MODELS, MEDIA AND MOTION: USING THE WEB TO SUPPORT MULTIMEDIA DOCUMENTS

DICK C.A. BULTERMAN
CWI: Centrum voor Wiskunde en Informatica
Kruislaan 413, 1098 SJ Amsterdam, Netherlands

The World-Wide Web has been used extensively to present hypertext documents that have a limited mixture of text and simple graphics which are distributed via the public Internet. The performance characteristics of the Internet have made the delivery of complex multimedia documents (that is, documents that include time-based components) difficult. An effort is currently underway by members of industry, research centers and user groups to define a standard document format that can be used in conjunction with time-based transport protocols over Inter- and Intranets to support rich multimedia presentations. This paper outlines the goals of the W3C's Synchronized Multimedia working group and presents an initial description of the first version of the proposed multimedia document model and format.

1 Introduction

The World-Wide Web is generally seen as *the* embodiment of the information infra-structure in today's information age. The Web's clever application of traditional technologies and its universal acceptance among a wide range of users has provided an information sharing backbone that is unique in history. The success of the Web is based largely on the use of a simple document format [8] and a straightforward (sub)document transfer protocol [9]. Using nothing more than a text editor and (if possible) an existing document as an example, even the most novice users can create complex hypertext documents which, using a fetch-and-store transport protocol, can be accessed from a variety of client computers across the Web.

The simplicity of the Web's structure is both a blessing and a problem. It has been a blessing because it has allowed a wide range of users to participate in the information infra-structure. Unfortunately, this simplicity has also limited the types of information that can be placed in Web documents. In a age where even low-end PCs have some support for audio and often video media, the Web has offered little or no support for fetching and displaying such time-based media items through standard document interfaces. HTML documents cannot express the synchronization primitives required to provide a coordinated presentation, and the HTTP protocol cannot provide the guaranteed delivery of time-based media objects required for continuous media data.

The development of Java extensions to HTML, known as *Dynamic HTML* [5], provide one approach to introducing the necessary synchronization support into Web documents. This approach has the advantage that the author is given all of the control offered by a programming language in defining interactions within a document; this is similar to the use of the scripting language Lingo in CD-ROM authoring packages like Director [6]. Such an approach has the disadvantage that defining even simple synchronization relationships becomes a relatively difficult task for Web users who have little or no programming skills—the vast majority.

An alternative to using a programming language is the use of a declarative multimedia document format. In such a format, the control interactions required for multimedia applications are encoded in a text file as a structured set of object relations. The first system to propose such a format was CMIF [2],[3],[4]. Other more recent examples are RTSL [17] and MADEUS [10].

In this paper, we describe a new declarative format for Web-based multimedia applications. This format is being developed by the Synchronized Multimedia working group of the W3C. While the development of the format is still in its formative stages, a review of the principles of this work can be useful to developers and researchers who are interested in the general direction of multimedia support for the WWW.

Section 2 presents background material that is useful in understanding the transfer of multimedia data in open networks. We begin with a short description of a typical Web-based application (the Web News), and then follow with a description of the infrastructure that can be used to actually transmit document components. This section closes with more background on the W3C SYMM working group. Section 3 provides an overview of the major aspects of the evolving format for describing multimedia applications. Here, we consider the encoding of temporal and spatial aspects of a presentation, as well as some rudimentary specification of alternate behavior based on characteristics of the presentation environment. We also consider the initial support planned for hypermedia aspects of documents. Section 4 closes with a discussion of open issues and related work.

2 Web-Based Multimedia: Environment and Typical Applications

This section provides background information that will help define the types of applications and the support environment expected for first-generation Web-based multimedia documents. We begin with the description of a typical example, we then discuss the intended operational environment and close with a description of the W3C SYMM working group.

2.1 A Sample Document: The Web News

Multimedia applications have the general characteristic that they integrate a strong notion of time in a presentation. This is a sufficiently broad definition to encompass a wide range of applications, but it perhaps too broad to be of any great value for building a simple support mechanism for anything as anarchistic as the WWW. In order to focus our attention on the class of applications that this paper concerns itself with, we present an example of a generic Web application: that of a network newscast.[a]

Several media objects can be defined that can make-up such a newscast. Let us assume that, for whatever reason, the objects shown in Figure 1 have been selected to make up the presentation. For purposes of this example, we do not care how each of these objects have been created, nor do we care where they are stored. What we do care about is that they have not been pre-packaged into a composite object that is fetched from a single source.

While storing the audio and video as separate objects will increase the synchronization burden on the playback environment, it increases the flexibility of the over-all presentation. One could conceivably substitute one audio track for another without having to rebuild the entire application. (This can be useful in multi-lingual environments.) Such substitution is not to be taken lightly, however, since often some form of content-based synchronization will may required among data objects.

