User-Centered Abstractions for Adaptive Hypermedia Presentations

Dick C.A. Bulterman
CWI: Centrum voor Wiskunde en Informatica
Kruislaan 413
1098 SJ Amsterdam, The Netherlands
+31-20-592 41 47
Dick.Bulterman@cwi.nl

1. ABSTRACT

This paper describes document modelling constructs that support alternate content choices for generalized hypermedia presentations. While there has been much work done on adaptive hypermedia documents in the context of low-level quality-of-service adaptation, little attention has been paid to support of user-level adaptation of multimedia content. Taking examples from the domains of information accessibility for the visual/hearing impaired, multilingual information presentation, and content adaptation in distance learning, we show how simple interfaces to rich hypermedia documents can give decided benefits to the user community.
We discuss our work in terms of experiments from the CWI CMIF project and indicate how these solutions have been integrated with the W3C SMIL language in the GRiNS editor and player for Web use.

1.1 Keywords

User-centered adaptability, accessibility, hypermedia interfaces, CMIF, GRiNS, SMIL.

2. INTRODUCTION

One of the major benefits of on-line electronic publications and presentations — perhaps the only real benefit — is that the content of these documents and presentations are not by nature static. The dynamic nature of on-line information often is associated with *timeliness*: information can be requested on-demand and can be updated at whatever rate the supplier feels is necessary or feasible. The dynamic nature also manifests itself in *selectivity*: consumers can choose the subset of a total information space that meets their needs and suppliers can decide which classes of information they want to provide to a particular client.

Suppliers and consumers of Web-based information have made extensive use of both the timeliness and selectivity available inherent to the Web. In spite of these successes, the general level of adaptability within individual Web presentations remains minimal. Here, the term ‘adaptability’ is used in its general sense: changing the ultimate presentation to satisfy a set of run-time constraints. The types of run-time constraints that steer adaptive presentations can vary widely.

For example, the introduction of the URI in HTTP 1.1 [5] provides for greater selectability within the context of a single object access, but it does not provide this support with a general presentation-based framework for altering related objects in a common manner. For instance, the selection and rendering of two URI’s within the same page are treated as independent events, without any common control available at the integrating level. For this reason, the use of URIs can probably be best classified as providing *target adaptation*: the information associated with the target of a single reference can be altered, with the adaptation being controlled by the target. Such an approach is especially useful in situations with minimal intra-reference constraints (as within an HTML [15] page), but it presents problems when temporal or semantic relationships exist within the larger presentation. The single-focus nature of the URI does not allow coordination among groups of URIs within a page unless the semantic structuring of the enclosing container anticipated this at author time.

Research in low-level quality of service (QoS) algorithms concentrates on adaptability at a different level of abstraction and for different reasons than its higher-level counter-part [13],[14],[18]. At the risk of over-generalizing, we can say that the goal of most QoS research is to change individual objects within a presentation so that that presentation, in whole or part, conforms to the resources available across the server--client transmission pipeline. This can be classified as *encoding adaptation*: adaptation consists of some form of selection among semantically equivalent encodings (such as a dense version of an image versus a sparse encoding) or dynamically changing low-level buffering policies to achieve some minimal level of presentation quality based on the characteristics of the data encoding.

Although the use of target and encoding adaptation address different levels of abstraction, they both share the property that they manipulate individual objects outside the context of the total presentation. Neither addresses a more basic user-driven need to influence the way that related groups of information are selected and controlled during a presentation. This type of adaptation could be classified as *aggregation adaptation*: the control of semantically related high-level flows of information within a presentation, so that the needs/desires of different classes of users are met from within a single presentation focus. Unfortunately, 'aggregation adaptation' is such an ugly term that we would not want to be responsible for introducing it; as a result, we use *user-centered adaptation* instead.

This paper describes abstractions that can be used to define presentations in such a way that user-centered adaptation is supported. The main abstraction is that of the *channel*: a grouping mechanism that can be used to control the selection of related sets of media objects that are targeted for specific user needs *within* a larger presentation. The benefit of using the channel is that it adds to the richness of a presentation without dramatically increasing presentation authoring or maintenance costs. Put another way, the channel abstraction removes a barrier to supporting tailored content for specific user groups — not by providing some sort of magical creation of multiple streams of data, but by providing the author with the ability of (incrementally) extending the content of a presentation and the user with the option to select those content projections that best suit their needs.

This paper is structured in four main sections. We begin our discussion with several examples of where user-centered adaptation can be useful. We then describe the channel approach as used in the CWI CMIF model and environment [1]. This is followed by a discussion of how the channel concept is itself being adapted for use with W3C's SMIL language. We close with a summary of lessons learned from our multi-year experiments with the user-centered concept.

3. EXAMPLES OF USER-CENTERED ADAPTATION

In order to provide a concrete basis for discussion, we consider three applications of user-centered adaptivity. These are: support for multiple projections of information for the hearing and visually impaired (most recently, this falls under the heading *accessibility*); support for multiple languages within one presentation; and support for user-centered decomposition or recomposition of a document based on variances in available transport or end-user resources. These examples are not intended to be exhaustive; rather, they are used to provide an overview of the diverse circumstances when user-centered control over information flows is desirable.

