical challenge for Web TV, e-learning, commercial presentations. Many applications can be built for using multimedia presentations, in particular through the Internet. However, in this context, some issues have to be carefully considered. We list below the most important ones and the way we take them into account in this paper.

(a) The compatibility of the presentation with the client system. Different formats exist, and thus, the user must adapt his computer configuration to the application. Our framework being meant to reach as many users as possible, we chose as a system requirement the Macromedia Flash technology which is the most commonly used currently, if compared with Java or SMIL.

(b) The reuse of preexistent standards. In the web context, XML has become a standard to exchange information between different applications. Consequently, our language, called ARM, is based on the XML syntax.

(c) The performance of a presentation played over the Internet. Streaming media over the Internet implies that parsing a multimedia document should be reduced to the minimum, so that resources are used for the scheduling and the synchronization of media in the presentation. To cope with this problem, we reuse media that are already stored in the buffer or in the cache, instead of reloading them. To know which media to preserve in the buffer or in the cache, we introduce inheritance.

(d) The maintenance of the applications must be as easy as possible, especially when considering multiple hyperlinks. In our model, a given multiple hyperlink is specified in a single XML document, even if the hyperlink is shared by several documents.

In this paper, we propose a solution to improve navigation in multimedia presentations. For a good navigation in a multimedia presentation, we must first assume the continuity of media when
navigating. For example, let us consider the situation presented in Fig.1. The multimedia presentation is composed of two documents $H_1$ and $H_2$, a document being a possibly accessible screen in the presentation. Both documents have three regions in which media are displayed, and these regions have relative positions and sizes. The two documents $H_1$ and $H_2$ display the same video in region $R_3$. The only difference between the two documents is that different media are displayed in region $R_3$. Therefore, when switching from one document to the other, to have a real fluid navigation, the video should be displayed without interruption in region $R_1$.

![Fig. 1 – Two multimedia documents](image)

The other issue in multimedia presentations is the spatial repartition of the regions in the presentation and their ability to change. Let us consider the switch from $H_1$ in Fig 1 to $H_2$ in Fig 2. $H_1$ and $H_2$ contain three regions that display the same media, but these regions do not have the same spatial repartition. Since current solutions use fixed regions that cannot be changed, switching from $H_2$ to $H_1$ is not possible without restarting the video from the beginning. The new format presented in this paper defines a spatial inheritance and an overloading process that enable to reshape regions while keeping unchanged the playing of the media. Consequently, using our approach, switching from $H_1$ to $H_2$ becomes possible.

In our approach, we propose an XML based format for multimedia presentations, in which it is possible to specify how to preserve the continuity of media between documents by means of what we call temporal inheritance.

The implementation of our approach is based on an XML format for multimedia presentations and a third-part architecture: a streaming media server, a Flash player, and a hypermedia document server.

The paper is organized as follows: Section 2 presents the basics of our approach. Section 3 deals with the ARM language and Section 4 describes the architecture of our implementation. Section 5 discusses the synchronization issues and how inheritance can be used to that purpose. Section 6 presents conclusions and future work.

![Fig. 2 – The change of spatial shape of the presentation](image)

## 2. THE PROPOSED APPROACH

### 2.1 Basic Notions

A multimedia document consists of a set of regions in which media are displayed, and more generally, a multimedia presentation is a set of multimedia documents, among which the user can navigate through hyperlinks, based on the notion of scenario.

In order to formalize these notions, a region is seen as a pair of the form $<R, d>$ where $R$ is a unique identifier associated to a given region and where $d$ is a tuple of attribute values that describe where and how to display the region. In particular, these attributes define the absolute position in the screen, the width and the height of this region. For the sake of simplification, a region $<R, d>$ will be simply denoted by its identifier $R$ when the attribute values in $d$ are not needed.

A flow $F$ is a set of media such as video, animation, picture, etc. In a flow, the diffusion of media are events and these events are organized using the qualitative relations of Allen’s algebra [1], built up using the primitives before ($<$), after ($>$), during ($d$), contains ($d_i$), overlaps ($o$), overlapped-by ($oi$), meets ($m$), met-by ($mi$), starts ($s$), started-by ($si$), finishes ($f$), finished-by ($fi$), and equals ($=$).