Figure 2 shows two views of the newscast example, taken at different times in the presentation. On the left side, we see a portion of the introduction of a story on the growth of the World-Wide Web. In this portion, the anchor is describing how sales of authoring software are expected to rise sharply in the next six months. Figure 2(b) shows a point later in the presen-

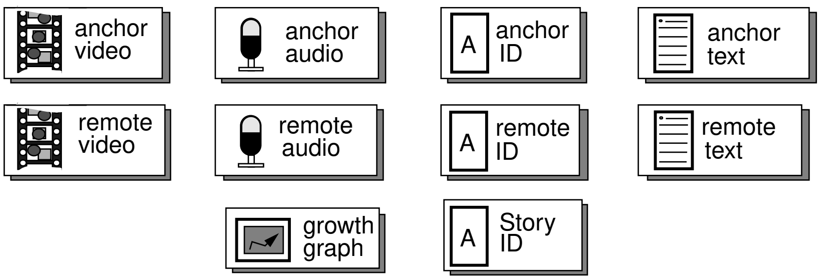
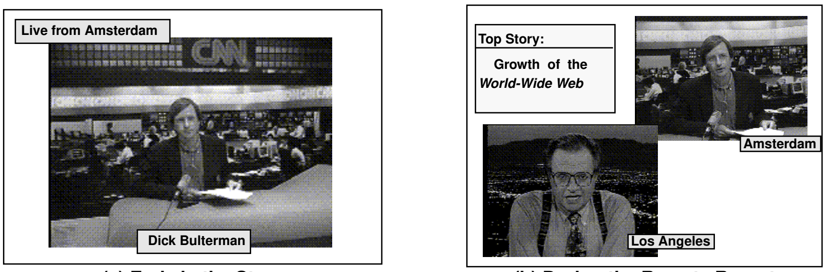


Figure 1: Media objects for use in (part of) a Web-based newscast.

[a]In 1991, as part of the first public CMIF paper, a network newscast was also used as a prototypical example. Sometimes the electronic world moves as slowly as the real one!



(a) Early in the Story **(b) During the Remote Report**

Figure 2: Two views of the Web newscast.

tation, when the anchor is chatting with a remote correspondent in Los Angeles, who is describing how local Hollywood stars are already planning their own audio/video homepages on the Web.

The presentation described up to this point is dynamic in terms of content, but static in terms of structure—the entire presentation is played as if it were a single, composite object. A more interesting extension of this example is given in Figure 3. Here we see the result of 'clicking' on the anchor: in addition to the presentation we first saw, an additional window has popped up which contains the anchor's home page. The ability to incorporate links to other pieces of content in the presentation transforms the static semantic structure in to a dynamic one, which is a powerful mechanism for creating complex presentations.

A total description of the implementation details of the Web newscast is beyond the scope of this paper. We will, however, refer to it as a running example in the sections below.

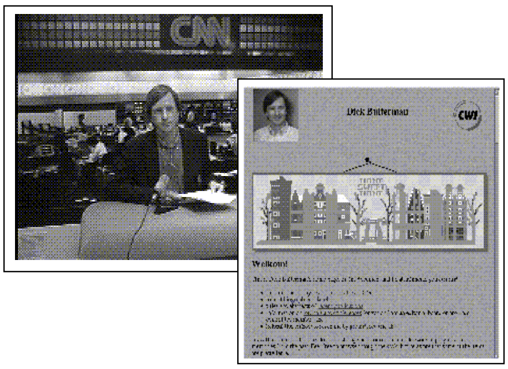


Figure 3: Augmenting information during the news.

2.2 *The Execution Environment*

Presentations of the type discussed in the previous sessions can be transmitted over any type of network infrastructure, including the null infrastructure: CD-ROM. In a CD-ROM environment, it is possible to analyze standard system characteristics to determine the feasibility of presenting a document on a particular system. In a networked environment, this is often impossible. Especially in connectionless networks, authors cannot know ahead of time how many transport resources will be allocated to any one server-client data stream; this problem increases when each piece of data is saved as a separate object on separate servers.

In order to bring order into the chaos of sending time-restricted data over the Internet, the Internet Engineering Task Force (IETF)[7] has been developing a number of protocols that can serve to better manage the transfer of information between clients and servers. These include protocols for resource reservation, real-time transport, real-time control and real-time streaming of data. The deployment of some of these protocols is just starting to become a reality over IP-based public networks. While it is not clear if all of the protocols will be supported by all components of the network infrastructure—the universal acceptance of resource reservation seems doubtful—there is a concerted effort underway to augment the strict fetch-and-buffer approach used by HTTP.

