

Dynamic Multimedia Integration with the WWW

Koji Sato, Toshihiro Takada, Shigemi Aoyagi, Toshio Hirotsu, and Toshiharu Sugawara
NTT Network Innovation Laboratories
Nippon Telegraph and Telephone Corporation
3-9-11 Midori-cho, Musashino, Tokyo 180-8585 Japan
{koji, takada, aoyagi, hirotsu, sugawara}@t.onlab.ntt.co.jp

Abstract

This paper presents a novel framework for seamlessly integrating continuous media, such as audio and video, with the World Wide Web (WWW). Continuous Media with the Web (Cmew) enhances the interactivity of continuous media by associating hyperlinks with spatial-temporal parts of the media. The scenario control architecture in Cmew provides flexible and dynamic control over continuous media in multimedia documents. The Cmew media player has been implemented as a Java applet, which enables its use in the current WWW environment.

1 Introduction

Many multimedia applications based on the World Wide Web (WWW) have appeared as a result of advances in data compression techniques and high-speed networks. The WWW provides flexible browsing and navigation of various kinds of information on the Internet through hyperlinks. However, support for continuous media, such as audio and video, is less advanced. It is limited to VCR-like operations, so hyperlinks cannot be associated with the spatial-temporal subparts of the media.

Hyperlinks for continuous media have been proposed in many hypermedia systems [2][3][5][6][7][8][12]. However, these systems provide only hard-wired single-headed hyperlinks, and the hyperlinks need to be activated by users. Also, they cannot dynamically manipulate media objects and the temporal relationships between them. These limitations restrict the potential of multimedia applications. To create sophisticated interactive multimedia applications, dynamic control over continuous media is required.

We have developed a scenario control architecture for structuring synchronized continuous media in an interactive multimedia document. Event-driven hyperlinks and script-based scenario descriptions provide flexible and dynamic control over continuous media in the document.

2 Scenario Control Architecture

The goal of Continuous Media with the Web (Cmew) [11] is to seamlessly integrate continuous media with WWW en-

vironments. Cmew allows users to uniformly manipulate various kinds of information on the Internet, such as text, images, audio, and video, through hyperlinks.

2.1 Structure of Synchronized Continuous Media

In Cmew, continuous media in a multimedia document are organized in tree structure. The leaf nodes are "movies", and intermediate nodes are "scenarios". The top of the tree is a root scenario. Each media node is addressed by a Uniform Resource Locator (URL), so it can be located on any WWW server on the Internet.

Movies represent the whole or part of media data, such as MPEG-1, QuickTime, or AVI files. Fragment identifiers attached to the URLs can specify the temporal subparts of the media data. Movies are primitive units for structuring a media tree.

Scenarios are containers for media nodes. They abstract the synchronized presentation of media nodes. Scenarios group child nodes and describe temporal relationships between them using hyperlinks. Hyperlinks are associated with spatial-temporal regions of the nodes. Scenarios are organized hierarchically, referring to other scenarios as their child nodes. Several scenarios may share the same child node, and associate different hyperlinks with it.

Figure 1 shows the hierarchical structure of media nodes in a multimedia document. The root scenario has two movies and two scenarios as its child nodes. Their temporal relationships are defined by hyperlinks in the scenario. The child scenarios have their own structures.

2.2 Hyperlinks

Hyperlinks represent temporal relationships between media nodes and associations of information related to them. They are activated by "events" which media nodes generate. Every media node, except for the root scenario, sends the events to its parent scenario in order to inform the parent scenario of the user's actions on it or its state transitions. The parent scenario is responsible for handling the events for activating hyperlinks. The events are generated when:

- The user manipulates the mouse or the keyboard within the display area of a movie.

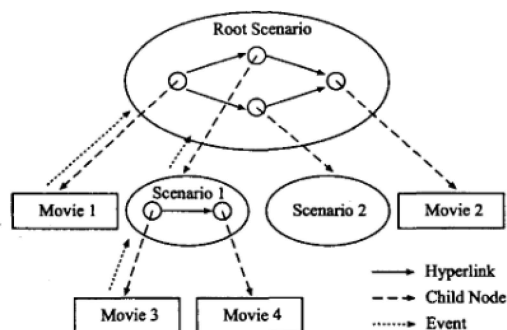


Figure 1: Hierarchical Structure of Media Nodes

- Media nodes start or end playing.
- The playing time of a movie reaches a specified time.

That is, the hyperlinks are actuated by users, or are automatically followed by state transitions of the nodes.