#### Definition 1

A hypermedia document is a triple $(R, F, A)$ where:

- $R$ is a set of regions
- $F$ is a set flows
- $A$ is a set of pairs of the form $(R, F)$ where $R$ is in $R$ and $F$ is in $F$. Such a pair is called an association.

We note from Definition 1 above that a given region $R$ appears in at most one association of $A$. Thus, when a region $R$ is associated to a flow $F$, this region cannot be associated with another flow $F'$. In this case, $R$ is said activated, meaning that it appears on the screen, and that the associated flow $F$ is displayed on the screen. When a region $R$ is not involved in an association of $A$, then $R$ is said inactive, meaning that $R$ cannot be seen on the screen.

#### Definition 2

Let $H$ be a set of hypermedia documents and let $H_0$ be in $H$. A presentation $P$ is a pair $(H_0, G)$, where $G = (H, L)$ is a graph such that:

- $L$ is a set of hyperlinks, seen as pairs $(H_i, H_j)$ where $H_i, H_j$ are in $H$.
- $L$ contains at least one pair $(H_0, H)$ where $H \in H$, and $L$ contains no pair $(H, H_0)$, for any $H$ in $H$.
- $H_0$ is called the start document of $P$.

#### Definition 3

Given a presentation $P = (H_0, G)$, a scenario $\sigma$ is a path in $G$ starting from $H_0$.

Intuitively, a presentation specifies the possible navigations through hyperlinks in a given set of hypermedia documents, whereas a scenario defines one such navigation.

As an example, the multimedia documents $H_1$, $H_2$ shown in Fig. 1 can be defined as follows:

- $H_1 = (R_1, F_1, A_1)$ with $R_1 = \{<R_1, d_1>, <R_2, d_2>, <R_3, d_3>, \}$, $F_1 = \{F_1, F_2, F_3\}$ and $A_1 = \{(R_1, F_1), (R_2, F_2), (R_3, F_3)\}$. 

1386
For spatial inheritance, given two documents \( H_1 = (R_1, F_1, A_1) \) and \( H_2 = (R_2, d_1, A_2) \) with \( R_2 = \{ <R_1, d_1>, <R_2, d_2>, <R_3, d_3> \} \), \( F_2 = \{ F_1, F_2 \} \) and \( A_2 = \{ (R_1, F_1), (R_2, F_2), (R_3, F_3) \} \).

We may consider \( F_1 \) as a flow composed of a single media called video.flv. We can also notice that in our example, in \( H_1 \), \( R_1 \) has an association with flow \( F_1 \), whereas the association changes in \( H_2 \).

Denoting by \( H \) the set \( \{ H_1, H_2 \} \), \( P = (H, G) \) (where \( G = (H, (H_1, H_2), (H_2, H_1)) \)), is a presentation that allows only to switch either from \( H_1 \) to \( H_2 \) or from \( H_2 \) to \( H_1 \). Notice that \([ (H_1, H_2), (H_2, H_1)) \) is a possible scenario, whereas \([ (H_2, H_1), (H_1, H_2)) \] is not.

On the other hand, the multimedia document \( H_3 \) shown in Fig. 2 is defined as follows:

- \( H_3 = (R_3, F_3, A_3) \) with \( R_3 = \{ <R_1, d_1>, <R_2, d_2>, <R_3, d_3> \} \), \( F_3 = \{ F_1, F_2 \} \), and \( A_3 = \{ (R_1, F_1), (R_2, F_2), (R_3, F_3) \} \).

We note that the tuples of attribute values associated to \( R_1 \) and \( R_3 \) have changed from document \( H_1 \) to \( H_3 \), since the shapes of these regions are not the same in the two presentations.

### 2.2 Inheritance and Overloading

According to [4], time used to create multimedia documents can really decrease with the reusability. Reusability can be achieved using the notions of inheritance and overloading. In this paper, we define two kinds of inheritance, namely the spatial inheritance and the temporal inheritance, while overloading allows for partial spatial inheritance.

For spatial inheritance, given two documents \( H_1 = (R_1, F_1, A_1) \) and \( H_2 = (R_2, F_2, A_2) \) with the same set of regions (i.e., \( R_1 = R_2 \)), we can switch from \( H_1 \) to \( H_2 \) without having to parse the definition of the regions. We do so by indicating in the definition of \( R_3 \) that the regions spatially inherit from those of \( R_1 \). In this way, resources are saved for the processing of streaming and of sequencing, instead of loosing time parsing information that have already been considered.