Figure 4 shows the general relationship of these protocols to one-another. IP and UDP are standard protocols from the Internet suite. They are often implemented within the operating system kernel or as part of particular network device controllers. The other protocols are currently implemented as part of the application itself, rather than as part of the low-level operating system support. This is probably a transitional situation.

RSVP [12],[13] is a resource reservation protocol that was designed for uni-directional multicast applications, but which also could be used for sim-

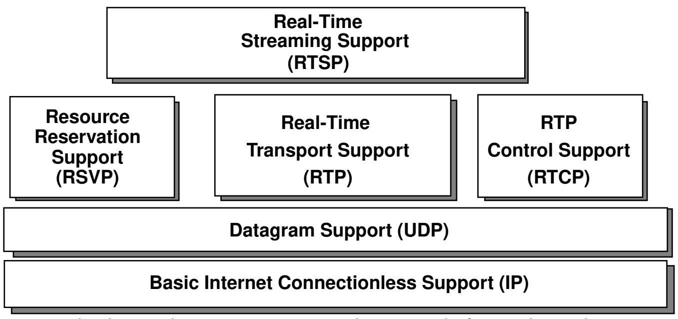


Figure 4: Types and relationships among network protocols for multimedia.

plex unicast applications. As part of application initialization, a request can be made of intermediate network components to reserve resources at a particular quality of service (QoS) level for the lifetime of the application. From a Web perspective, the most interesting aspect of RSVP is that it is the first serious attempt at defining a network-wide reservations scheme. While it seems unlikely that RSVP will play a major short-term role in public networks, it may be a useful tool within Intranets for guaranteed end-to-end bandwidth allocation.

RTP [15],[16] and RTCP [14] are the *Real-time Transport Protocol* and the *RTP Control Protocol*, respectively. Together they form a pair of transmission protocols that can serve as the basis for supporting time-based data delivery. While it may be natural to expect that a real-time protocol provides services to guarantee real-time (or on-time) packet delivery, this is not the case with RTP/RTCP. Instead, these protocols provide a framework and a set of building blocks with which a given application can create its own servicing algorithms to support on-time delivery of data packets. In this way, the particulars of the application and the data types being transferred can be used to determine the best support strategy, rather than relying on RTP/RTCP providing a 'one size fits all' type of service.

In practice, both RTP and RTCP make use of a local transport protocol to actually ship data between the source and destination(s) of a transfer. RTP/RTCP support unicast and multicast transfers if the underlying transport mechanism does as well. RTP is a packaging protocol that allows application data to be time-stamped and wrapped inside transport-level packets, such as those provided by UDP. From time-to-time, RTCP packets are sent between the source and destination(s) with transfer statistics. These can be used by the sender/receiver to adjust the manner in which data is buffered at either end of the transfer, or it can be used to dynamically select appropriate data encodings—but only if this is supported by the application itself.

RTSP [18] is a relatively recent streaming protocol that can be integrated with the protocols discussed above. (A streaming protocol is one that does not wait for an entire object to be delivered before rendering can begin.) As with RTP, RTSP does not a complete streaming solution; rather, it provides a framework in which applications-level streaming support can be implemented. To date, RTSP has been used commercially by Progressive Networks as part of their RealAudio/RealVideo suite [11].

Having a set of transport and control protocols is an essential basis for supporting multimedia applications, even if these "protocols" provide only skeleton services. For the work described in this paper, these skeletons provide a common starting point.

2.3 Goals of the W3C Synchronized Multimedia Working Group

Until the beginning of 1997, there was no coordinated effort for non-proprietary multi- and hypermedia support through the World-Wide Web. In February of that year, the WWW Consortium (known as W3C [21]) initiated a working group on synchronized multimedia [22]. The SYMM working group was formed to develop a specification which carries the working title of (H)MML, or the (Hyper-) Media Markup Language [23].[b]

The development of (H)MML grew out of a realization that a large class of hypermedia applications could (and probably should) be represented in a declarative format rather than as a computer program or high-level script. A declarative specification is often easier to edit and maintain than a program-based specification, and it can potentially provide a greater degree of accessibility to the network infrastructure by reducing the amount of programming required for creating any particular presentation. The success of HTML for hypertext documents has demonstrated the willingness and ability of Web users to create documents using a simple, structured format. In many ways, (H)MML builds on this concept.

The W3C SYMM group has restricted its attention to the development of a common format, without specifying any particular playback or authoring environment. At present, approximately ten organizations have indicated an interest in developing prototype play-out environments. There also has been some support for developing or adapting authoring environments to generate (H)MML encodings.

