The Link *vs.* the Event: Activating and Deactivating Elements in Time-Based Hypermedia

Lynda Hardman
CWI, Amsterdam, The Netherlands
(e-mail: Lynda.Hardman@cwi.nl)

Patrick Schmitz
Microsoft BARC, San Francisco, CA, USA
(e-mail: pschmitz@microsoft.com)

Jacco van Ossenbruggen
CWI, Amsterdam, The Netherlands
(e-mail: Jacco.van.Ossenbruggen@cwi.nl)

Warner ten Kate
Philips Research, Eindhoven, The Netherlands
(e-mail: warner.ten.kate@philips.com)

Lloyd Rutledge
CWI, Amsterdam, The Netherlands
(e-mail: Lloyd.Rutledge@cwi.nl)

Abstract

Activation and deactivation of media items plays a fundamental role in the playing of multimedia and time-based hypermedia presentations. Activation and deactivation information thus has to be captured in an underlying document format. In this paper we show that a number of aspects of activation and deactivation information can be captured using both link structures and events in time-based hypermedia. In particular, we discuss how deactivation and activation can be specified, how the activations and deactivations can be initiated and potential (synchronization) relationships between the elements involved.

We first introduce the notions of time-based scheduling and event-based scheduling and then present a short summary of linking. We discuss the similarities between event-based scheduling and linking. We describe a number of aspects of activation and deactivation that can be specified within a document. We then discuss how activation and deactivation information can be recorded in link structures and events.

1. Introduction

An informal notion of *hypertext* is that a piece of information is linked to another piece of information and, when a reader clicks on the first item, a second item comes into view. In contrast, an informal notion of *multimedia* is that a number of items play on the screen, and the items playing change with time. These views of hypertext and multimedia class the interactions into two different and unconnected worlds, thus restricting the potential for specifying

interaction within presentations. We argue that unifying these views simplifies the perceived complexity of models of linking and event-processing in time-based hypermedia, allows extra functionality to be expressed and facilitates the authoring task.

We are concerned primarily with distinctions and approaches for authors and system developers. For the end-user there should be little perceived difference, other than they are not always expected to click with a mouse to see new information, but that other interaction means are possible, e.g. moving a mouse over an object reveals extra information. When creating tools for building and playing information-rich presentations, developers have been historically guided by a hypertext approach (e.g. creating sets of HTML pages) or a multimedia approach (e.g. the large multimedia CD-ROM market). Our aim is to show that these "traditional" views of hypertext and multimedia can fruitfully be merged and allow the creation of richer and more flexible information environments incorporating various ways of giving access to both static and time-varying information.

In the paper we make the distinctions of time-based and event-based scheduling and describe the notion of link activation. We show that event-based scheduling and link activation are extremely similar processes which emphasize different aspects of encapsulating activation and deactivation information. This not only sheds light on the commonalities of the structures but also allows each to be extended to be able to express further useful interaction phenomena. For example, links can be extended with other methods of activation, and transition information can be added to event-based activation.

A number of the issues discussed in the paper can be illustrated using SMIL 2.0 (1) (Synchronized Multimedia Integration Language). SMIL 2.0 is a W3C language specification for describing declarative web-based hypermedia documents. The specification contains descriptions of a number of language profiles, one of which is the SMIL 2.0 language. All references in this paper are to the SMIL 2.0 language profile, unless stated explicitly otherwise.

The paper is structured as follows. In the following section we define time-based and event-based scheduling and in Section 3. we describe hypertext linking. In Section 4. we specify the information that can be expressed in relation to activation and deactivation of elements within a presentation. In Section 5. we discuss how these types of information are recorded in link structures and events and include detailed examples from SMIL 2.0. The paper concludes with a summary of observations about the advantages and disadvantages of recording activation and deactivation in link and event structures and suggests directions for future work.

2. Scheduling

In this section we present briefly the notions of time-based scheduling and event-based scheduling.

2.1 Time-based Scheduling

Time-based scheduling specifies predictable temporal relationships among elements included in a presentation. In other words, the temporal relations are *resolved* and can be calculated before the presentation is played at runtime (2), (3). A presentation consisting of only time-based scheduling has a single timeline with which the different elements in the presentation are synchronized. A user can navigate back and forth along the timeline, in the same way that a user

