A Temporal Model for Interactive Multimedia Scenarios

Naël Hirzalla, Ben Falchuk, and Ahmed Karmouch
University of Ottawa

Many authoring tools let authors create scenarios, but very few let them create an active multimedia scenario that will not only play itself back, but will change course dynamically, depending on user interactions. Our temporal model provides a new way to represent asynchronous and synchronous temporal events, allowing authors to create scenarios that offer viewers seamless, transparent options.

When we think of documents, we usually think of books. Authoring a book requires writing a linear storyline that describes events that happen in time. Creating a multimedia presentation is considerably more complex. Unlike a book, which is unimedia and linear, multimedia presentations often contain media that must occur simultaneously or in some related way, and the author must specify all these relations. For instance, a book, which consists of words strung together, is most often read from front to back. Multimedia presentations contain many other media types, such as audio and video, that can be rendered in parallel. Further, the presentation may be interactive and subtly different at each runtime. We need new document models and methods of representing temporal relations in multimedia documents.

One of the main issues in temporal models of scenarios is the model's degree of flexibility in expressing different temporal relationships. In this article, we are mainly concerned with the temporal behavior of scenarios, not the other attributes of the document such as its layout, quality, and playback speed. We studied how to represent and model multimedia *scenarios* (fully specified temporal entities involving multiple media) that "play themselves back"—that is, multiple media are rendered automatically before the users' eyes—while letting the user interact with the running presentation, "driving" it in a custom direction. We provide a new representation for asynchronous and synchronous temporal events that lets authors create scenarios offering viewers non-halting, transparent options.

A temporal model for active multimedia

Perhaps the most prevalent temporal model is the timeline,[1] which aligns all *events* (see "Definitions" sidebar) on a single axis representing time. Since the events all appear in the order in which they should be presented, exactly one of the basic point relations—"before" (<), "after" (>), or "simultaneous to" (=)—holds between any pair of events on a single timeline (Figure 1). The timeline model, though simple and graphical, lacks the flexibility to represent relations that are determined interactively, such as at runtime.

For example, assume a graphic (say a mathematical graph) is to be rendered on the screen only until a user action (say a mouse selection) dictates that the next one should be rendered. The start time of the graphic is known at the time of authoring. However, the end time of the graphic depends on the user action and cannot be known until presentation time. Hence a traditional timeline, which requires a total specification of all temporal relations between media objects, cannot represent the scenario. Multimedia authors need a model that accommodates partial specifications or interactive multimedia scenarios.

We expanded the traditional timeline model to include temporal inequalities between events. Our goal was to model active multimedia documents graphically within a timeline.

Background work in active multimedia

Modeling user actions as media objects was an early, effective way of representing active multimedia scenarios. Traditionally, the vertical axis of the timeline includes only media such as text, graphics, audio, and video. Creating a new type of media object called *choice* increases the power of the timeline model. This new object is associated with a data structure that contains several important fields:

- user_action,
- region, and
- destination_scenario_pointer.

1070-986X/95/$4.00 © 1995 IEEE

User_action completely describes what input should be expected from the user, such as "key-press-y" or "left-mouse." *Region* establishes which region of the screen (if applicable) is a part of the action; for instance, the user clicking inside the region "rectangle(100,100,150,180)" initiates some result, such as loading a new file. *Destination_scenario_pointer* names a pointer to some other part of the scenario, or even a different scenario. For instance, the author can specify that if the user clicks on a region of the screen at a certain time, the presentation will "jump" to a new chapter.

We can place this new media object, choice, directly on the traditional timeline. As an example, let's take a presentation scenario that renders a video about the American presidents Clinton, Bush, and Reagan, with a narrative audio track. Text boxes display the name and statistics of each president as he is introduced in the short video clip. On the traditional timeline, the scenario looks like Figure 2a.