3.1 Accessibility for the hearing/visually impaired

Slowly but surely, it is becoming clear that hypermedia content created for the electronic information infrastructure can not assume that all users are "equally abled". From a market-driven perspective, this had typically meant that some users had fast machines/connections and some slow, but from a societal perspective — with more than a gentle amount of prodding from the US Federal Government and consumer groups [4] — it is now also becoming apparent that the flexibility that is inherent in digital communication can and must be used to ensure that at least the blind and the deaf are not totally disenfranchised within the information society.

The needs of the deaf and the blind are, of course, often complimentary, but they do illustrate why user-centered control of a presentation is important. Nearly everyone is familiar with captioned television programs, which are the primary means that broadcast TV is made available to the deaf [11]. It is important to realize, however, that such captioning does not only serve the deaf. There are many instances when, out of consideration to office-mates or in the noisy context of the factory floor, a substitute encoding for an audio stream could be useful. (Think of watching the news in a noisy airport departure lounge.)

While captions for the deaf are relatively common for TV, they are rare for media such as audio. (The support for captioned radio, for example, is minimal.) This is also true for audio annotations of visual material for the blind. While most current-generation personal computer architectures have been tuned to decode MPEG-1 (and sometimes MPEG-2) videos without special hardware, the technically simpler process of embedding text-to-speech encoding has not yet been packaged as a system 'must'. Even so, the problem and the user-level abstractions required to support the presentation of semantically equivalent content by syntactically different encoding is the same.

There is a temptation to provide support for special needs in special versions of content or with special tools developed for specific target groups. Unfortunately, such 'special' treatment brings with it large scale extra costs, effectively creating a barrier for what seems like a small market segment. If, on the other hand, facilities were provided to manage different projections of content within a single document source, all users could benefit from the introduction of such facilities in the mainstream market. At runtime, the user could select the encoding or set of encoding that best suit their needs. In so doing, it is important to realize that the choice is often not based on media type, but on collections of more abstract information. Not *all* audio needs to be translated to text — certainly not the background music, which can simply be turned off — and not even all of the spoken text will always be relevant.

3.2 Multi-lingual presentations

In most ways, the problem of constructing multi-lingual applications is functionally equivalent to the accessibility problem. Simply put, the problem for users of a presentation containing a French audio component may not be that they can't hear, but that they can't 'hear' (or understand) French. In all other respects, the situations are identical.

While the problem is the same, the solution for multi-lingual presentations is often different. A deaf user is not served by substituting one audio track by another audio track, but for the multi-lingual case this is often a useful strategy. (This is the dubbing approach.) Alternatively, it may be more appropriate to switch from audio to text, and provide the alternative encoding along with the original. (This is the subtitling approach.) A key point is that while the selection of substitution method (dubbing or subtitling) should be made by the user, it is normally always made by the content provider. This limits the ultimate reusability of content and increases the cost for supporting special needs.

3.3 Adapting content for distance learning

Where adaptation based on accessibility and multi-lingual support are primarily concerned with high-level selection of information components, user-centered adaptation at the encoding level is also important.

Consider a video lecture being distributed nationally via a Web-like network. At some points in the lecture, the user may be content with a couple of still images per minute and a reasonable audio connection (or its equivalent from an accessibility point of view, in the language of the user's choice). When the material gets particularly detailed, more detail in the presentation may be required. Thus, the still images get replaced by high-resolution video. While this all seems like standard QoS management, the key aspect is that a change in service policy is not made in response to the bandwidth available on the transmission line or the size of the local buffers, but simply because the user wanted more or less detail in the presentation itself.

4. *CHANNELS*: CMIF's APPROACH TO USER-CENTERED ADAPTATION

Although user-centered adaptation is often seen in terms of a specific user group needs, the examples above are intended to motivate that adaptation can be seen as a single, abstract concern that takes many forms.

One of the original goals of CWI's CMIF project [1] was the definition of an information grouping abstraction that would be useful in specifying collections of related content during authoring, and then selection one or most sets of content at runtime by the user based on that user's needs. The name given to this abstraction was the *Logical Resource Channel*, or simply the *Channel*.[6]

4.1 Overview of the Channel functionality

The purpose of a channel is to be a grouping abstraction for a set of media items that share some common attributes. These may include physical attributes such as screen position or text color, or they may be logical attributes, such as natural language or presentation priority.

The channel provides a logical thread upon which media objects can be placed. This thread can be turned on or off during the presentation based on the needs of the user or the user's agent (that is, the user interface or the runtime support system). In this way, the Channel abstraction can be used to support user-centered adaptation, but it can also be used by the runtime system to select more traditional QoS adaptation of content alternatives.

CMIF channels have not only a strong logical association among media objects on that channel, they also share presentation rendering and scheduling associations as well. In CMIF, it is not appropriate to speak of *the* audio or video channel—as if there was only a single video output stream—but rather *an* audio or video or text or image channel. An application may have many different text, video, audio, image or control channels, each of which is tailored to a specific logical grouping. Ultimately, the objects on a channel may get directed to a renderer of a particular type, but such a renderer may be responsible for multiple concurrent high-level information streams.