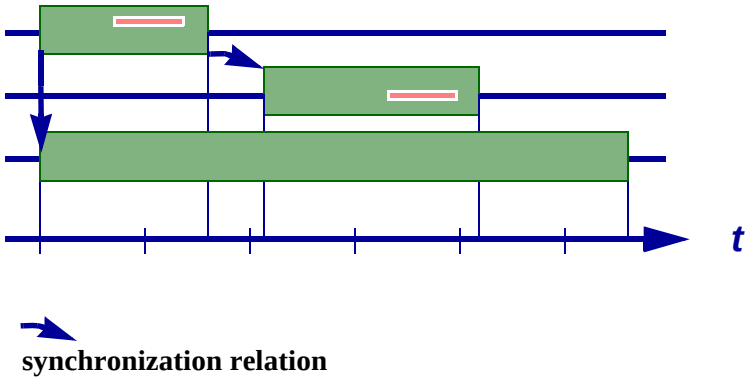

Figure 1. Linear time-based presentation

can forward and reverse a videotape. This type of presentation is not affected by the activation of links, other than to stop the presentation before it has reached its scheduled end time. This style of scheduling can be specified in constraint-based systems, including Firefly (4) and Madeus (5), and timeline-based systems such as Macromedia Director, see chapter 4 of (6).

In order to calculate the scheduling of a presentation, the temporal information within the underlying document can be extracted to calculate the start and end times of media items with respect to a single time-axis. This results in a temporally linear presentation, for example, see Figure 1.

Time-based scheduling is useful for synchronizing multiple elements within a single linear presentation, but it does not provide for non-predictable changes, for instance caused by user interactions or other external entities, that lead to non-linear presentation or adaptation of presentation style. These can be modelled through event-based scheduling and link traversal.

2.2 Event-Based Scheduling

Time-based scheduling takes no account of elements that start up at a time that is only known during the playing of the presentation, i.e. at runtime. We term these elements as having *unresolved* timing. For example, mousing over an area of a picture causes it to highlight, or clicking on an object removes it from the display. This allows dependency graphs to be constructed, where media items are displayed or removed depending on the current state of the presentation, on the interaction with it and on external triggers. Often this type of behaviour is encapsulated in scripts of procedural code which is driven by the variability in potential dependencies. By starting at the graph structure we aim to capture this type of behaviour in a declarative model of event-based activation (7).

Restricting behaviour to a declarative model enables the independent optimization of the implementations along the media production chain, such as authoring tools and players. Each of

them can strictly interpret the declared semantics, and thus can optimize the preparation and actuation of the semantics, with confidence that there will be no side effects. Procedural code cannot be interpreted in this sense, which means that from the perspective of the implementation, once the script is running it can do anything. This means that the implementation cannot make any assumptions about what happens when the script begins to run. Even in cases where only a small amount of scripting seems necessary, e.g. to describe the value of one attribute as a function of another, as soon as the script runs there is no control of the result. Declarative event-binding is an approach which can alleviate this problem.

Elements with *unresolved* timing have an undefined begin or end time. The element is specified somewhere within the document, but the begin or end time is determined by some activation at runtime. Activation may be event-based (such as by a user-input event), link-based (with a link targeted at the element), or externally based (e.g. in the case of SMIL by a DOM call such as the beginElement() or beginElementAt() methods). The temporal schedule is thus resolved only at runtime. The event-activation support provides a means of associating an event with the begin or end time of an element. When the event is raised (e.g. when the user clicks on something), the begin time of the element is resolved at runtime.

Types of events include user interaction, e.g. moving the mouse over an object, clicking on an object, or similar behaviour using the keyboard. Other sources of unpredictable events include the pre-loading of media items, streamed events (e.g. alerts such as advertisements, stock updates) and system events (e.g. “bandwidth available/notification”, “ connection established”, “pre-loaded”, “clocks synchronized”).

Event-based scheduling constructs a schedule during runtime, based on the sequence of events that occur during runtime. For a time-based presentation each instantiation of the schedule during runtime is identical. For an event-based presentation each instantiation is potentially different. For example, a presentation consists of three images. When the user moves the mouse over an image, a corresponding piece of text appears. The schedule differs for each playing of the presentation since the timing is fully determined by the user’s actions.

2.3 Combining Time-Based and Event-Based Scheduling

While time-based scheduling tends to be good for storytelling, it has limited support for user-interaction. Event-based systems, on the other hand, model multimedia as a graph of event bindings. Event-based systems provide flexible support for user-interaction, but generally have poor scheduling facilities and are best applied to highly interactive multimedia.

A powerful approach is to combine both types of scheduling in a single model, as is the case with SMIL 2.0. Begin and end times of media items can be pre-specified, using time-based scheduling, or specified to start when some event occurs, using event-based scheduling. In both cases, when an element is started up its timing becomes attached to the runtime timeline.

While event-based scheduling appears to be primarily a scheduling mechanism, it is closely related to the activation specified using links.