Now suppose we wish to create some additional, in-depth timelines for each president that the user can jump to by making some selection during the playback of the video introducing that president—an example of active multimedia. To do this, we must first author each in-depth scenario. Then we add three choice objects (Figure 2b) to the original timeline whose data structures contain the following commands:

- user_action = left-mouse

Definitions

We use the following definitions throughout the article, borrowing the first five from Buchanan and Zellweger.[2]

Events	Points at which authors can synchronize the display of media objects (text, video, and so on). Our work focuses on start and end events, but we can generalize to internal events.
Synchronous events	Events with predictable times of occurrence, that is, whose temporal placement is known in advance.
Asynchronous events	Events with unpredictable times of occurrence and durations, that is, whose time of occurrence cannot be known in advance.
Temporal equality	A synchronization constraint requiring that two events either occur simultaneously or that one precedes the other by a fixed amount of time.
Temporal inequality	A synchronization constraint requiring, for example, that events *A* and *B* occur so that *A* precedes *B* by an unspecified duration, by *at least* some fixed time, or by *at least* some fixed time and *at most* another fixed time.
Hypermedia	Implies store-and-forward techniques where user actions, typically mouse-clicks on hot-spots, cause the system to retrieve a new page of data, which can be image, text, video, and so forth. There are usually no temporal relationships between media.
Passive multimedia	Implies a fully synchronized document that plays itself back in time, synchronizing all media objects together.
Active multimedia	Implies hypermedia-type choices presented during the playback of a multimedia document that allow the user's interaction to drive the playback.

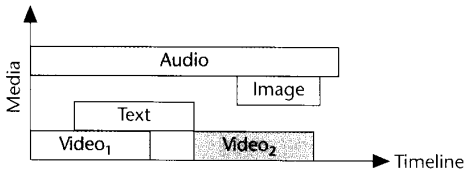
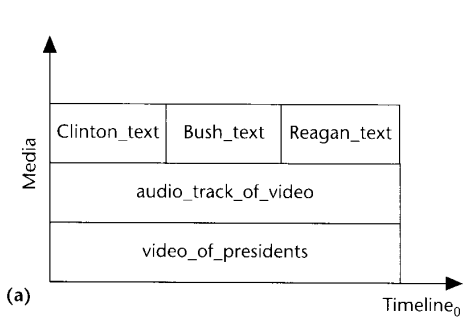


Figure 1. The basic timeline model graphically describes how media within a presentation are arranged over time.



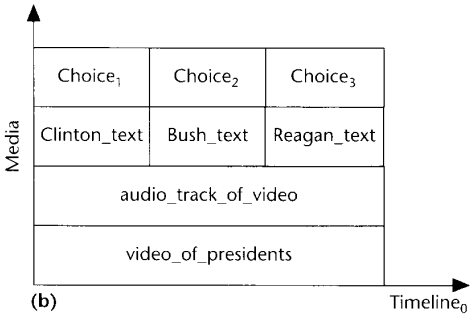


Figure 2. Representing an example presentation with (a) a simple timeline and (b) a timeline that includes choice objects.

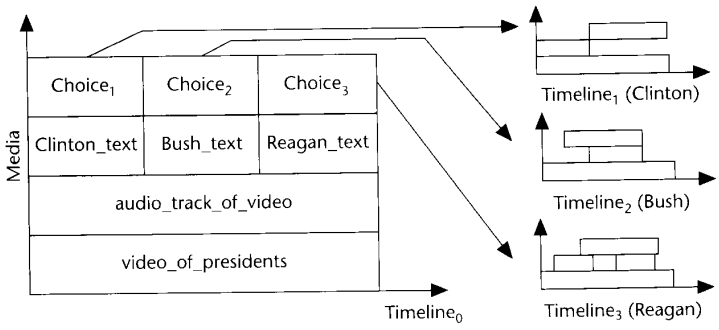


Figure 3. The choice objects from Figure 2b each jump to separate destination scenarios.

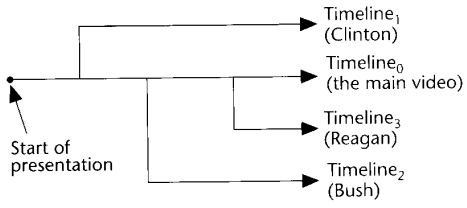


Figure 4. A timeline tree representation of the interactive scenario in Figure 3. Selecting a choice object launches the user into a new timeline.