Any media item that is activated needs to be placed somewhere on the screen or an a loudspeaker. When several objects are rendered to the same space, it may make sense to manage this space based on a set of common attributes. Similarly, the actual rendering of a media object needs to be handled by some piece of code or device that is also constrained by a set of common properties. Finally — and most importantly — it may be that a set of media objects can be grouped not only by space and type, but also based on their semantic properties. For example, in a new broadcast, a "Dutch Audio" channel could contain all of the audio in that document that is spoken in Dutch. Alternatively, a "Anchor-Audio-Dutch" channel could be defined that contains the Dutch version of all of audio associated with the anchor. If a "Anchor-Audio-English" channel existed, a user potentially select which language they wanted at runtime.

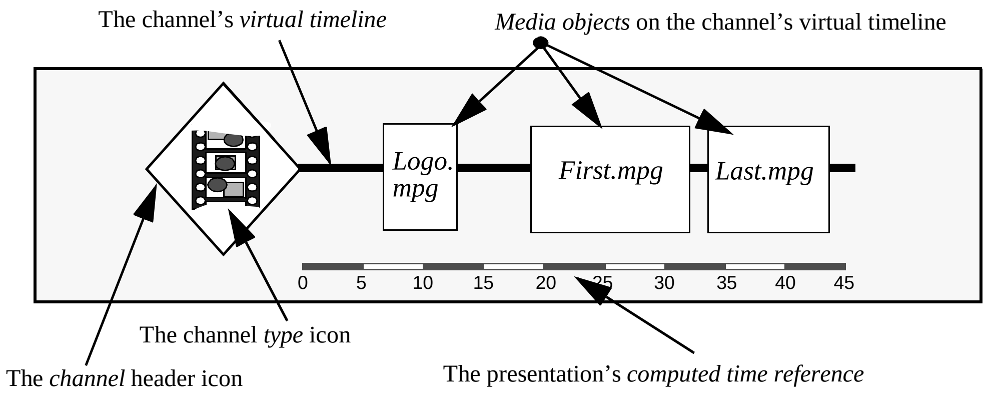

The channel's *virtual timeline*
Media objects on the channel's virtual timeline
Logo. mpg
First.mpg
Last.mpg
0 5 10 15 20 25 30 35 40 45
The channel *type* icon
The *channel* header icon
The presentation's *computed time reference*

Figure 1. The CMIF *logical resource channel* visual abstraction

Figure 1 shows a visual representation of a channel. The channel has a header that contains channel attributes. Each channel supports a particular type of output. This type can range from simple text, through complex composite multimedia data, to non-visible control operations. (A video channel icon is displayed.) Associated with a channel is its virtual timeline; media item activation instance descriptors are placed on this timeline. The timeline is virtual in that it does not show fixed offsets within the presentation: hyperlink and/or timing control (like a loop) in a particular instance will determine a presentation's 'real time'. Associated with the virtual timeline is the presentation's computed time reference line. This timeline is computed by the scheduler based on information available in the CMIF data structure.

The current run-time environment at CWI supports sixteen channel types:

- audio-only: *sound, Midi*
- video-only: *movie, MPEG*
- audio-video: *MPEG*
- text channels: *HTML, text, label*
- illustration: *graph, image*
- background: *layout*
- control: *cmif, Null, Python, Shell, Socket, VCR*

Other special-purpose channels also exist, such as the *Morse Code* channel; this channel takes text and produces Morse code audio output.

The list above emphasizes the physical properties of the media type, rather than the logical association among groups of generic media objects. This reflects our experience that, from an author's perspective, there are several levels of grouping that need to be managed within an application simultaneously. These are: layout grouping, renderer grouping and semantic grouping.

While the view shown in Fig. 1 is fairly typical for time-line based systems, the combination of multiple virtual timelines into a presentation timeline is less typical. This is shown in Figure 2. Here we see a presentation fragment in which one video channel, two audio channels and two text channels are shown. The runtime projection of this presentation will in all probability not have all channels active at once. Most users won't want multiple language channels active at the same time (neither for the audio or the text versions). Different mixes of active channels might be activated during a particular rendering of the presentation, depending on the requirements of the user. A Dutch-speaking blind person may not want video or text, but only the Dutch-language audio channel. An English speaking student of Dutch may want the Dutch audio and the English captions active (or vice versa), while a passenger on a crowded flight

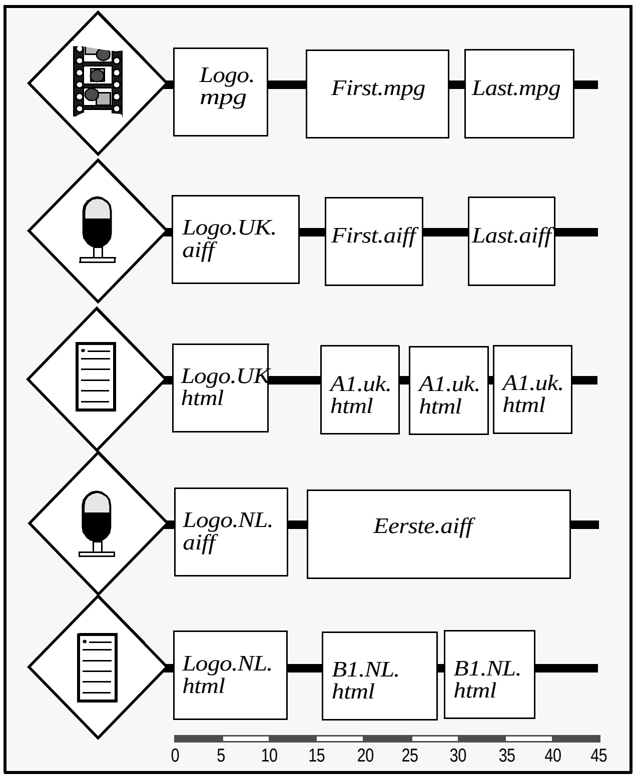


Figure 2. Presentation fragment containing multiple *logical resource channel*s.