3. Linking

3.1 Linking in hypertext

A link in hypertext specifies a relationship between two or more pieces of information. Most often this relationship is accessible for traversal to the user through some sort of user interaction—most commonly by clicking on the source anchor of the link. A widely cited model for hypertext, including a formalization of the model, is the Dexter hypertext reference model (8). This was developed by a number of hypertext system designers to capture the essence of hypertext, and thus linking. A Dexter link consists of a number of link-end specifiers, each of which has an associated direction (TO, FROM, BIDIRECT, NONE) and a reference to the element being linked. A Dexter link can thus consist of a pair of link-end specifiers, one TO and one FROM, or a larger collection of specifiers with varying values for the direction. Probably the most commonly encountered hypertext link is the `a` element in (X)HTML. This defines a source element (equivalent to a Dexter link-end specifier with direction FROM) as enclosed within the `a` element and a destination element (equivalent to a Dexter link-end specifier with direction TO) as specified by the `href` attribute

More recently, the XLink (9) specification has been developed by W3C. This describes a number of linking modelling primitives which can be used in the definition of linking elements in XML languages.

3.2 Linking in multimedia

Explicit linking constructs in multimedia are much less common. Instead, since multimedia has the notion of a timeline, simple scripts allow the traversal of the presentation timeline. This provides functionality similar to a single source, single destination link mechanism. For example, Figure 2 shows a multimedia presentation with a link from an initially playing element to the middle of a second element. Figure 3 shows the same presentation where a user has activated the link, at time t_{act}, thus fast-forwarding a part of the presentation. The user sees the presentation as scheduled up to the point that the link is selected, and then (after the player has skipped over part of the presentation) from the point of the link's destination. This is possible because the schedule of a linear multimedia presentation can be calculated beforehand.

3.3 Link in time-based hypermedia

A link in time-based hypermedia is more complex because of the interaction between the running (source) presentation and the destination of the link. Should these two become part of the same synchronized presentation, or should there be two independently running presentations? In hypertext, the source of a link can be removed without any further consequences. Where the source element is part of a synchronized whole it is unclear how much of the presentation can or should be removed. Figure 4 shows a link in time-based hypermedia at runtime. It illustrates a juke-box application, with, for example, a brightly coloured background with a list of music and video clips that can be selected. When the user selects a track, the currently playing music and video is faded out and the selected one faded in. The list and the background continue playing undisturbed.

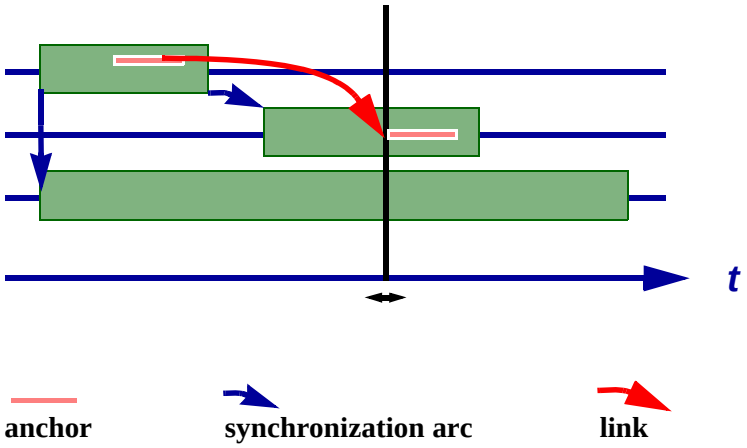

Figure 2. Link in a multimedia presentation

In the case that the destination of the link is not active, then information is also needed on whether a single element should be played, or whether the specified destination is part of a larger presentation. Specifying the context for the source and destination of the link is a solution to this (10). Other related questions are whether the destination should be displayed in the same or a different window, and whether the original presentation should continue to play, or pause.