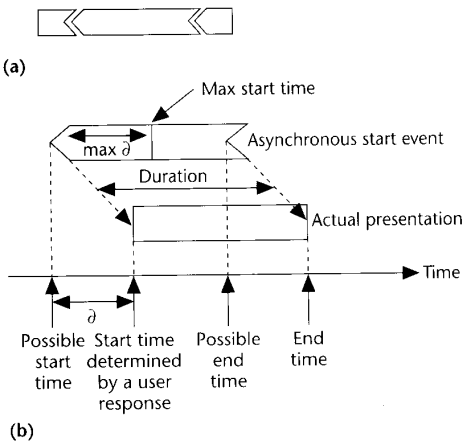


Figure 5. Our temporal model can use (a) its three basic representation units to graphically represent, for example, (b) user response delay.

- region = the appropriate layout location on the screen
- destination_scenario_pointer = the appropriate scenario

In effect, this completes the active multimedia scenario. Each choice data structure refers to some other subscenario, as Figure 3 shows.

Note that choice objects, like other media objects, have a duration: The user has a window of opportunity to make the action that initiates the choice. If the associated action is not made during this time, the user loses the chance to make it. That is, if the user does not make the required mouse selection while the text object Clinton_text is being rendered, she will continue to see the rendering of timeline$_0$ and will receive a new choice, the one associated with President Bush. If the user does make the appropriate choice during the choice object's duration, rendering continues from the destination timeline and the original scenario terminates.

We also developed a way to represent the active multimedia scenario graphically so that authors can visualize their work, using a tree of timelines. The interactive scenario described above can be represented by the timeline tree in Figure 4. Our early work modeled non-halting user interactions and choices—that is, interactions had temporal longevity. As a result we could model transparent choices that did not force the presentation to halt and wait for input, but were integrated seamlessly into the presentation.

The enhanced temporal model

Another issue that arises in active multimedia is how to represent events whose start or end times are not known at authoring time. Our enhanced timeline model splits the traditional rectangle that represents a media object on a timeline into three basic units (Figure 5a). The edges of these units, which represent start or end events, are either straight or bent. Straight lines represent synchronous events and bent lines represent asynchronous events.

We can only establish the start time of an asynchronous event at runtime. This time will be shifted ∂ seconds to the right on the time axis, where ∂ is a nonnegative number representing the user response delay (Figure 5b).

Defining the maximum value for ∂ is necessary in some cases to keep a presentation moving. For example, Figure 5b specifies a limit to the value of ∂, "max start time," so that if the user does not respond to some interactive prompt, a default response starts rendering the media object. The length of the unit from the left sharp point to the right straight line represents the maximum value of ∂.

We could form many different shapes from the three basic units in Figure 5a, but only six of them apply to interactive scenarios. These six shapes and their descriptions appear in Figure 6. To form the shapes, we assumed two things. One, an event that is temporally related to an asynchronous event is itself asynchronous. For example, if the start time of a media with a specific duration is unknown, then its end time is also unknown. Second, the user response delay is a nonnegative value.

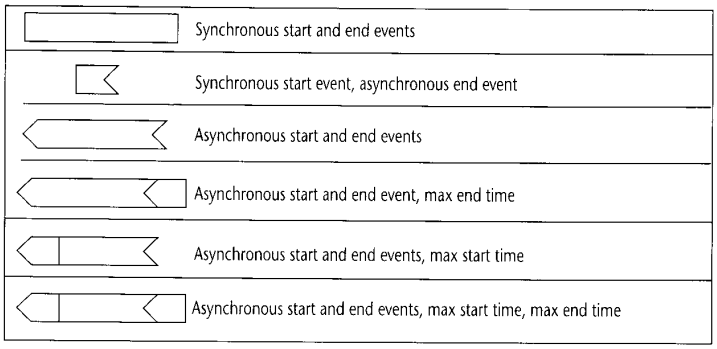


Figure 6. The six types of units in our enhanced temporal model for active multimedia.