may want to see only the English captions and the video. Note that the choices do not need to be made solely by the user. If the playback environment realizes that streaming audio support is not available (or if the user has not made a micro-payment for the video), then it could itself choose to deactivate certain channels or to, say, substitute still images for the videos (assuming that such alternatives were specified by the author).

Figure 3 shows how the channel concept is used to support user-centered adaptation. In Figure 3(a), we see a fragment from a network-delivered newscast. Figure 3(b) shows the same newscast with captions turned on for the main anchor (upper right) only. Figure 3(c) shows that line-by-line captions have been replaced by block captions in the upper left corner. The user can select which set of channels is to be active interactively during the presentation, or the selection can be based on that user's preferences profile.

An example of how the Web News application is managed using channels in the GRiNS authoring environment [3] is given in Figure 4. Here we see several parallel set of objects that are placed on a collection of channels. Each of the channels identifies a separate logical timeline, influencing its own presentation sub-scheduler. If events on one channel have an influence on events in another channel, they are indicated by CMIF's sync_arcs [2]. Since a channel may be active or inactive based on user preferences, the actual resolution of synchronization relationships occur at runtime rather than as a part of a presentation's static analysis.

From a CMIF perspective, the channel view is only one of the views that an author has on the relationships in a document. The main view is something that we call the hierarchy view; this is where the main temporal relationships are defined. The channels are simply management abstractions that can be widely applied to a variety of resource control needs.

4.2 Contrasting *channels* with other approaches

There are several ways that alternative content can be added to a presentation. Perhaps the most obvious form is to simply write a program that analyzes the runtime situation and 'does the right thing'. This is the approach taken by HTML-4.0, often called Dynamic HTML [16] The most problematic aspect of this is that most document authors are not programmers. Even if they were, the user-centered nature of the adaptive process would make the task of integrating system- and user-needs with a single code fragment sufficiently complex that such an approach would serve as a disincentive to creating generalized alternative presentations.

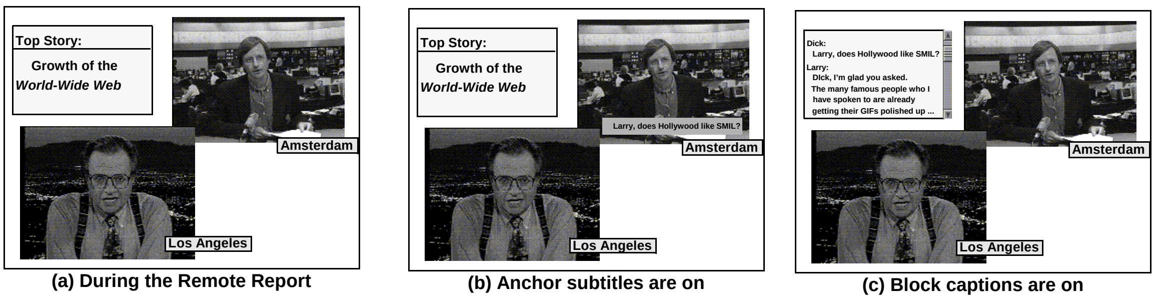
Another approach is to dynamically create the entire presentation at runtime using indirect access to all possible objects stored in a database and then only rendering those objects that are required at a particular moment in the presentation [10]. While such an approach has interesting possibilities, it does not present a particularly manageable approach for a document author. It transforms the relatively simple coordination task of specifying alternatives within a presentation to the complex task of automatic authoring which is beyond the capabilities of most users or systems to implement.

Another approach is to define explicit alternatives within the contents for each particular media object. This is similar to the approach used within HTML for individual media objects:

```
<img src='anchor.gif' alt="Dick's face">
```

The problem with this approach is that a single piece of content may have many alternatives associated with it, of may different types. Generalized line-item substitution then requires complex substitution mechanisms that model activation scripts.