Each hyperlink has a conditional expression that specifies events. When the specified events are generated, the expression is satisfied and the hyperlink is activated. Multiple hyperlinks can be followed simultaneously. One of the following forms is used to describe the condition:

- *source anchor*
- $\text{and}(\text{source anchor}_1, \dots, \text{source anchor}_n)$
- $\text{or}(\text{source anchor}_1, \dots, \text{source anchor}_n)$

A *source anchor* specifies a particular event. It consists of a source node which generates the event, the type of event, and a hotspot defined by a spatial region and a temporal region on the source node. The spatial region includes a shape descriptor (rectangle, oval, or polygon) and coordinates data defining the area on the source node. The temporal region contains a time interval during which the hotspot is available. When the specified event is sent, the *source anchor* is satisfied. If the conditional expression is an **and** or **or** expression, the hyperlink has multiple source anchors; i.e., it is an “aggregate” hyperlink. An **and** expression is satisfied when all of its source anchors are satisfied. An **or** expression is satisfied when any one of its source anchors is satisfied.

The destination of a hyperlink is:

- A media node contained in the scenario.
- Any related information on the Internet.
- A built-in or user-defined function in the scenario.

When the hyperlink is activated, the destination is presented or invoked. The attributes of the hyperlink can specify the spatial position of the destination media node in the document, or the window in which the related information is displayed. Also, they can describe whether the source nodes are retained or replaced when the hyperlink is followed.

The destination functions improve the interactivity and extensibility of the scenarios. For instance, they can dynamically add or remove media nodes and hyperlinks to or from the scenarios, and control the playback of the scenarios. They can also determine the destinations of the hyperlinks in response to user’s inputs.

Hyperlinks represent synchronization of media nodes. A hyperlink between two media nodes causes these nodes to play in sequence. The start-time synchronization between media nodes is achieved through the simultaneous activation of multiple hyperlinks. An aggregate hyperlink is used for the end-time synchronization.

Hyperlinks are defined outside the media nodes to which they are applied. This approach has several advantages. The hyperlink can be easily managed and changed without any modification of the media nodes, and hyperlink definitions are independent of the media formats. These features encourage the efficient sharing and reuse of the media nodes.

3 Implementation

The Cmw media player is the implementation of the scenario control architecture. This media player has been developed as a Java applet to incorporate Cmw into the WWW environment. In contrast to other hypermedia systems, Cmw requires only a Java-enabled WWW browser to present multimedia documents.

3.1 Cmw Media Player

The Cmw media player uses Java Media Framework (JMF) version 1.x [10] to display continuous media. JMF provides application programming interfaces (APIs) used to present time-based media in Java programs. Discrete media, such as text and images, are described in HTML files, and displayed directly in WWW browsers.

For security reasons, network connections for a Java applet are restricted to the host from which the applet comes. Therefore, if scenarios refer to media nodes on other hosts, the Cmw media player must be digitally signed and granted permission to access these hosts.

The Cmw media players are positioned in an HTML file. When a scenario plays multiple movies simultaneously, the HTML file must contain a sufficient number of media players to present them. Each media player is given a unique name, and is referred to in the scenario by that name.

3.2 Movies

Movies are media data that JMF 1.x supports, including MPEG-1, QuickTime, and AVI files. In addition to VCR-like operations, the Cmw media player makes movies “navigable” using hyperlink definitions in scenarios. To inform users of the existence of hyperlinks, the shape of the mouse cursor is changed when the cursor moves into a hotspot on a movie. The destination of the hyperlink is also presented on the message field of the media player.

3.3 Scenarios

Scenarios are described in scripts. Pnuts [9], which is a script language for Java, is used for writing scenarios. Pnuts provides access to all public fields and methods of the Java APIs. It can also use variables and first-class functions.

Script-based scenario descriptions make it possible to compose dynamic and interactive scenarios. They incorporate programming features into the scenario construction. Scenario scripts enable the scenarios to dynamically modify their structures and behaviors.

The Cmw media player loads the scenarios over the network, and invokes the Pnuts interpreter to execute them. The URL of a root scenario is passed to the media player as a parameter in the HTML file.

The Cmw class library is implemented to construct scenarios. This class library includes Java classes to represent movies, scenarios, anchors, and hyperlinks. These classes enable the integrated and transparent handling of various media types. The Cmw class library and the standard Java class libraries can be used in scenario scripts.