Because of our first assumption, we introduced a symbol—Choice$_i$, abbreviated C_i, where i refers to timeline$_i$, $i \geq 0$—to distinguish between temporal equalities and temporal inequalities with other asynchronous events. Events that share temporal equalities (that is, that do not admit terms such as "at least" or "at most") carry the same symbol; otherwise the symbols differ.

Timeline$_i$ is a timeline that branches from timeline$_j$ ($j < i$), because of the unknown user response time. ∂_i then refers to the user delay in responding to the first event with symbol C_i. Associated with C_i is a data structure that determines the user action that initiates the object (for instance, pressing the key "k" or clicking the mouse within a rectangular region of the playback space). L is the actual length of the presentation, while $L_{\min}$ is the minimum duration. X marks the possible end of the presentation. O_j refers to the origin point of timeline$_j$. If time$_i(j)$ is the instant of time when j first appears with respect to timeline$_i$, then we have the following formulas:

$$\text{time}_i(O_j) = \text{time}_i(C_j) + \sum_{n=i}^{j} \partial_n, i \neq j, i \geq 0 \quad (1)$$

$$L_{\min} = \text{time}_0(X) \quad (2)$$

$$L = \text{time}_0(X) + \sum \partial \quad (3)$$

To illustrate further, Figure 7 represents a scenario in which a media object is rendered immediately after a user action, such as a mouse selection, and the object ceases to be rendered after a second user action. Furthermore, the second user action will only be effective five seconds

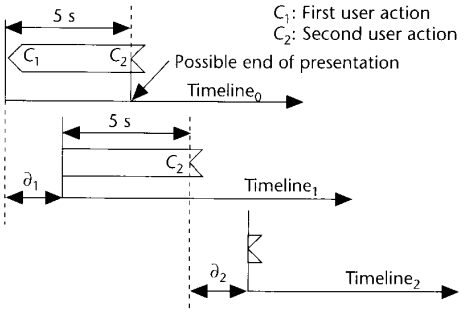


Figure 7. Multiple timelines illustrate how the duration of a presentation (timeline$_0$) is affected by the delay in the first (timeline$_1$) and second (timeline$_2$) user actions.

or more after the first action. The start and end times of the object involved are hence unknown at authoring time. Since the two events have no temporal equality such as "end event occurs exactly five seconds after start event," both must have different symbols, C_1 and C_2.

The start time will become known at runtime when the user performs some action to initiate the object. Since the time it takes the user to respond is not known in advance, a different timeline (timeline$_1$) that clips out the user delay represents the result of this user action. Similarly, the object's end time only becomes known when the second user action takes place. From Figure 7 we can write the following (where $\partial = 0$):

$$\begin{aligned} \text{time}_0(O_1) &= \text{time}_0(C_1) + \partial_1 \\ &= \partial_1 \end{aligned}$$

$$\begin{aligned} \text{time}_0(O_2) &= \text{time}_0(C_2) + \partial_1 + \partial_2 \\ &= 5 + \partial_1 + \partial_2 \end{aligned}$$

$$\begin{aligned} \text{time}_1(O_2) &= \text{time}_1(C_2) + \partial_2 \\ &= 5 + \partial_2 \end{aligned}$$

In this example, the minimum duration of a presentation is 5 seconds, while the actual duration

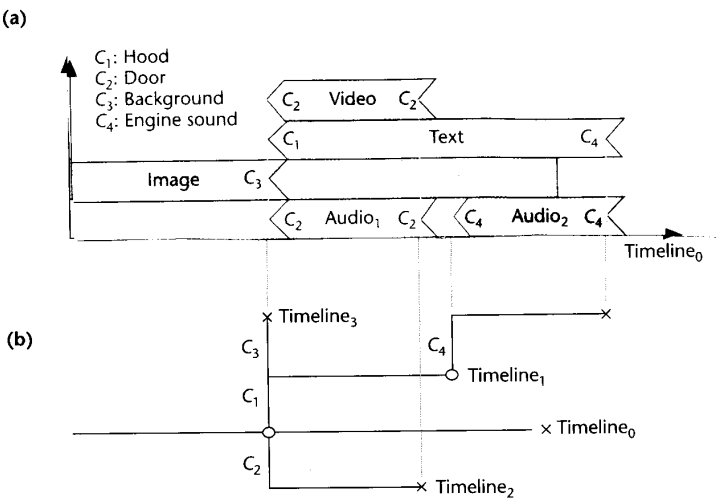


Figure 8. Our timeline-tree model represents interactive scenarios like this car demonstration using (a) an expanded timeline and (b) a tree-like structure that traces all possible timelines.

of the presentation equals 5 seconds plus the user delays ∂_1 and ∂_2.