A slightly more flexible approach is Transparent Content Negotiation (TCN) [7]. In this approach, the document contains a single reference to an object. At time of access, a process of negotiation takes place that allows the 'right' object to be extracted from the server. While this represents an improvement over line-item substitution, it typically hides the entire substitution process. Users have little control, since users don't get to see and evaluate the alternatives. Also, since each object is developed


Top Story:
Growth of the
World-Wide Web
Amsterdam
Los Angeles
Top Story:
Growth of the
World-Wide Web
Larry, does Hollywood like SMIL?
Amsterdam
Los Angeles
Dick:
Larry, does Hollywood like SMIL?
Larry:
DIck, I'm glad you asked.
The many famous people who I have spoken to are already getting their GIFs polished up ...
Amsterdam
Los Angeles

(a) During the Remote Report **(b) Anchor subtitles are on** **(c) Block captions are on**

Figure 3. Three views of a newscast.
(a) shows the basic version of the broadcast; (b) shows the addition of captions on the anchor video
(c) shows the use of block captions as a running transcript

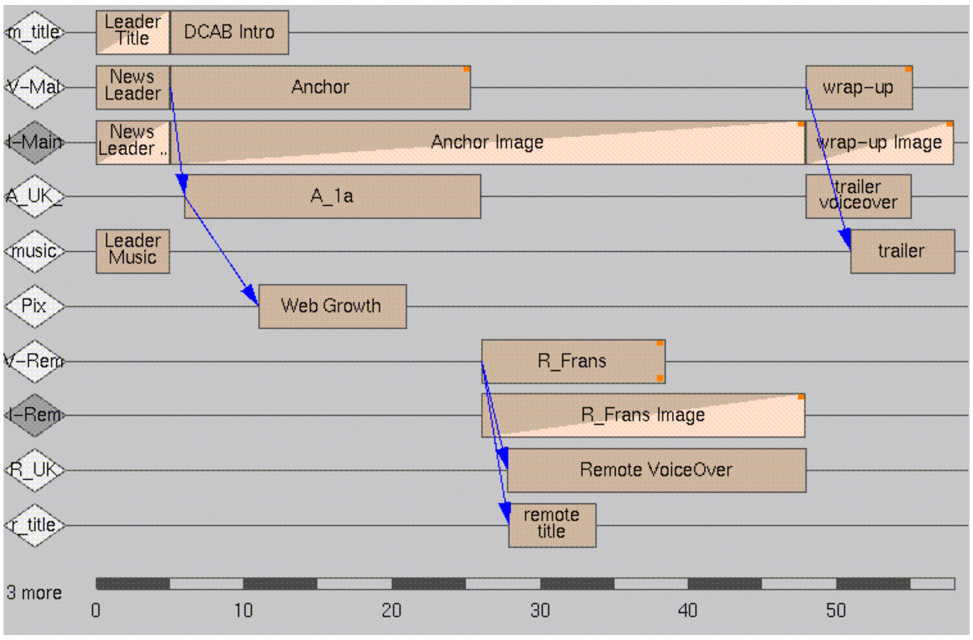


**Figure 4. The collection of a set of channel that make up an application fragment.
Each channel represents a virtual timeline, all of which are combined to form the presentation's composite time at runtime.
(Time flows from left to right.)**

independently, defining common user-level grouping abstractions across these independent objects is a difficult problem.

An intuitive approach is to define separate projections for each composite presentation, each of which could be configured for the alternatives available. Thus, one complete projection is generated for the Dutch version of the presentation and one for the English version. This is *not* the approach we take. Instead, we give each channel thread its own independent timeline. The entire presentation is made up of the sum of the active timelines at any particular moment.

In systems like Marcomedia's Director [8], a single application timeline can be annotated with a number of media objects. These objects are placed on the timeline with a graphic editor. In contrast, CMIF channels represent multiple timelines, each derived from the structure of an application [9]. (In our GRiNS authoring environment, the user never needs to place events on the channel; this is done automatically.) The advantage of this approach is that it allows any one of the channels to be selected or de-selected without violating any of the timing relationships in other channels.

It is interesting to compare this approach to resource management with that used in a typical HTML browser. Many browsers give the user the ability to turn images or sounds on or off. This is done to make presentation rendering a positive experience on slow communications lines. In the channel approach, a user could select a specific type of images that could be turned on or off — such as, show me all of the anchor videos, but don't show any commercials. This takes content substitution out of the context of the representation and into the context of a particular user's interests.

Finally, some systems attempt to solve a particular aspect of the user-centered adaptation problem only. For example, the language SAMI [9] developed within the STMPE community, specifies an approach to annotating individual content streams so that captions can be associated with associated pictures or video fragments. This approach is very useful within one set of options, but it does not provide a general approach to supporting user-centered abstraction and selection across multiple data types.

4.3 Extensions: high-level grouping of *channels*

At present, CMIF supports only a single level of grouping. If, for example, an application contained Dutch Text and Dutch Audio, these would be managed as separate channels. Each one could be manipulated by the user independently.

Selection of the active set of channels takes place at runtime. Figure 5 shows a typical dialogue box that provides the user with the channel selection mechanism. (Note that this process could be managed in a number of ways; we show the most obvious here.)