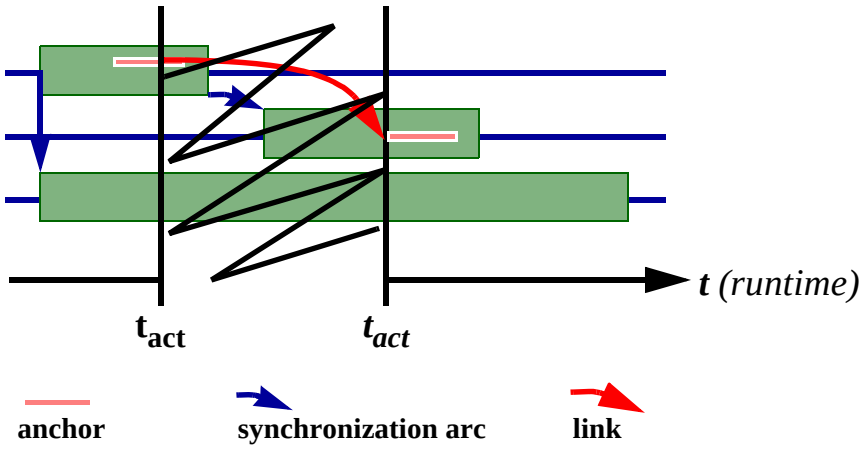

Figure 3. Following a link at runtime in a multimedia presentation

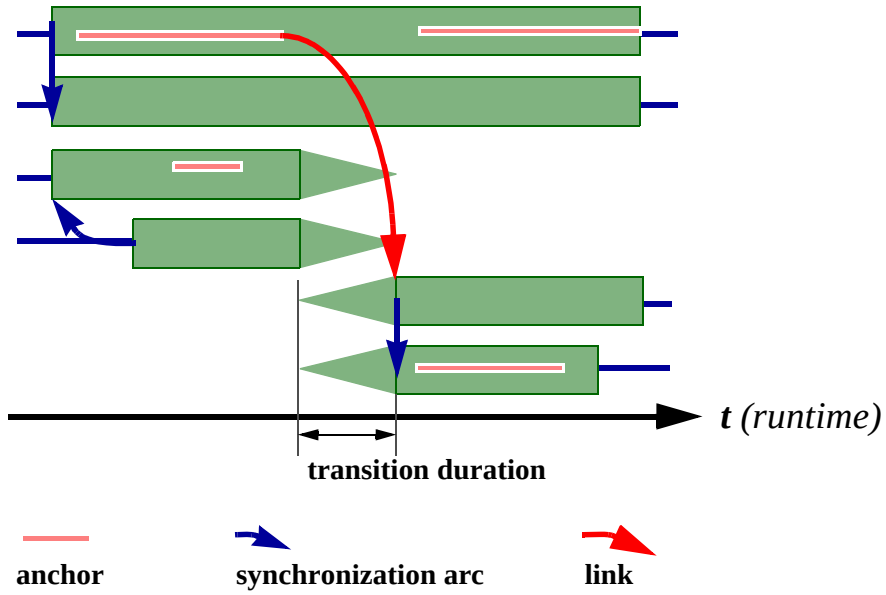

Figure 4. Following a link at runtime in a time-based hypermedia presentation

In the rest of the paper we discuss the types of information that can be specified when navigating through time-based hypermedia, and how this information can be associated with a link structure and/or an event.

4. Deactivation and activation information: Requirements

When a presentation is playing it has a number of active media items—the pieces of media perceived by the reader. As discussed in the previous sections, these can change according to a resolved time-based schedule or according to interactions described within the underlying document. Independent of the way the information is stored in the document, there are basically three aspects to consider:

- what will happen — which media items will stop playing or pause and which will start playing: section 4.1 Deactivation and Activation of Elements;
- when will this happen — what are the initiation conditions and how are conflicts resolved: section 4.2 Activation Initiation;
- how will this happen — what types of transitions are applied and where will the new items be displayed: section 4.3 Relationship between source and destination.

We discuss aspects of each of these in turn, giving definitions and describing behaviour that can usefully be captured in a document.

4.1 Deactivation and Activation of Elements

We define the *source elements* to be the set of active elements that are discontinued or paused as part of the activation and deactivation specifications. There may be other active media items in

the presentation which are unaffected by the activation specifications—these will continue to play according to their time-based schedule.

Similarly, we define the *destination elements* to be the elements that are started up as part of the activation and deactivation specifications. There is not necessarily a predefined relationship between the source and destination elements, so that, potentially, (a number of) the destination elements could already be playing. In this case there are two options:

(1) The destination elements that are already playing continue to play as if nothing had happened and any other destination elements not already playing are started. While this seems to be an obvious default, it destroys any potentially desired synchronization relations among the destination elements. For example, if the destination consists of a slide show timed to an accompanying sound track and the sound track was already playing, then the correspondence between the sound track and the timing of the slides will be lost.

(2) The already playing items are restarted, in which case the synchronization relations within the destination elements are preserved. This may also have disadvantages, where, e.g., unsynchronized background music is unnecessarily interrupted each time the reader jumps back to the first slide.

Both of these options are valid, and should be specifiable as part of the activation specification.

Seeking

When a destination element is to be activated, it needs to be located—either somewhere in the current presentation, or in another presentation. In a text environment this is straightforward, since either the current document is scrolled until the destination element is reached, or a new document is opened. In the latter case the document is opened in the same window or in a new window and the destination element is scrolled to. In time-based presentations the destination element occurs at some time in the presentation, as well as at a position (spatially or in the text-flow). Starting up the presentation and displaying it at a time other than its start time is termed as *seeking*.