Note here that we are closer to effectively representing active multimedia. Consider the following example: A user is presented with a graphic of a car. Embedded onto the presentation screen—actually, modeled in the scenario as a combination of a user action (mouse click) and a region on the screen—are three hot spots: the hood, the door, and the background. The user can either choose one of three options by clicking on one of the hot spots or opt not to make a choice at all. Each choice triggers either text explaining the features of the car's engine, a video-audio clip showing and explaining the interior of the car, or the disappearance of the car, respectively. If the user does not respond within a certain time frame, the car image disappears.

If the user chooses the engine and gets the text object, she might then choose to listen to the engine by making another interactive selection from the playback area. The presentation ends after the audio plays. Figure 8a shows how the enhanced temporal model would represent this scenario. Note that, since the end events of "Text" and "$Audio_2$" objects have temporal equality—that is, the events end at the same time—we label both events with one symbol, C_4.

Timeline tree

Our timeline-tree model represents interactive scenarios using a tree-like structure. (The "Related Work" sidebar shows other models for representing interactive senerios.) Figure 8b shows the tree corresponding to the interactive scenario in Figure 8a. The small circles represent branches where user actions may change the course of the scenario. If the

Related Work

We examined six temporal scenario models while formulating our timeline-tree model: the timeline, the petri-net, the CMIFed authorer, Firefly, the interval-based approach, and the relational grammars approach.

We adopted the timeline temporal concept because we found it simple, graphical, and easy to grasp. This makes it attractive as an authoring tool, particularly to nontechnical people who do not want to learn programming concepts to author documents using scripts. However, the model is not flexible enough to include any indeterminism that results from, for example, user interactions.

The petri-net model[5] accommodates temporal inequalities, such as when a delay is unknown at authoring time, thus allowing user interactions. However, unlike the timeline model, the graphical nature of the petri-net model can become complex and difficult to grasp when the document becomes relatively large, as Figure A shows. It illustrates a possible petri-net representation of the scenario shown in Figure 1.

The CMIFed multimedia authorer[6] provides a traditional timeline-type visualization, called the "channel view," shown at the left in Figure B. Authors can synchronize media objects using CMIFed sync-arcs. The hierarchy view, shown on the right in Figure B, offers a novel way of visualizing both the structure of the scenario and the synchronization information using nested boxes. Vertically stacked boxes are executed in sequential order, while horizontally arranged boxes are executed in parallel.

Firefly[1] is a powerful system that supports and models synchronous and asynchronous behaviors. Each media object is modeled by two connected rectangular nodes representing start and end events. Any other event used for synchronization is represented by a circular node placed between the start and end events. Asynchronous events contained in a media item are represented by circular nodes that float above the start event. Finally, temporal equalities between events are represented by labeled edges connecting these events.

In complex interactive documents, we feel that the Firefly model becomes hard to trace, especially when many edges are labeled with time delays. Figure C shows the car example depicted in Figure 9 using the Firefly temporal view. The Firefly representation leaves two questions: Are the four asynchronous events in the car image activated after the

start event, and what is the difference or relationship between "No response" and the "Background" asynchronous event? Unlike our temporal model, the Firefly temporal view cannot represent the temporal relationships between the activation and deactivation of asynchronous events with other events.

Wahl and Rothermel[4] presented a high-level, interval-based temporal model whose powerful operators include both temporal equalities and inequalities between events. As in other complex interactive documents, scenario representations are hard to trace. In addition, although the operators can be applied to asynchronous events (whose start-times are not known in advance but exist in the actual presentation), the operators cannot be applied to those events that might not occur at all in the actual presentation. For instance, Figure 9 shows a slide show example[4] in which every event will definitely occur.