The individual selection of active channels is a powerful user control mechanism that allows considerable flexibility in dynamically shaping a presentation. Unfortunately, it can burden the casual user with too many choices. For this reason, we are also investigating the a higher degree of

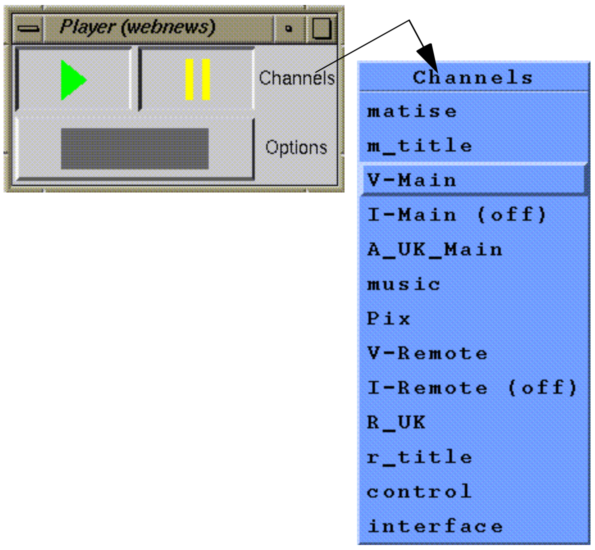


Figure 5. The channel selection interface.
Within the VCR controls, a selection facility is shown that allows individual channels to be turned on and off.

hierarchical grouping of channels. Individual channels can then be bundled together in a channel group and be manipulated as a composite complex channel. This limits the flexibility for any one channel, but makes structuring of related information in an application easier to define. We currently support this functionality via the user interface, but not through the a channel hierarchy. Support for this feature is currently being integrated into the GRiNS authoring interface.

5. ADAPTING ADAPTATION: CMIF's *CHANNELS* AND *SMIL*

A channel is a logical grouping mechanism with which a related set of objects can be managed by the CMIF player. The relationship within the channel can be based on various factors, including:

- common encoding properties,
- common semantic properties,
- common resource use properties, or
- common scheduling requirements

It could also be based on some other grouping that we have not yet envisioned.

The nature of a CMIF channel is such that it serves as a means of isolating a related collection media objects from their runtime projections. This was done to promote reuse of objects, but also reuse of structure within a document. (The hierarchical structure of the applications basically says 'what' and 'when' of an object's use, and the channel architecture says 'how', 'where' and maybe 'why'.) A disadvantage of this approach is that the channel concept overloads the notions of layout, rendering activation and logical grouping.

During the W3C's development of the SMIL language [17], issue of selectability of content in a presentation received a great deal of attention. Early on, it was decided that a `SWITCH` construct would form the basic selection primitive in the encoding. A `SWITCH` allows a series of alternatives to be specified for a particular piece of content, one of which is selected by the runtime environment for presentation. An example of how a `SWITCH` might be used to control the alternatives that could accompany a piece of video in a presentation would be:

```
 ...
 <par>
  <video src="anchor.mpg" ... />
  <switch>
   <audio src="dutch.aiff" ... />
   <audio src="english.aiff" ... />
   <text src="dutch.html" ... />
   <text src="english.html" ... />
  </switch>
 </par>
 ...
```

This fragment (which is pseudo-SMIL, for clarity) says that a video is played in parallel with one of: Dutch audio, English audio, Dutch text, or English text. SMIL does not specify the selection mechanism, only a way of specifying the alternatives.

There are two problems with this approach. First, it restricts the resolution of a `SWITCH` to a single alternative. (If you want Dutch audio *and* Dutch text, you need to specify a compound `SWITCH` statement, but in so doing, you always get the compound result.) More restrictively, it requires the author to explicitly state all of the possible combinations of input streams at author time. If the user wanted Dutch audio and English text, this possibility must have been considered at authoring time.

An example of a CMIF Channel-based solution to the same problem is given in the following document fragment:

```
 ...
<channel>
 ...<!-- channel definitions -->
</channel>
 ...
<par>
 <video src="anchor.mpg" channel="anchorV".../>
 <audio src="dutch.aiff" channel="anchorDA".../>
 <audio src="english.aiff" channel="anchorEA".../>
 <text src="dutch.html" channel="anchorDT".../>
 <text src="english.html" channel="anchorET".../>
</par>
 ...
```

This example says: a video is accompanied by four other data objects, all of which are (logically) shown in parallel. This is, of course, exactly what happens: all five do run in parallel, but it could be that only the video and one audio stream are actually selected by the user (or a user agent) to be rendered during the presentation. The point is: at author time you know which logical streams are available, but it is only at runtime that you know which combination of all

```
<?xml:namespace ns="http://www.cwi.nl/GRiNS-group" prefix="GRiNS" ?>
 <smil>
  <head>
   <layout>
    <!-- define projection regions -->
   </layout>
   <GRiNS:user_attributes>
    <GRiNS:u_group id="nl_aud" u_state="RENDERED" title="Dutch Audio Cap" override="allowed" />
    <GRiNS:u_group id="uk_aud" u_state="NOT_RENDERED" title="English Audio Cap" override="allowed" />
    <GRiNS:u_group id="nl_txt" u_state="NOT_RENDERED" title="Dutch Text Cap"override="allowed" />
    <GRiNS:u_group id="uk_txt" u_state="NOT_RENDERED" title="English Text Cap" override="allowed" />
   </GRiNS:user_attributes>
  </head>
  <body>
   ...
    <par>
     <video src="announcer.rm" region=”a”/>
     <text src="news_headline.html" region="b"/>
     <audio src="story_1_nl.rm" GRiNS:u_group="nl_aud" region="c"/>
     <audio src="story_1_uk.rm" GRiNS:u_group="uk_aud-cam" region="d"/>
     <text src="story_1_nl.html" GRiNS:u_group="nl_txt"/>
     <text src="story_1_uk.html" GRiNS:u_group="uk_txt"/>
    </par>
   ...
 </body>
</smil>
```

Figure 6. A short example using GRiNS's user-group attribute to define user-centered presentation alternatives in SMIL.

potentially available stream actually meet the user's needs.