3 (H)MML: Structured Document Format Specification

This section presents a summary of the proposed document format that is being developed by the W3C SYMM working group. We begin with a statement of the general characteristics of the format, and then consider the format's major components:

- *temporal specifications*: mechanisms to encode the temporal structure of the application and the refinement of the relative start and end times of events;
- *spatial specifications*: the primitives provided to support simple document layout;
- *alternative behavior specification*: the primitives to express the various

[b] As of this writing, the name used within the working group is MML. Since this name is already in use for other formats, I have added the (H) prefix to uniquely identify the format. This name may be changed upon public release of the specification.

optional encodings within a document based on systems or user requirements; and
- *hypermedia support*: mechanisms for linking parts of a presentation.

Each sub-section starts with a statement of general principles and then a description of the model components.

The goal of (H)MML is to provide a declarative, text-based encoding of the behavior of hypermedia applications. Once encoded, the document should be able to be played on a wide-range of (H)MML browsers. Such browsers may be stand-alone presentation systems that are tailored to a particular user community or they could be integrated into standard browsers.

3.1 Media Objects

Perhaps the most fundamental aspect of (H)MML is that it is an *integrating* format. Unlike HTML, an (H)MML document contains only structure and media object description information—it does not contain any data associated with the objects themselves. The display software for (H)MML documents (either stand-alone players or adapted browsers) must be able to render the individual data components based on the description in the (H)MML file. The use of an integrating format is essential for multimedia applications: unlike text, even the most trivial audio or video object can contain massive amounts of data. Storing these items within the document would make its size unmanageable.

References to individual media objects are made via *media object instance* specifiers. These are of the general form:

```
<{type} SRC="{protocol}:{location}/{name}" {attributes}>
```

where `type` indicates the type of data (such as text, image, video, etc.), the `SRC` field provides the familiar object reference, and the `attributes` fields provide details that are discussed below.

3.2 Temporal Specifications

If we were to define the Web Growth story outlined in 2.1 in terms of its overall structure, we would wind up with a representation similar to that in Figure 5. Here we see that the story starts with a opening sequence (perhaps containing a logo and a title), and ends with a similar closing segment. In between is the “meat” of the story. It contains an introduction by the local anchor, followed by a report by the remote correspondent, and then concluded by a wrap-up by the local anchor. This ‘table of contents’ view defines the basic structure of the story. It may be reusable, in that many stories may be similarly structured.

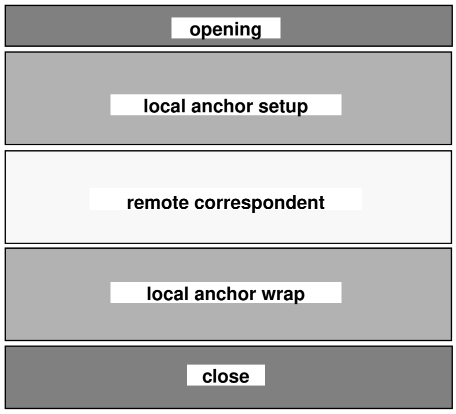


Figure 5: The structure of the Web Growth story. (Time flows from top to bottom.)

This structure view, without any references to particular media objects, does not provide sufficient detail to describe an example, but it can be used to define a general sequence of story elements. A more common view of an application is shown in Figure 6, where we see a timeline of the Web Growth story. Rather than illustrating structure, this view shows each of the compo-

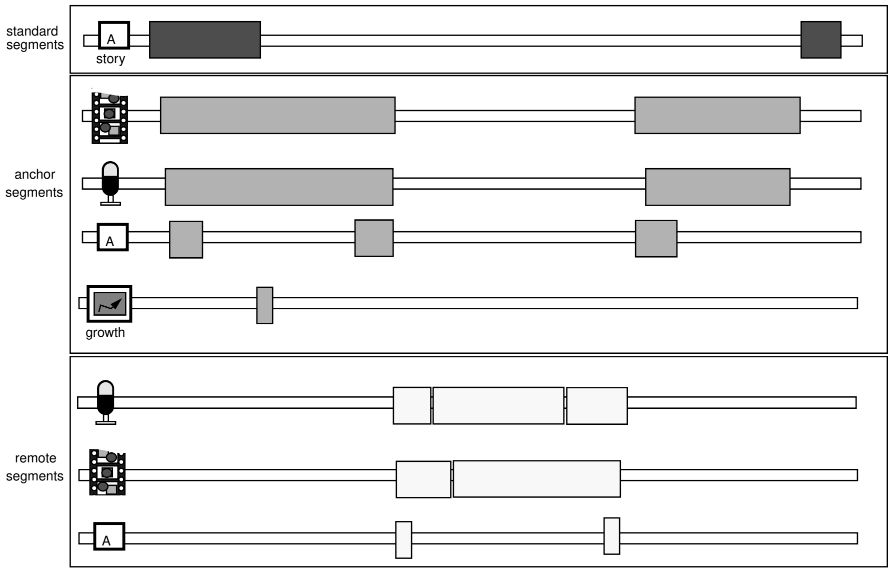


Figure 6: A timeline view of the Web Growth story. (Time flows from left to right.)

nents and their relative start and end times. (Note that, as part of the initial anchor setup, a reference is made to a graph that is active during only a part of the presentation.) While timelines provide an effective graphical representation of an application—so long as that application is not too complex—it is not a useful basis for encoding an application in a portable manner.