An example of a scenario script is shown in Figure 2. The scenario starts the movie *m1* on the media player *player1* specified by the function *start()*. Related information *info* is associated with the spatial-temporal region defined by the *Anchor* on *m1*. This information is presented in the window *InfoWindow* when the hyperlink is activated. Since *m1* also has hyperlinks to scenarios *s1* and *s2*, clicking the mouse on the specified hotspot will cause *s1* and *s2* to start playing on *player1* and *player2*, respectively. When both scenarios reach their ends, an aggregate hyperlink with an *and* condition is activated, and the movie *m2* starts playing on *player1*. In this scenario, *m2* has hyperlinks whose destinations are functions. The function *rateChange()* changes the playback speed of a movie, thus allowing users to dynamically control the movie. When *m2* reaches the end of the media file, the function *end()* is invoked, and the scenario finishes the execution.

Each scenario has its own context. Variables and functions defined in a scenario are only accessible within it. Therefore, the scenario can refer to other scenarios without name conflicts becoming a problem. This facilitates the reuse of scenarios.

Hyperlink activation might be delayed when the destination is a media node, because all the media data must be loaded through networks before presentation¹. To minimize this delay, the Cmw media player loads the media nodes in advance. When the player starts playing a movie, it looks for hyperlinks whose source node is the movie. If the destinations of the hyperlinks are media nodes, the media player starts loading the media files in the background while playing the movie. This preloading mechanism helps create highly responsive interactive applications.

Figure 3 shows the Cmw media player. The player is presenting a movie in the left window and a related document associated with an actor in the movie in the right window.

¹JMF 1.x does not support streaming.

```
m1 = Movie("http://.../movie1.mpg#start=0.0,end=30.0")
m2 = Movie("http://.../movie2.mpg")
s1 = Scenario("http://.../scenario1.pnut")
s2 = Scenario("http://.../scenario2.pnut")
info = Info("http://.../info.html")

function rateChange(movie, rate) {
    movie.stop()
    movie.setRate(rate)
    movie.start()
}

Hyperlink(Anchor(m1, "MouseClicked",
    Period(0.0, 10.0),
    Rectangle(100, 150, 50, 50)),
    info, "InfoWindow")
Hyperlink(Anchor(m1, "MouseClicked",
    Period(10.0, 30.0),
    Rectangle(20, 20, 180, 180)),
    s1, "player1")
Hyperlink(Anchor(m1, "MouseClicked",
    Period(10.0, 30.0),
    Rectangle(20, 20, 180, 180)),
    s2, "player2")
Hyperlink(and([Anchor(s1, "EndOfScenario"),
    Anchor(s2, "EndOfScenario")]),
    m2, "player1")
Hyperlink(Anchor(m2, "MousePressed",
    rateChange, [m2, 5.0f])
Hyperlink(Anchor(m2, "MouseReleased",
    rateChange, [m2, 1.0f])
Hyperlink(Anchor(m2, "EndOfMedia"), end)

start(m1, "player1")
```

Figure 2: Scenario Script

4 Related Work

Hyperlinks for continuous media have been proposed in many hypermedia systems. Elastic Charles [2] is a hypermedia journal which provides an additional structural layer on audio, video, and text to link related portions of these media. The Amsterdam Hypermedia Model [5] is an extension to the Dexter hypertext reference model [4], adding timing constraints and link contexts to the basic hypertext notions. HyperCafe [8] is an experimental hypermedia prototype which provides temporal, spatial-temporal, and textual links to present alternative video clips and related text. In these systems, however, the sources and destinations of the hyperlinks are limited to only the data stored in the systems. Therefore, they do not allow users to interact with continuous media and other information on the Internet.

WWW-based hypermedia systems have been developed as the explosive growth of the WWW. Vosaic [3] incorporates video and audio into hypertext pages for the WWW. Vosaic associates continuous media with meta-information that describes a hierarchical structure and hyperlink specifications for these media. The Hypermedia Presentation and Authoring System [12] incorporates the Media Relation Graph which describes the temporal relationships between media objects in hypermedia documents. MPEG-4 [7] is the inter-