In the relational grammars model,[7] multimedia documents are presented automatically based on parsing and translation. Grammar rules map content to the look and feel of a spatially and temporally laid-out document. While our system does not address spatial layout, this model does not address user interaction with an active document (as opposed to a hypermedia document), as ours does. Also, our system is concerned mainly with interactive scenarios that allow more than one temporal playback, while relational grammars mainly address documents with alternate spatial layouts.

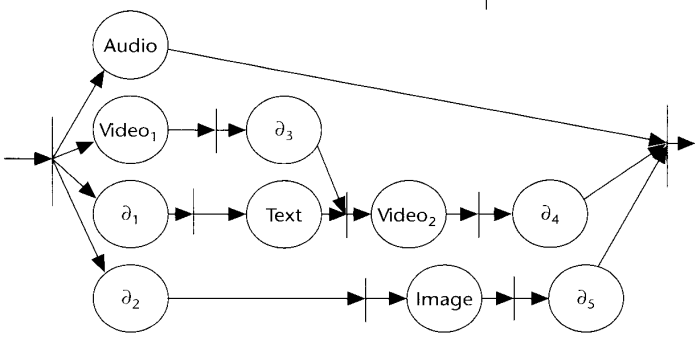


Figure A. How the petri-net model would represent the scenario of Figure 1.

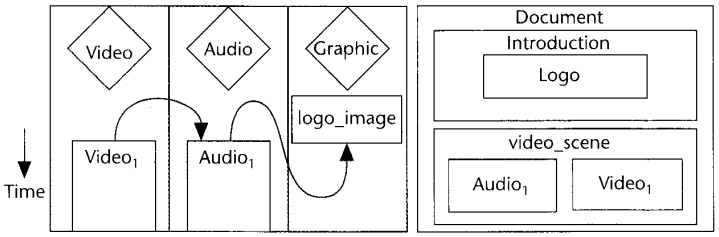


Figure B. CMIFed channel view (left), showing sync-arcs to synchronize the start of the audio and video with the end of the logo, and hierarchy view (right).

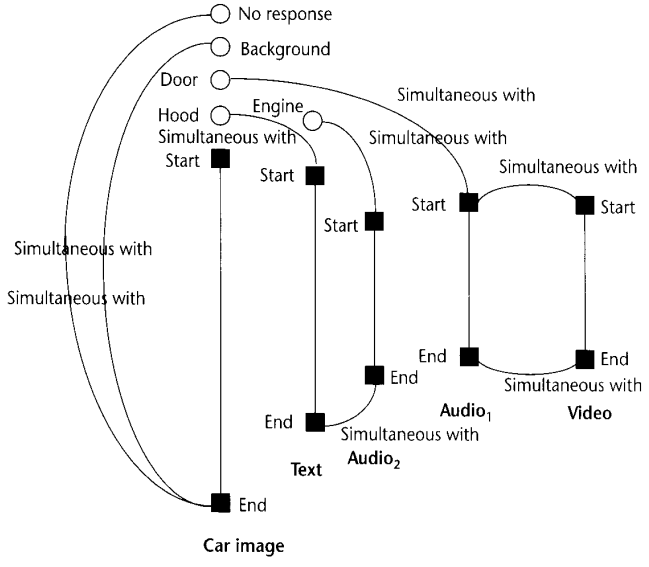


Figure C. Firefly's temporal view of the car example from Figure 9.