It is true that, logically, the alternatives indicated by the channel construct could be represented as a set of `SWITCH` statements, although the resulting `SWITCH` should become explosive in size. Use of a Channel-like mechanism would significantly simplify the specification of user-centered alternative. The author could specify all of the individual components of a presentation at author time and then organize them in terms of their logical channel threads. The user (or user's agent) can then select which sets are active at runtime. An 'initial state' attribute can be used to define the default behavior.

The SMIL V1.0 recommendation currently supports both the notions of the `SWITCH` and a partial mechanism for controlling adaptive behavior called the *system test attribute*. The `SWITCH` provides a conventional branching structure that allows alternatives to be defined at authoring time. The system test attributes consist of a set of pre-defined (primarily system-related) attributes that describe dynamic aspects of the environment which can then be tested at run-time. For example:

```
<text src="cap.html" system-captions="true" .../>
```

will cause the object 'cap.html' to be rendered if *system-captions* evaluates to true.

The system test attribute mechanism is a significant extension over the SWITCH because of the way that it decouples the authoring and playback associations among a set of alternatives. Even so, it only partially meets the needs for user-centered control because of the static nature of the attributes themselves (they are defined as part of the language, and can't be extended easily by users).

An alternative approach, based on the CMIF Channel model, has been integrated into the GRiNS authoring and playback interface for SMIL [3]. In this extension to SMIL, a new grouping mechanism for attributes is used that allows a document author and a document viewer to define and select combinations of content that meet the needs of the application user. This is illustrated in Figure 6.

The basic constructs used by the GRiNS environment are:

- `user_attributes`: a section within the SMIL `head` that contains definitions of each of the user groups.
- `u_group`: an author-defined grouping of related media objects. These are defined within the section user_attributes that make up part of the document header, and they are referenced within a media object definition.
- `u_state`: this is the evaluated state of the `u_group`. If it evaluates to true, the associated object is rendered. If more than one `u_group` is associated with an object, all must evaluate to `RENDERED` for the object to be rendered. The initial behavior for the `u_group` is given in the definition (if nothing is specified, it defaults to `RENDERED`). The run-time state is defined by the user or the user agent. If a particular playback environment does not (or cannot) support user selection, the `u_state` attribute controls the author-specified default presentation.
- `override`: the author is given the ability to block overrides to the initial state by explicitly prohibiting this in

the `u_group` definition. It is up to the runtime environment to enforce this attribute. The attribute can also be used to influence adaptive behavior at lower level in the transport hierarchy.

The example in Figure 6 shows how user groups can be applied to the News example developed in this paper. Note that the example uses XML Namespace notation: the attributes are defined as specific to the GRiNS environment, allowing the document to also be played on other players without user-grouping support as well. In this latter case, the GRiNS tagged attributes and definitions are ignored. Note also that, in our environment, a graphical editor actually builds the SMIL code and inserts the required XML code, relieving the user of this burden.

Lines 6 through 11 define the available groups. Each group contains an identifier and a title (which can be used by the user interface agent to label the group), as well as the (optional) initial state definition and override flag. In line 7, a `u_group` named `nl_aud` is defined for Dutch audio captions that is initially set to `RENDERED`. The other groups in this (very simple) example are set to `NOT_RENDERED`.

In lines 15 through 22, a SMIL `PAR` contruct is used to identify a portion of a presentation. In this `PAR`, a single video (line 16) is accompanied by two audio streams (18,19) and two text streams (20,21), one each for English and Dutch. The `PAR` also contains a text title that contains a headline. The interaction of the user interface and the initial state determine which objects are rendered. Note that the same attributes are used across the entire document, meaning that the user only needs to select his/her content preferences once to control related groups of information. In the example, user is free to have the video and headline text accompanied by any combination of English and Dutch captions. (Note that if two audio captions are selected, the player will need to determine how these are processed for delivery.) If another approach had been used (such as content negotiation [7]) then the structure of the content portions would have to be very complex to provide the same facilities to the user.

6. LESSONS LEARNED

The need to adapt the contents of a 30 frame-per-second NTSC or PAL sized video when presenting it over a 28.8Kbps modem is fairly clear. (That is, it is clear if we assume that our primary interest is in preserving the inter-frame latency rather than the contents of any one image.) Here, the “system” — perhaps guided by user-level preferences or author defined mandates — can make an informed decision on how to process a stream of presentation content.

We have found that the need for supporting user-centered adaptation is much less obvious. When such support *is* provided, it is often rooted in the technological constraints of individual information carriers. The fact that it is difficult to make a video tape (or a CD -ROM) that contains four parallel peer-level sound tracks, let alone videos/CD’s that allow some media objects — such as an image — in one variant of the presentation but not in others has unduly influenced the design of non-linear systems. Video tape is constrained by the physical width of the tape, the single-focus playback head and a primitive set of user-interface controls. CD-ROMs are constrained by the relatively high penalty for random access, which make linear (video-tape-like) presentation encoding much more efficient that a more scattered model of information organization. The Web knows no such constraints (other than the single connection that is often used to carry information into and out of a workstation).