Moreover, in the case of spatial inheritance, and in order to adapt the display of the regions to the user's interaction, it is possible in our approach to overload the definition of some regions in \( R_3 \). This means that the identifiers of these regions are preserved, but that their attribute values are changed according to the user's wishes. In an on-demand application, we can also dynamically add regions with this principle. In the case of overloading with spatial inheritance, only the overloaded regions are parsed, since the previously defined regions are temporarily stored by the player in a cache.

On the other hand, the continuity of the media between \( H_1 \) and \( H_2 \) is achieved by temporal inheritance. Temporal inheritance concerns the association between regions and flows. With temporal inheritance, when switching from a document \( H \) to another document \( H' \), it is possible to "preserve" an association \( (R, F) \) in such a way that the flow \( F \) is continuously displayed.

In our example, if the association between \( R_1 \) and video.flv is defined in \( H_3 \) as temporarily inheriting, then video.flv continues its diffusion when switching from \( H_1 \) to \( H_2 \). Notice that, consequently, our solution allows multi-directional linking.

### 2.3 Related work

In this section, we review the approaches of SMIL, XLink and Hytime and show their limitations.

#### 2.3.1 SMIL

The SMIL 1.0 proposition [8] was recommended by the W3C. Our model is based on the same markup framework to be coherent with this reference. However, SMIL lacks of ability for a real navigation between documents.

Indeed, with SMIL, when switching from a document \( H \) to another document \( H' \), two possibilities are offered to the user by means of the attribute show of a SMIL link. The two values of this attribute are either replace, meaning that \( H \) is stopped and \( H' \) is started, or pause, meaning that \( H \) is temporally stopped to display \( H' \) and then, \( H \) is resumed at the point it was stopped.

If the link is defined using the replace value in the case of our example (see Fig. 3), video.flv (displayed in \( R_1 \)) is restarted from the beginning when \( H_2 \) is run. As will be seen next, our approach allows either to restart or to continue the display of the video when switching from document \( H_1 \) to document \( H_2 \).

In SMIL, the targeted document replaces the calling document thus SMIL allows the diffusion of multimedia documents where associations are not preserved from one document to the other. Even when using the pause option of the presentation \( H \) until presentation \( H' \) terminates, SMIL does not allow a real fluid navigation, because the target of a hyperlink target is always the whole screen (i.e., the set of all regions). This implies the full reconstruction of associations, which prevents the media from being continuously displayed. We note that, in the last specification SMIL 2.0 of SMIL, it is possible to move or to resize regions only in the case where the changes are specified in a predefined animation. Such changes are also possible in our approach, in the more flexible case when users directly interact with the presentation.

#### 2.3.2 XLink

Many XML applications require linking features more sophisticate than those provided by SMIL. XLink [9] addresses these requirements by allowing multiple links construction. In XLink, a given set of flows \( \{ F_1, \ldots, F_n \} \) can be associated to a set of regions \( \{ R_1, \ldots, R_n \} \), through a hyperlink of the form \( \langle F_1, \ldots, F_n \rangle \rightarrow \langle R_1, \ldots, R_n \rangle \), meaning that, for \( i = 1, \ldots, n \), the flow \( F_i \) should be displayed in region \( R_i \).

In this way, a hyperlink is defined only when some associations change but not all. Moreover, the continuity of the diffusion of flows in unchanged associations is preserved when navigating through XLink hyperlinks. Notice however that this hyperlink processing has two main drawbacks. First, the displayed document does not change, meaning that it is not possible to switch from one document to another. Second, flows must contain one single media, and thus, for instance, a link involving a flow consisting of a video followed by an animation is not possible with XLink.

#### 2.3.3 HyTime

HyTime [3] provides a rich set of indirect addressing mechanisms, by means of multi-directional links and multi-ended links. HyTime is based on an SGML syntax, whose main drawback is that it is very heavy to parse in a web context.