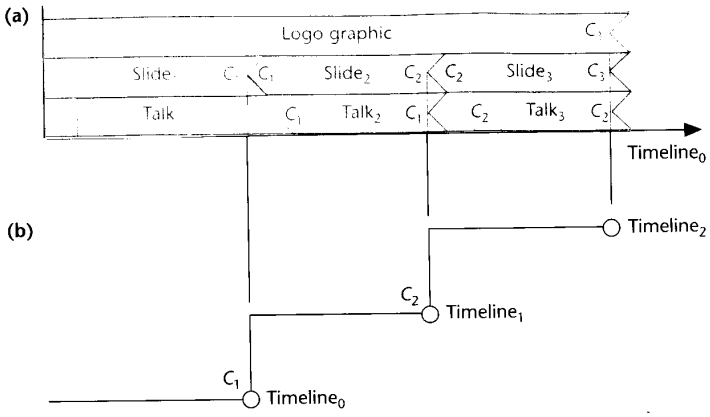


Figure 9. Modeling a slide show representation with our timeline-tree model using (a) an expanded timeline and (b) a treelike structure that traces all possible timelines.

user makes no choices, the current timeline simply plays itself out ($timeline_0$); otherwise, users traverse the timeline tree, viewing custom presentations ($timeline_1$ through $timeline_4$) determined by their choices. In the figure, each X represents possible ending points of the scenario.

Note that at most one choice, C_i, can be selected at a time. Consequently, the presentation flow will branch to $timeline_i$. The *timepath* outlines the set of timelines the actual presentation follows. For example, if the user in the above scenario selected the hood of the car (C_1), the presentation flow would branch from $timeline_0$ to $timeline_1$. Then, if she chose to listen to the sound of the engine (C_4), the presentation-flow would branch to $timeline_4$ until the presentation finishes. This timepath would be {0, 1, 4, X}. All possible timepath sets start with $timeline_0$ and end with an X, the end of the presentation. Therefore, we must change Equations 1 to 3 to refer to a certain timepath set.

$$\text{time}_i(O_j) = \text{time}_i(C_j) + \sum_{n=i,\, n \in \text{timepath}}^{j} \partial_n \; ; i, j \in \text{timepath}$$

$$L_{\min} = \text{time}_0(X); \quad X \hat{\text{I}} \text{ timepath}$$
$$L = \text{time}_0(X) + \sum_{n \in \text{timepath}} \partial_n ; X \in \text{timepath}$$

We can calculate the shortest possible timepath sets by examining the timeline tree:

$$L_{\min} = \text{time}_0(\text{closest}(X))$$

In the tree model, the circles represent the times that asynchronous events corresponding to the symbols shown at the circle become activated. Those events are deactivated only when the presentation flow branches to another timeline.

Example

The following example (also used by Wahl and Rothermel,[4] but without the logo graphic) illustrates the use of different timelines. Figure 9a shows how an author can create a scenario with the following properties: Any slide starts being rendered three seconds before its corresponding audio to allow a silent period for viewers to form a first impression of the slide. After a talk is finished, the current slide continues to be rendered until a viewer responds with some input. A logo graphic is rendered for the whole duration of the scenario. As Figure 9b shows, the start event of $Slide_2$ has a temporal equality with the end event of $Slide_1$, as does $Talk_2$; thus, they all utilize the same symbol, C_1. The end event of $Slide_2$, however, has a temporal inequality with its own start event (since it ends interactively), as do $Slide_1$ and $Slide_3$; thus, each needs a different symbol.

We can visualize this scenario using the timeline-tree. Figure 10 shows the scenario from $timeline_1$'s point of view, after C_1 has been invoked. Thus the timeline-tree does not show $Slide_1$ or $Talk_1$, and it depicts the logo graphic as being already in progress.

Furthermore, complex documents that might involve many choices and many user interaction points demand the ability to select a subset of the scenario for inspecting, for example, a specific timeline and a specific set of choices. Figure 11 shows only media objects initiated by the action C_1.

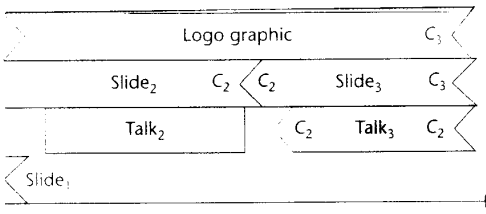


Figure 10. Viewing a specific timeline, $timeline_1$.

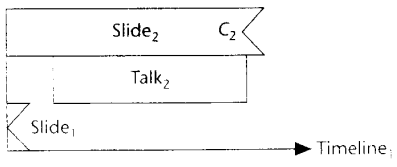


Figure 11. Viewing a specific timeline and specific choice list, $timeline_1$ AND $\{C_1\}$, the results of the query, "What media objects are initiated by action C_1?"