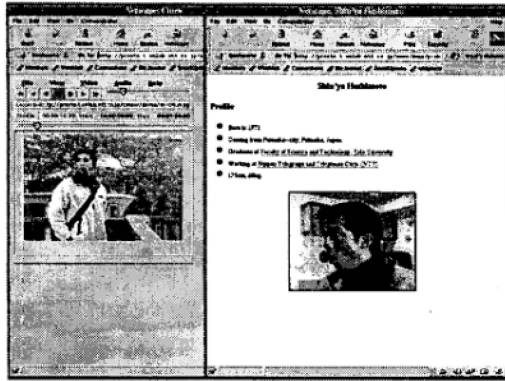


Figure 3: Cmeow Media Player

national standard for media compression. MPEG-4 can associate a URL with an audiovisual (AV) object. Synchronized Multimedia Integration Language (SMIL) [6] is a language used to describe interactive multimedia presentations. The author of a SMIL document can specify the spatial and temporal layout of media objects, and associate hyperlinks with them. These systems make it possible to associate hyperlinks with continuous media. However, they do not provide dynamic control over continuous media. Therefore, they cannot manipulate the structures and behaviors of multimedia documents at runtime.

5 Future Work

Scenario scripts provide a text-based approach for authoring multimedia documents. They are manually created by using text editors. Although scenario scripts are powerful tools for producing dynamic and interactive multimedia documents, marking up hotspots on media nodes and defining the temporal layout of these nodes is a time-consuming task. To make scenarios easier to create, we are designing a GUI-based authoring tool that automatically generates scenario scripts.

We have also proposed hyperlinks for audio data [1]. When audio data in which hyperlinks are embedded are played, icons representing these hyperlinks are briefly displayed. Users can click on these icons to activate the hyperlinks. Audio hyperlinks will be integrated in the next version of the Cmeow media player.

6 Conclusion

Cmeow provides a powerful framework for integrating continuous media with the WWW. Event-driven hyperlinks and script-based scenario descriptions enable flexible and dynamic control over multimedia documents. The Cmeow media player is implemented as a Java applet so that it can be incorporated into the WWW environment.

Acknowledgments

We are grateful to Shigeki Yamada and Rikio Onai for their helpful suggestions and observations.

References

- [1] Shigemi Aoyagi, Toshihiro Takada, Koji Sato, Toshio Hirotsu, and Toshiharu Sugawara. Cmeow/A: An Access Method for Audio Data with Link Information. In *Proceedings of the 6th International Conference of Distributed Multimedia Systems*, July 1999.
- [2] Hans Peter Brøndmo and Glorianna Davenport. Creating and Viewing the Elastic Charles — A Hypermedia Journal. In *Hypertext: State of the Art*, chapter 5, pages 43–51. Intellect Limited, 1990.
- [3] Zhigang Chen, See-Mong, Dong Xie, Aamod Sane, Yongcheng Li, and Roy H. Campbell. Video and Audio: Organization and Retrieval in the WWW, 1997. <http://choices.cs.uiuc.edu/Papers/New/www5/www5.html>.
- [4] Frank Halasz and Mayer Schwartz. The Dexter Hypertext Reference Model. *Communications of the ACM*, 37(2):30–39, February 1994.
- [5] Lynda Hardman, Dick C. A. Bulterman, and Guido van Rossum. The Amsterdam Hypermedia Model: Adding Time and Context to the Dexter Model. *Communications of the ACM*, 37(2):50–62, February 1994.
- [6] Philipp Hoschka. Synchronized Multimedia Integration Language (SMIL) 1.0 Specification, June 1998. <http://www.w3.org/TR/REC-smil>.
- [7] MPEG Subgroups. Overview of the MPEG-4 Standard. ISO/IEC JTC1/SC29 /WG11 N2725, March 1999.
- [8] Nitin “Nick” Sawhney, David Balcom, and Ian Smith. HyperCafe: Narrative and Aesthetic Properties of Hypervideo. In *Proceedings of Hypertext’96*, pages 1–10, March 1996.
- [9] Sun Microsystems, Inc. *Pnuts: A Script Language for Java*. <http://www.etale.com/pnuts/doc/pnuts.html>.
- [10] Sun Microsystems, Inc., Silicon Graphics Inc., and Intel Corporation. *Java Media Players*, November 1997. <http://java.sun.com/products/java-media/jmf/forDevelopers/playerguide/index.html>.
- [11] Toshihiro Takada, Koji Sato, Shigemi Aoyagi, Toshio Hirotsu, and Toshiharu Sugawara. Cmeow: Integrating Continuous Media with the Web. In *Proceedings of the MTAC’98*, pages 136–140, September 1998.
- [12] Jin Yu and Yuanyuan Xiang. Hypermedia Presentation and Authoring System. In *Proceedings of the 6th International World Wide Web Conference*, pages 153–164, April 1997.