More generally, without going into the details of HyTime heavy syntax, even if multi-directional links are possible, the presentation has pre-orchestrated scenarios, in which regions are predefined and cannot be changed. In particular, the size and the
spatial position of a region cannot be changed during the presentation if the continuity of the display of the associated media is to be preserved. In our example, the region $R_1$ does not have the same attribute values in $H_1$ and $H_3$. Thus, in none of the approaches recalled above, it is possible to have a link from $H_1$ to $H_3$ so that the video in $R_1$ is continuously displayed when switching from $H_1$ to $H_3$.

### 2.3.4 Reset and Continuous Modes

To summarize the features of the approaches described above and compare them with our proposal, we introduce two modes, called the **reset mode** and the **continuous mode**, as possible ways to display a flow when activating a hyperlink.

- **The reset mode.** This mode implies that the activation of a hyperlink resets the entire presentation. This can be achieved in two ways when activating a hyperlink: (a) replace the document by another one, or (b) fully reload the document. Thus, the reset mode does not preserve the continuity of the display of the media, even in case (b).

- **The continuous mode.** This mode implies that the activation of a hyperlink preserves the continuity of the display of the media. This can be defined by: (c) a target link as in XLink or (d) an inheritance principle as presented in this paper.

### 3. THE ARM LANGUAGE

ARM (Advanced Rich Media) is an XML based descriptive format. Its declarative syntax is based on the SMIL format but it enables more features regarding the synchronization of the media during the navigation. The ARM format incorporates basic hypermedia principles such as spatial layout, temporal composition, synchronization and navigational hyperlinking. An ARM file has a **head** part and a **body** part presented in Fig. 4 and Fig. 5, respectively.

```xml
<arm>
  <head>
    <layout>
      <root-layout id="R0" backgroundColor="FFFF00" height="415" width="615"
        background="nameofpicture.jpg" backgroundAnim="nameofanimation.swf" />
      <region id="R1" height="250" width="400" top="155" left="5" backgroundColor="FFFF00" z-index="2" />
      <region id="R2" height="400" width="200" top="155" left="415" z-index="2" />
      <region id="R3" height="150" width="605" top="5" left="5" z-index="2" />
      <controlpanel height="30" width="30" top="170" left="170" />
    </layout>
  </head>
  <body><par>
    <seq region="R1">
      <image src="media1.jpg" dur="5s" />
      <image src="media2.jpg" begin="5s" dur="5s" />
      <image src="media1.jpg" dur="4s" />
    </seq>
    <seq region="R2">
      <image src="media4.jpg" dur="4s" />
      <image src="media5.jpg" dur="3s" />
      <image src="media6.jpg" dur="5s" />
    </seq>
    <seq region="R3">
      <image src="media7.jpg" dur="4s" />
      <image src="media8.jpg" dur="3s" />
      <image src="media9.jpg" dur="5s" />
    </seq>
  </par></body>
</arm>
```

### Fig. 4 – Head of an ARM file

The **head** describes the spatial layout, whereas other aspects (temporal, navigational) are described in the **body** part. Given a document $H = (R, F, A)$, the head part corresponds to the set of regions $R$, and the body part includes the set of flows $F$ and the associations in $A$. A set of ARM files composes an ARM presentation (Advanced Rich Media Channel). Each ARM file represents a document with its own definition of spatial regions and temporal description. ARM files, as for HTML, are stored in a reachable repository, or can be computed on demand over a database.

```xml
<body><par>
  <seq regions="R1">
    <image src="media1.jpg" dur="5s" />
    <image src="media2.jpg" begin="5s" dur="5s" />
    <image src="media1.jpg" dur="4s" />
  </seq>
  <seq regions="R2">
    <image src="media4.jpg" dur="4s" />
    <image src="media5.jpg" dur="3s" />
    <image src="media6.jpg" dur="5s" />
  </seq>
  <seq regions="R3">
    <image src="media7.jpg" dur="4s" />
    <image src="media8.jpg" dur="3s" />
    <image src="media9.jpg" dur="5s" />
  </seq>
</par></body>
```

### Fig. 5 – Body of an ARM file

Fig. 3 summarizes the different possibilities of each mode and shows where the changes of associations are specified. It should be noticed that:

- As SMIL links are limited to link one place in a presentation to an entire document, only the reset mode is available in this approach (see Fig. 3 (a) and (b)).
- XLink and HyTime provide the two modes reset (see Fig. 3 (a) and (b)) and continuous (see Fig. 3 (c)). Notice however that the continuous mode implies only one file.
- ARM provides the two modes in any case, although the continuous mode as in XLink is currently only partially implemented.
3.1 Spatial Definitions
An ARMC document associated to ARM files is divided into regions in which media are played. The spatial description starts with the `<layout>` markup (see Fig. 4.a). The first part, called `<root-layout>`, defines the spatial description of the main area, which is the one taken by the player. Regions are declared with a `<region>` markup. Notice that regions in ARM are not defined as a whole as in SMIL, but as objects that can be considered independently.

3.1.1 Root Layout
The `<root-layout>` part defines the size of the presentation, the background color and the default setting (background picture or animation). For example, the ARM code given in Fig. 4.a defines a player named R0 whose size is 415 pixels height and 615 pixels width, the background of the region being colored with an hexadecimal color and a Flash animation being played in the background.

3.1.2 Region
The presentation is divided into regions, and the markup, `<region>` describes the spatial and graphical aspects of the regions. The attributes defining the tuple d of a region are the following: (i) id, to identify each region as a unique object, (ii) backgroundColor, (iii) background, the background picture, (iv) backgroundAnim, the Flash animation which is played on the background, (v) title, (vi) top, left, width, height, (vii) zIndex, which allows regions to be defined as layers, (viii) type, regions can have plural behaviors, which is not described in this paper. For example, the code given Fig. 5 defines three regions named R1, R2, and R3.

3.1.3 Controlpanel
`<controlpanel>` is defined as a region whose role is to control the timeline of the whole presentation in order to synchronize all regions of the presentation. Notice that, in SMIL players, the control panel is defined by the player itself, but that, in the ARM player, the user can decide if the controlpanel region should appear and where.

3.2 Temporal Definitions
3.2.1 Body
The `<body>` part of an ARM file, seen as a document H = (A, F, R), contains the associations in A together with the definitions of the flows in F.

As explained previously, a flow is a set of media that are organized based on Allen's algebra [7]. The syntax we use is the same as SMIL: `dur, begin, end, dur` indicates the duration of the media, `start` and `end` respectively indicate the relative start time and end of the media. More precisely, the (optional) attribute `end` is used if the diffusion of the media is to be stopped before its actual end.

For example, in Fig. 5.a, we consider a flow associated to region R1 and that consists of three media: media1.jpg, media2.jpg, media3.jpg. Moreover, when the presentation starts, media1.jpg is displayed for 5 seconds, then a delay of 5 seconds is expected to start media2.jpg, which is displayed for 9 seconds, and then, media3.jpg is displayed for 4 seconds.

3.2.2 Activating Regions
Regions are defined in the body part, independently from the fact that they play something (active) or not (inactive). We recall that active regions are those that can be seen on the player screen, whereas inactive regions do not appear in the presentation. To be active, a region R must appear in a pair (R, F) of A, which is implemented through a `seq` markup in the document body. If the `seq` markup references no media, then R is active in the player and its default settings are displayed.

Notice that, only the `root-layout` region is always considered as active.

Since ARM presentations have pre-orchestrated scenarios, it is possible to know exactly what are the regions involved in the presentation, and what are the regions that appear in one or more document. Thus, when running a presentation, it is possible to activate or deactivate a region, which is not possible with other existing tools in which regions, once defined, are always considered as active.

Moreover, in order to save resources, the first document of the presentation contains the definition of all regions involved in the presentation, even if they are not active. In this way, during the execution of the presentation, when switching from one document to another, only overloaded regions have to be parsed.

We give more details on this important feature of our implementation in the next section.

3.3 Navigation
To switch from a document to the other, we use hyperlinks between documents. These hyperlinks can be either hyperlinks...
over time, or spatial hyperlinks, or a combination of both. A hyperlink over time allows going backward or forward in the presentation, and a spatial hyperlink allows changing the timeline associated to a single region.

3.3.1 The a markup
As shown in Fig 7, the a markup is defined as in HTML. The attribute href locates the resource to load or to follow through the hyperlink, and the attribute target is detailed in next subsection.