Conclusions

We feel that modeling scenarios using our timeline approach is superior to using the traditional timeline, since our model allows fully interactive, transparent user interaction with the scenario. Thus a much wider and more interesting range of scenarios becomes possible, such as interactive books, encyclopedias, and movies, with interaction modeled as a part of the scenario.

Our model for defining interactive documents can be used to create documents in many different settings. In our opinion, using the model to create interactive instructional course material is probably the best application. Not only can an author create fully synchronized multimedia courses, but users will not be limited to simply watching the presentation. They can interact with it, browse, and explore. Other useful applications include creating interactive news articles for multimedia news-on-demand or creating interactive entertainment for video-on-demand and interactive cinema. **MM**

Acknowledgments

We would like to acknowledge the input, advice, and proofing of Omar Megzari, a researcher and colleague of ours at the Multimedia Information Research Laboratory, and also the Ottawa Carleton Research Institute and the Telecommunications Research Institute of Ontario for their support.

References

1. G. Blakowski, J. Huebel, and U. Langrehr, "Tools for Specifying and Executing Synchronized Multimedia Presentations," *Proc. 2nd Int'l Workshop on Network and Operating System Support for Digital Audio and Video*, 1991, pp. 271-279.
2. M. Buchanan and P. Zellweger, "Specifying Temporal Behavior In Hypermedia Documents" *Proc. ACM Conf. on Hypertext*, ACM Press, New York, 1992, pp. 262-271.
3. L. Hardman, D. Bulterman, and G.V. Rossum, "The Amsterdam Hypermedia Model," *Comm. ACM*, Vol. 37, No. 2, Feb. 1994, pp. 50-62.
4. T. Wahl and K. Rothermel, "Representing Time in Multimedia Systems," *Proc. Int'l Conf. on Multimedia Computing and Systems*, CS Press, Los Alamitos, Calif., 1994, pp. 538-543.
5. T.D.C. Little and A. Ghafoor, "Synchronization and Storage Models for Multimedia Objects," *IEEE J. Selected Areas Comm.*, Vol. 8, No. 3, Mar. 1990, pp. 413-427.
6. R. Rossum et al., "CMIFed: A Presentation Environment for Portable Hypermedia Documents," *Proc. ACM Multimedia 93*, ACM Press, New York, 1993, pp. 183-188.
7. L. Weitzman and K. Wittenburg, "Automatic Presentation of Multimedia Documents Using Relational Grammars," *Proc. ACM Multimedia 94*, ACM Press, New York, 1994, pp. 443-451.

Naël Hirzalla is a research assistant in the Multimedia Information Research Laboratory (MIRLab). He is also a PhD candidate in the Department of Electrical Engineering at the University of Ottawa, Canada. He received his B.Eng. from the Jordanian University of Science and Technology, Jordan, and his MASc from the University of Ottawa, Canada. His research interests include computer networks and multimedia databases.

Ben Falchuk is an MS candidate in the department of computer science at Carleton University. He received his B.Mathematics from the University of Waterloo. His current research interests are multimedia, graphical user interfaces, and applications of multimedia technology, including news of the future.

Ahmed Karmouch is a professor of electrical engineering at the University of Ottawa. He holds an Ottawa Carleton Research Institute/Natural Sciences and Engineering Research Council (OCRI/NSERC) Industrial Research Chair in real-time multimedia distributed database systems. He is also director of the Ottawa Carleton Institute of Electrical Engineering and director of the Multimedia Information Research Laboratory. Karmouch earned an MS and a PhD in computer science from the University of Paul Sabatier, Toulouse, France. His research interests include distributed multimedia databases, cooperative systems, interactive multimedia information systems, multimedia synchronization and communications, distance learning, and multimedia news applications.

Contact Karmouch at the Department of Electrical Engineering, University of Ottawa, Ontario, K1N 6N5, Canada, e-mail karmouch@elg.uottawa.ca.