The timeline view does provide an insight into the actual temporal relationships among the elements in the presentation. Where Figure 5 assumed that the opening segment occurred entirely before the first anchor segment, we see that there is actually some overlap of the opening text and the anchor audio and video. The remaining structural partitioning remains correct, although it is clear that within each element a number of media events occur in parallel or sequentially.

If we were to combine the structure and timeline views, we might end up with the representation shown in Figure 7. This representation shows that the story is made up of a number of events that occur sequentially or in parallel. Within the parallel events, some start at the exactly the same time, while others start at an offset relative to each other.

As with the timeline, a structure diagram is an unhandy way to encode a portable document. Instead, the (H)MML format uses the following two structuring elements, taken from CMIF:

`<seq> ... </seq>`: A collection of objects that occur in sequence.

`<par> ... </par>`: A collection of objects that occur in parallel.

Elements defined within a `<seq>` group have the semantics that a successor element is guaranteed to start after the completion of a predecessor element. Elements within a `<par>` group have the semantics that, by default, they all start at the same time. Once started, all elements are active for the time

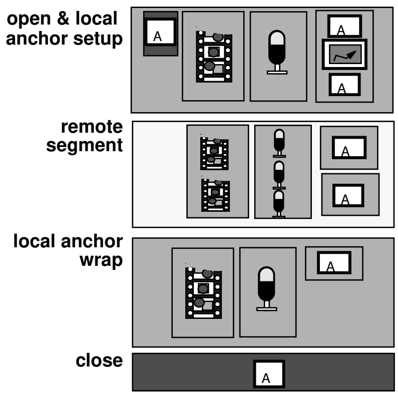


Figure 7: The revised structure of the Web Growth story.

determined by their encoding or for an explicitly defined duration. Elements within a `<par>` group can also be defined to end at the same time, either based on the length of the longest component or on the end time of an explicit master element. Note that if objects within a `<par>` group are of unequal length, they will either start or end at different times, depending on the attributes used to define the group.

The structural elements can be nested to describe applications of arbitrary complexity. A partial encoding of our example story is shown in Figure 8. (Note that the syntax of the object references is generalized to improve readability.) The encoding shows that the Web Growth story contains a sequence of parallel groups, some of which contain nested `<seq>` and `<par>` elements. In the first `<par>`, we see that the text object *story_heading* is presented in parallel with the initial anchor setup. This setup consists of an audio and video stream that is played in parallel, along with a sequence of media objects: a text label, followed by an image and then a text label.

If we compare Figure 8 with Figure 7, we see that the (H)MML encoding as it is presented gives only the coarse structure of the application's temporal

```
<mml>
   <seq>
      <par>
         <text src="story_heading.html" />
         <par>
            <video src="rtsp:local_anchor.mpg" />
            <audio src="local_anchor.aiff" />
            <seq>
               <text src="anchor_name.html" />
               <image src="growth_graph.gif" />
               <text src="anchor_location.html" />
            </seq>
         </par>
      </par>
      <par>
         <seq>
            <video src="rtsp:rem0.mpg" />
            <video src="rtsp:rem1.mpg" />
         </seq>
         <seq>
            <audio src="rtsp:rem0.aiff" />
            <audio src="rtsp:rem1.aiff" />
            <audio src="rtsp:rem2.aiff" />
         </seq>
      </par>
. . .
   </seq>
</mml>
```

Figure 8: The basic encoding of the Web Growth story.

relationships. For example, the actual overlap between the *story_heading* and the anchor audio/video tracks is minimal. The timeline also shows that the display of the label *anchor_name* is to happen shortly after the anchor appears in view. Also, the image *web_growth* in the nested `<seq>` group needs to appear on the screen when the anchor refers to it in the story. To handle these types of situations, (H)MML provide three types of timing control relationships:

- *explicit durations*: a `DUR="`*time*`"` attribute can be used to state the presentation time of the object;[c]
- *absolute offsets*: the start time of an object can be given as an absolute offset from the start time of the enclosing structural element by using a `BEGIN="`*time*`"` attribute;
- *relative offsets*: the start time of an object can be given in terms of the start time of another sibling object using a `BEGIN="`*object_id* + *time*`"` attribute.

(Unless otherwise specified, all objects are displayed for their implicit durations—defined by the object encoding or the length of the enclosing `<par>` group.)