The hyperlink is directly defined in the ARM document, the link is defined over media composing the flow (text, video, ...). In the example, the link is defined over a video, meaning that when the video is displayed the hyperlink can be activated. It should be noticed that the resource can be:
- A single media, in which case only one association in the set A of associations in the current ARM file is changed (see Fig 3 (c))
- An ARM file in which the changes of associations are defined (see Fig 3 (d))

Moreover, the principle of anchors is kept to allow indirect addressing and is extended to perform links over time.

Currently, the # symbol is used in SMIL to access the timeline at the beginning of a media; in ARM the # symbol is not limited to synchronize the presentation with the beginning of a given media, but also with any instant in the timeline of the player.

3.3.2 Target
The possible values for target are _arm, _top, _blank, _self or the name of a region. The value _arm opens the new resources in the root layout of the player, and enables to change of document. The value _blank allows opening the current resource in a new player. The value _self is used when the hyperlink loads the resource into the region where the hyperlink has been activated. If the name of a region is specified, then the resource is played in this region. The default value is _top, in which case the resource is played on the web page that contains the player. The use of target allows defining associations as in XLink, and thus is limited to associations between regions and flows where flows are a unique media. However, the change of associations for flows that are composed of numerous media are assumed by the principle of inheritance, as explained in the next section

4. INHERITANCE
We have already said that, when switching from one document to another, the continuity of the streamed media should be possible, and that regions could be reshaped if necessary. In this section, we present our implementation of inheritance and overloading that address these issues.

4.1 Spatial Inheritance and Overloading
In [7], the authors notice that the same temporal behavior can be defined in different ways for different documents. With the same idea in mind, we use inheritance to ease the treatment of documents having a similar behaviour, because as already explained, parsing is a heavy task for a player not only dedicated to this work. Since ARM presentations are pre-orchestrated scenarios, using spatial inheritance, each needed region is already known when loading the first document of the presentation.

4.1.1 Spatial Inheritance
Since regions are assumed to be defined in the first ARM file, even if not all of them are active, they have already been integrated into the player.

In order to parse only once the definitions of regions that do not change during the presentation, we add to the <layout> markup an attribute, called herit, with no as default value, and yes as another possible value. If herit value is yes, the current document inherits from all predefined regions in the most recently displayed document.

Using the formalism given in Section 2, let us consider H1 and H2 defined as follows:
- H1 = (R1, F1, A1) with
  - R1 = {<R1, d1>, <R2, d1>, <R3, d1>},
  - F1 = {F1, F2, F3} and
  - A1 = { (R1, F1), (R2, F2), (R3, F3) }
- H2 = (s-herit, F2, A2) with
  - F2 = {F1, F2, F3}, and
  - A2 = { (R1, F1), (R2, F2), (R3, F3) }

s-herit in R1 indicates that the regions in H2 are the same as those in R1, that is R2 = R1. Thus, regions are not re-defined in the new ARM file, and the player preserves the regions from the
4.1.2 Overloading

Moreover, the object-oriented structure provides the possibility of overloading methods, making it possible to overload regions, and thus to change the attribute values of a region.

Fig. 9 shows the ARM definition of H₂ that inherits from the spatial structure of H₁. Note that in this example, R₁ is redefined because its attributes have changed: the region is now 400 height and 400 width, whereas it was 250 height and 400 width previously in H₁.

<layout herit="yes">
  <region id="R1" left="5" top="155" height="400" width="400" z-index="2" backgroundColor="FFFF00" />
</layout>

Fig. 9– Overloading regions

When defining in a targeted document a spatial inheritance coupled with overloading, if the id of the region already exists, and if the targeted ARM document is loaded, then the player only reshapes this region. This helps to solve the issue of the dynamic fit of a region (as in the example of switching from H₁ to H₃ of Fig. 1 and 2), while avoiding parsing the other unchanged regions.

We note that this principle can also be used to add new regions. In Fig. 10, a new document is defined as containing a new region named RegionDoe, which was not present in the file H₁. Using this possibility in a future version of the ARM architecture, we can consider on demand presentations, supported by inheritance.