It is unfortunate that most multimedia presentations mimic the VCR for their control model, even when it is accompanied by navigation-based links. The use of the Channel concept provides a richer alternative to the user and it provides for a more managed authoring process for those applications that consider user-centered adaptation to be commercially attractive (or socially responsible).

The use of user attributes to support user-centered adaptation is not meant to be the exclusive means of partitioning application control information. For certain aspects of adaptation, the `SWITCH` presents a convenient framework; this is often the case when there are a limited number of choice points in a document. In other instances, the system attributes approach of SMIL is useful, although it is limited by the fact that authors cannot add their own attribute types. The use of user attributes is most appropriate when the user needs to participate — directly or indirectly — in the decision making process.

In adapting the CMIF Channel to the user group, the non-grouping aspects of the Channel have been eliminated. This will have the positive effect of isolating the logical grouping concerns of the Channel, but it will happen at a cost: the association of type information (audio, video, text, etc.) is often a vital component of selecting information when concerned with supporting the needs of the accessibility. If, in fact, it turns out that there is a more than casual relationship between semantic associations and information encoding type — which has been our CMIF experience to date — then the decoupling of type from the Channel may need to be recoupled.

7. ACKNOWLEDGEMENTS

This work has been carried out as part of the CWI CMIF hypermedia authoring and presentation project. This work has been funded most recently by the ESPRIT-IV project CHAMELEON of the European Union.

Users interested in evaluating the GRiNS environment should consult the following URL for details:

`http://www.cwi.nl/GRiNS`

8. REFERENCES

[1] D.C.A. Bulterman, R. van Liere and G. van Rossum. A Structure for Transportable, Dynamic Multimedia Documents (with R. van Liere and G. van Rossum). *Proceedings of 1991 Usenix Spring Conference on Multimedia Systems,* Nashville, TN, 1991 pp. 137-155.

See also http://www.cwi.nl/~dcab/.

[2] D.C.A. Bulterman. Synchronization of Multi-Sourced Multimedia Data for Heterogeneous Target Systems, in Network and Operating Systems Support for Digital Audio and Video (P. Venkat Rangan, Ed.), LNCS-712, Springer-Verlag, 1993, pp. 119-129.

[3] D.C.A. Bulterman, L. Hardman, J. Jansen, K. S. Mullender and L. Rutledge. GRiNS: A GRaphical INterface for creating and playing SMIL documents, *Proc. WWW-7*, Brisbane, Australia April 15-18. 1998. Computer Networks and ISDN Systems 30 (1998), 519-529

[4] Federal Communications Commission (USA). Notice of inquiry on access to telecommunications services and equipment for the disabled on the internet - Public Notice. [WTB] Released: 11/05/1996.

[5] R. Fielding, J. Gettys, J. Mogul, H. Frystyk and T. Berners-Lee. Hypertext Transfer Protocol — HTTP/1.1. IETF RFC 2068, January 1997.
URL: ftp://ftp.isi.edu/in-notes/rfc2068.txt

[6] L. Hardman, D.C.A. Bulterman and G. van Rossum. The Amsterdam Hypermedia Model: Adding Time and Context to the Dexter Model CACM 37(2), Feb. 1994, pp 50-62.

[7] K. Hotlman and A. Mutz. Transparent Content Negotiation in HTTP, IETF RFC 2295.
URL: ftp://ftp.isi.edu/in-notes/rfc2295.txt

[8] Macromedia, Inc.: Director 6.5.
URL: http://www.macromedia.com/software/director/

[9] Microsoft SAMI Form.
URL: http://www.microsoft.com/enable

[10] MONET: Extending databases for multimedia.
URL: http://www.cwi.nl/~monet/modprg.html

[11] NCAM: CPB/WGBH National Center for Accessible Media. URL: http://www.boston.com/wgbh/pages/ncam/

[12] C. Roisin, M. Jourdan, N. Layaïda and L. Sabry-Ismail: Authoring Environment for Interactive Multimedia Documents, Proc. MMM'97, Singapore. See also: http://opera.inrialpes.fr/OPERA/multimedia-eng.html

[13] H. Schulzrinne. RTP: Real-time Transport Protocol.
URL: http://www.cs.columbia.edu/~hgs/rtp/.

[14] H. Schulzrinne. A real-time stream control protocol (RTSP). URL: http://www.cs.columbia.edu/~hgs/rtsp/draft/draft-ietf-mmusic-stream-00.txt (11/26/96).

[15] W3C. HTML Version 3.2.
URL: http://www.w3.org/MarkUp/.

[16] W3C: HTML 4.0: W3C's Next Version of HTML.
URL: http://www.w3.org/MarkUp/Cougar/

[17] W3C SMIL 1.0 Recommendation.
URL: http://www.w3.org/TR/REC-smil/.

[18] L. Zhang, R. Braden, D. Estrin, S. Herzog, S., and S. Jamin. Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification. URL ftp://ftp.ietf.org/internet-drafts/draft-ietf-rsvp-spec-16.ps.