The specification of a relative start time is a restricted version of CMIF's *sync_arcs* [4] to define fine-grain timing within a document. At present, only explicit time offsets into objects are supported, but a natural extension is to allow *content markers*, which provide content-based tags into a media object.

If we use the attributes defined above, the initial part of the application's encoding can be re-written as illustrated in Figure 9.

```
<mml>
   <seq>
      <par>
         <text dur="8.2s" src="story_heading.html" />
         <par sync>
            <video src="rtsp:local_anchor.mpg" />
            <audio id="a1" src="local_anchor.aiff" />
            <seq>
               <text begin="2.0s" dur="3s" src="anchor_name.html" />
               <image begin="a1+12.3s" dur="3.6s"
                    src="growth_graph.gif" />
               <text src="anchor_location.html" />
            </seq>
         </par>
      </par>
. . .
   </seq>
</mml>
```

Figure 9: The refined encoding of the Web Growth story.

[c]The syntax used to express time is still under discussion within the group.

3.3 Layout Specifications

Since HTML is based on a text flow model, one of its advantages is that an author is able to define a presentation without worrying about the exact positioning of individual objects in that presentation. The actual "look and feel" of the document was determined at run-time depending on the screen space allocated to the browser and on user preferences. This de-coupling of content and presentation led many in the SYMM group to consider presentation layout to be beyond the scope of the format. While some relationship does exist between the timing hierarchy and the ultimate presentation layout, it became clear that, for multimedia applications, extra facilities were required to determine the relative positioning of media objects.

A generalized solution to presentation layout was still considered outside the scope of the format, but it has been agreed that, to support some form of inter-operability among the potential variety of (H)MML players, some rudimentary form of layout control is required. To this end, it was decided that four types of layout schemes would be supported by the format:

- *the null (default) layout*: a layout scheme that is appropriate for very simple documents (those containing one video and one audio, each of which consume all of the available resources);
- *a bare-bones basic layout*: basic positioning is supported, without trying to solve the general layout problem;
- *a hook for external layouts, with (H)MML as the master*: an (H)MML document forms the basis of a presentation, but which uses facilities available in a particular player;
- *a hook for external layouts, with (H)MML as the slave*: an external player starts an (H)MML document as part of a larger presentation.

It was further agreed that all (H)MML renderers should support the same semantics for default and basic layout—thus providing a means for maintaining compatibility—and that each renderer would also have the option to support its own master or slave layout semantics. The creator of an (H)MML document would be free to choose the layout scheme which met the application's needs.

The general capabilities of the `mml-basic` layout specification is given in Figure 10. Each visible or audio object is rendered to a layout *channel*. (These

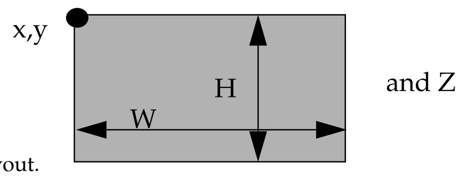


Figure 10: `MML−Basic` layout.

Figure 11: A channel partitioning within the Web news application.

channels have their roots in similar constructs used in CMIF and RTSL.) A channel is an abstract entity that can describe screen space, audio channels, or any other rendering resource. Associated with a screen channel are x, y coordinates that define the upper-left anchor point of the channel, plus a `HEIGHT` and `WIDTH`, measured in pixels or percentages. Channels also have an integer `Z` depth associated with them, with greater values indicating objects that are closer to the viewer. As such, the channel is a reusable, indirect reference to a set of presentation coordinates. Figure 11 shows the channel map associated with a part of the Web newscast. This portion has six visible channels (plus associated audio channels). The presentation of all channels is managed by the renderer.

Each media object instance contains a channel reference:

```
<video href=”rtsp:anchor.mpg” channel=”anchor”>
```

which refers to a similar tag in a layout specification. The layout specification itself has the form shown in Figure 12. In order to accommodate systems that

```
<mml>
   <layout type=”text/mml-basic”>
      ...
      <channel name=”anchor” x=”50%” y=”100” z=1
        height=”256” width=”256” />
      ...
   </layout>
   <seq>
. . .
      </par>
   </seq>
</mml>
```

Figure 12: Defining layout specifications.

(a) unconstrained growth of an HTML text object **(b) constrained growth of an HTML text object**

Figure 13: Constraining layout regions.

have richer (or poorer) layout semantics than `mml-basic`, an alternative layout definition scheme is provided. This is discussed in Section 3.4.

If a particular media object has a "natural" height and/or width, these values can be left unspecified. Care must be taken to make sure that objects of undefined size are bound to a particular screen area; if not a situation such as that illustrated in Figure 13 could occur.