<layout herit="yes">
  <region id="RegionDoe" left="527" top="138" height="79" width="313" z-index="2" backgroundColor="#FECF2A" />
</layout>

Fig. 10 – Defining a new region

4.2 Temporal Inheritance

Temporal inheritance is used to preserve the association between a region and its flow when navigating, thus allowing the region to keep unchanged its timeline, as illustrated in Fig. 11.

In this example, in the file H₁ (see Fig. 11.a), the flow of R₁ is defined as in Fig. 7. Thus, when the video is displayed, a hyperlink can be activated to reach the ARM file displayed in Fig. 11.b, if in this second document, we have: <seq region="R1" herit="yes" />. In this case, if a hyperlink is activated at the 15th second, the resulting timeline is given in Fig.11.c and the region R₁ continues to play the video for 42 - 15 = 27 s.

More generally, when the seq mark up for a region has value yes for its attribute herit, a temporal inheritance is specified and the flow played in the associated region remains the same. If the value of herit is no, each media of the flow is reloaded. In this latter case, our approach works as SMIL: the buffer is emptied and the process is run from scratch. On the other hand, with temporal inheritance, we are able to handle multidirectional linking and to use the cache for components that have already been loaded in the buffer, while preserving the continuity of the display.

<layout herit="yes">
  <region id="RegionDoe" left="527" top="138" height="79" width="313" z-index="2" backgroundColor="#FECF2A" />
</layout>

Fig. 11 – Temporal inheritance

It is important to note that temporal inheritance and spatial inheritance can be used independently from each other, depending on the purpose to achieve.

5. IMPLEMENTATION

Based on the fact that Macromedia Flash content reaches 98.3% of Internet viewers [5], the architecture of our implemented system is composed of a Flash player, a Flash Communication server to stream media, and a document server that delivers ARM documents.

5.1 General Architecture

The two components concerned by the synchronization are the Flash player and the Flash Communication Server (FCS). Fig. 12 shows the FCS where media are located. In order to allow the streaming of audio, video, and other continuous media, the FCS delivers media to the Flash player by means of a persistent RTMP connection.

A client-side Flash player has been dedicated to handle user’s interactions. The Flash player sends requests both to the FCS
and to the ARM documents server. The ARM documents server sends ARM files that describe the presentation, the Flash player interprets these files and requests to the FCS the media that are needed by the ARM file.

The ARM documents server has two functionalities: (a) manage the ARM documents, and (b) respond to the player by sending ARM files as a web server would do.

Referring to Fig. 12, the manager component is dedicated to authoring of multimedia documents, and stores it in a database called BDD. The server component queries the database and delivers ARM files to the Flash player according to the scenario created by the author with the manager component.

The player is the central component of the architecture; its resources should be carefully used to provide the best streaming performance.

5.2 Player architecture
As shown in Fig. 13, the player architecture includes four modules: the parser, the sequencer, the display manager and the event manager. ARM files are sent to the parser that interprets the spatial structure of the presentation and then its temporal and navigational composition.

The sequencer allows media to be played at the right time and at the right place by the display manager. The display manager detects user’s interactions with the interface. Events are sent to the event manager, that either asks the ARM documents server another ARM file, or gives orders to the sequencer to resynchronize.

6. CONCLUSION AND FUTURE WORKS
We have introduced the ARM format, a format for describing hypermedia documents on the WWW and the navigation between these documents by means of inheritance. ARM has a SMIL based description with spatial and temporal structures. Whereas other works describe active regions only, ARM enables to consider active or inactive regions, the activation of a region depending on the user’s interaction.

Moreover, despite the evident possibilities of Hytime and XLink, these approaches do not optimize the usability of the components. In our context, using inheritance, when switching from one file to another, the player has not to parse already considered information, which improves the performance of sequencing media. This allows the player to focus on the streaming process and the synchronization of media. Our approach is dedicated to presentations which do not involve rapid changes in their spatial and temporal relations such as Web TV, e-learning courses.

We have implemented our approach by an XML based format for multimedia presentations and a third-party architecture: a streaming media server, a Flash player, and a hypermedia document server. A beta version of the Flash player that allows inheritance is now available.

Additionally, we are currently implementing the ARM documents server and our future works include the following: (i) enhance the region behavior to allow for more user interaction capabilities (ii) implement an authoring tool for ARM documents to be delivered offline (iii) search for a mechanism for contextual referencing of ARM presentation.

REFERENCES