3.4 Alternate Behavior Specifications

If the presentation infrastructure for an (H)MML file were known beforehand, one file could be tailored to that environment easily. Unfortunately, this is nearly never the case in the Web. Clients will access a document from a wide range of locations, and will encounter a wide range of transmission and server delays.

Providing truly adaptive documents, that match their performance and appearance characteristics to the resources available is a fascinating research topic. (It is fascinating in large part because it is unsolved in the general case.) In the context of (H)MML it was felt that some form of support for adaptive behavior was required, even for simple first-generation documents.

The solution adopted by the SYMM group was to provide a means for defining alternate behavior within a document, but to rely totally on the presentation environment to resolve which of the alternatives would be selected at run-time. This selection could take place based on profiles, user preference, or environmental characteristics. Clearly, specifying this type of behavior within the confines of an adaptive format is a challenging proposition.

As an example of specifying alternate behaviors, consider the following:

```
<switch>
   <audio profile=bad_uk src=low-res.aiff />
   <audio profile=good_uk src=hi-res.aiff />
   <audio profile=bad_nl src=low-res.aiff />
   <audio profile=good_nl src=hi-res.aiff />
</switch>
```

In this fragment, we assume the presence of four alternate audio files. Two of these files contain English-language audio, and two contain Dutch-language audio. A high resolution and low resolution version is available for each language. At run-time, the player could evaluate the alternatives based on its own algorithms and select the alternative that is most appropriate. Since all choices are semantically equal, any particular item can always be considered to be a "correct" choice (if perhaps not optimal); as a result, a player could always choose to select the first alternative if it wished.

An example of the application of alternative is the use of multiple layout specifications in a document. Suppose that an author wanted to guarantee some measure of compatibility across all players, but that s/he also wanted to exploit the fancy features offered by a particular environment. In this case, the following specification could be used:

```
<switch>
   <layout type="text/mml-basic">
      . . .
   </layout>
   <layout type="text/cmif">
      . . .
   </layout>
</switch>
```

If the application was played on a non-CMIF player, then it could choose to use the `mml-basic` layout features. If it was played on a CMIF player, then CMIF's multi-window functionality could be exploited.

3.5 Hypermedia and (H)MML

As the current Web has demonstrated so effectively, a networked information infrastructure is based in large part on its ability to reference related pieces of information from within a presentation. With HTML, this is a straightforward process: each document has a single focus (the browser window or frame) and anchors and links can be easily placed within the document text. In an (H)MML presentation, the situation is much more difficult. First, the location of a given anchor may move over time—and even if it does not move, it still may be visible for only part of the object's duration. Second, since (H)MML is an integrating format, conflicts may arise on ownership of

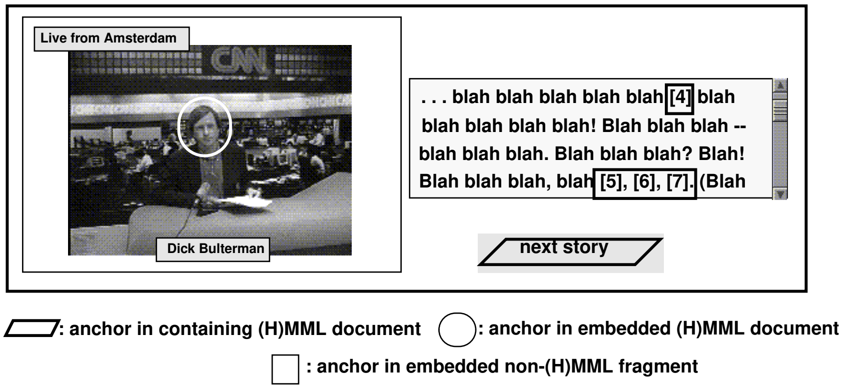


Figure 14: Problems when integrating hyper-navigation and (H)MML.

anchors and the semantics of following any given link.

As an example of the problems that can be encountered with links, consider the situation outlined in Figure 14. Here we see three visible channels, one containing an embedded (H)MML presentation, and two containing HTML text. The top HTML text is a conventional page (with internal links) while the bottom HTML text (labelled *next story*) contains an (H)MML link. Since (H)MML integrates many presentation data types, it is important to know if following a link from any given anchor will result in intra-object or inter-object navigation. We can identify three situations in which a link can be defined: as a link defined by the containing (H)MML document, as a link defined by an embedded (H)MML document—that is, an (H)MML sub-document, or a link defined within a non-(H)MML component.

If the link is defined by a containing (H)MML document, such as the *next story* link, activation of the link affects the presentation of the whole (H)MML document. This effect depends on the value of a `SHOW` attribute, which may have values:

- `REPLACE`: the presentation of the destination resource replaces the complete, current presentation (this is the default);
- `NEW`: the presentation of the destination resource starts in a new context (perhaps a new window) not affecting the source presentation; or
- `PAUSE`: the link is followed and a new context is created, but the source context is not replaced but is suspended.

If the link is defined by an embedded (H)MML document, such as by following the link placed over the head of the news anchor, activation of the link affects only the embedded (H)MML document. The effect depends on the value of the `SHOW` attribute as described above.

If the link is defined by an non-(H)MML document which is embedded in an (H)MML document, such as following one of the references in the text in the upper right of the figure, link traversal can only affect the presentation of the embedded component and not the presentation of the containing (H)MML document. This restriction may be released in future versions of (H)MML.

General support for hypermedia is a complex task. Interested readers are invited to study the approaches defined for the Amsterdam Hypermedia Model [3], which serves as the basis for the current (H)MML proposal.

4 Current Status and Future Directions

As of the writing of this article (late September, 1997), may of the details of the encoding of (H)MML were being finalized. For the latest version of the specification, interested readers should consult W3C's web pages.

As part of the W3C approach to defining Web standards, the format will be submitted to interested parties, who can choose to develop prototype implementations. (At present some eight organizations are involved in developing prototype environments.) These implementations will be evaluated against a set of standard application examples to determine the viability of the encoding format. The format—plus changes—will then be submitted to the full membership.

While the specific choices made in developing (H)MML are interesting in their own right, perhaps the most interesting aspect of the format is that is can provide a common base for future research on various aspects of networked multimedia systems. Where in the past true multimedia applications were could never inter-operate with research platforms, (H)MML provides a foundation that will allow comparative examples to be developed that can support experimental research. If this can be accomplished, then the work put into the development of the specification will be worth the considerable effort it has required.

Acknowledgments

The work of the W3C's SYMM working group was coordinated by Philipp Hoschka of W3C/INRIA. A complete list of contributors is available at [22].

CWI's activity in this project is funded in part through the ESPRIT-IV project CHAMELEON [1] of the European Union. Additional sources of funding have been the ACTS SEMPER [19] and the Telematics STEM [20] projects.

References

In the spirit of Web-based information exchange, references to publications have been given to reports available via Web servers as much as possible.

[1] CHAMELEON: An Authoring Environment for Adaptive Multimedia Presentations. ESPRIT-IV Project 20597. See http://www.cwi.nl/Chameleon/.
See also http://www.cwi.nl/~dcab/.

[2] 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/.

[3] 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. See also http://www.cwi.nl/~dcab/.

[4] 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. See also http://www.cwi.nl/~dcab/.

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

[6] Macromedia, Inc.: *Director 6.0.* http://www.macromedia.com/software/director/

[7] IThe Internet Engineering Task Force: *Home Page.*
http://www.ietf.cnri.reston.va.us/

[8] W3C: *HTML Version 3.2.* See http://www.w3.org/MarkUp/.

[9] W3C: *Jigsaw Overview.* See http://www.w3.org/Jigsaw/

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

[11]PRealNetworks (formerly Progressive Networks). *Home page.*
See http://www.real.com/

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

[13]Zhang, L., Deering, S., Estrin, D., Shenker, S., and D. Zappala, *RSVP: A NewResource ReSerVation Protocol*, ftp://parcftp.xerox.com/pub/net-research/rsvp.ps.Z, IEEE Network, Vol 7, no 5, pp 8-18, September 1993.

[14]D. Bettati, et al: Connection Establishment for Multi-Party Real-Time Communication, Proc. NOSSDAV '95, Durham NH 1995. See http://www.cs.berkeley.edu/~amit/research/Postscript/estab-nossdav.ps.

[15]H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, rfc1889: *RTP: A Transport Protocol for Real-Time Applications*, 01/25/1996. http://www.cis.ohio-state.edu/htbin/rfc/rfc1889.html.

[16]H. Schulzrinne, *RTP*, `http://www.cs.columbia.edu/~hgs/rtp/`.
[17]RTSL: *Real-Time Streaming Language*. See `http://www.real.com/server/intranet/index.html`
[18]H. Schulzrinne: *A real-time stream control protocol (RTSP)*, `http://www.cs.columbia.edu/~hgs/rtsp/draft/draft-ietf-mmusic-stream-00.txt` (11/26/96).
[19]SEMPER: *Secure Electronic Marketplace for Europe*, EU ACTS project 0042, 1995-1998. See `http://www.semper.org/`.
[20]STEM: *Sustainable Telematics for the Environment*, EU Telematics Project EN-1014, 1995-1996.
[21]W3C: The World-Wide Web Consortium. `http://www.w3.org/`.
[22]W3C Synchronized Multimedia Working Group. `http://www.w3.org/AudioVideo/Group/`.
[23]W3C MML Draft Specification. `http://www.w3.org/AudioVideo/Group/WD-symm-format.html`.