4.2 Activation Initiation

For the activation and deactivation specifications to take effect, there needs to be some trigger to initiate the process. This consists of an activation condition and an activation source.

The *activation condition* is the runtime event that triggers the rest of the process. The events can be user interaction events, e.g. a mouse-click carried out by the reader, or media stream events, e.g. an event from a streamed video. For example, a football match is being broadcast as a live stream, and when a goal is scored an event marker is included in the video stream, triggering the display of information about the player who scored the goal.

The *activation source* is the element to which the activation condition applies. The most familiar use case is the element a user clicks on in a Web browser. The activation source may describe a complete media item, or may specify some part, for example a Dexter (8) anchor.

4.3 Relationship between source and destination

Synchronization

When discussing the notions of activating and deactivating elements there are two questions related to the issue of synchronization: firstly, how do the source elements (i.e. those which will discontinue or pause) relate to the synchronization of the presentation as a whole, and secondly, how do the elements in the destination relate to each other and to the active elements (i.e. those that continue to play unaffected)?

In the source case, there are a number of elements that are playing in the presentation. Some of these should be kept synchronized with one another, e.g. an audio track is lip-synchronized with a video track (*hard synchronization*). For others the synchronization is less important, e.g. some background music is accompanying a slide show (*soft synchronization*). When stopping the source elements the author would not want to stop the audio but leave the video running, but, on following a link from the slide-show, may want to fade out the background music only after removing the slides. The destination elements (those which are started up) may be intended to be either hard-synchronized or soft-synchronized with the playing presentation. The author needs control for all these cases.

Transitions

When stopping the playing of a media item and starting up another, a smoother presentation is created by allowing the media item to merge into the new one, using one of a large number of effects, e.g. see the SMIL transitions (11) and the SMPTE edit decision lists (12). This can often be specified at the beginning or end of elements within a time-based schedule, and is also useful when items are stopped before their scheduled end-time. Ideally, full control should be possible over the “out” transitions on the stopping source elements and the “in” transitions on the starting destination elements. This allows, for example, traversal among elements within a scene to perform cross-fade transitions, thus maintaining a sense of continuity. In contrast, when coming from another scene, the difference could be highlighted through having the source item fade to black (out-transition) and the destination element use an iris open effect (in-transition). While we focus on visual transitions, audio fades and cross-fades are also important transition effects.

Interrupt control

If a destination element is to be played in the same window as an item already playing then a number of options are possible

- it can replace the source element immediately;
- wait until the source has finished playing;
- or indeed not be played at all. (This last case is perhaps less appropriate for a user initiated activation, since the user really wants to see the destination.)

For example, in SMIL 2.0, the `excl` element enables authors to specify explicitly the way destination elements interrupt the playing elements. Some elements are allowed to interrupt a playing element, whereas others have to wait until the playing element has completed. This allows a television programming model to be specified, where a “program” video is paused when a “commercial” insert begins and resumes playing when the commercial ends. A commercial that tries to interrupt a playing commercial has to wait until the playing commercial finishes.

Destination display environment

The items in the destination take up resources that may potentially be in use by other playing parts of the presentation, e.g. windows for visual items and audio channels for audio. A destination display environment is needed to allow the specification of which resources the destination elements should use—should they re-use the resources of the source elements, or should new ones be created.

5. Deactivation and activation information: Link *vs.* the Event

Given the above aspects of activation and deactivation within a time-based hypermedia presentation we discuss each aspect in terms of link and event constructs. In particular, we look at where aspects of an activation/deactivation process have been omitted from the link or event construct. We also discuss incorporating combinations of these either within a single, complex and potentially unwieldy link structure or as multiple, light-weight but potentially difficult-to-maintain event descriptions. Each sub-section discusses the Amsterdam Hypermedia Model (AHM), the SMIL 2.0 `a` element, Xlink and the SMIL 2.0 event mechanism, in this order. The SMIL 2.0 constructs are summarized in Table 1.

5.1 Deactivation and activation of elements

Source and destination elements can be explicitly specified in the AHM (6). The source elements (those paused or stopped) are the link ends which are specified as stopping on traversing the link. Explicit attributes are provided for specifying whether an active link-end should stop playing, pause or continue. In Dexter (8) and in AHM terms the activated elements are given by the TO link-end specifiers. The AHM also allows a choice of whether the activated elements start playing immediately or are paused.

In SMIL 2.0, deactivation and activation mechanisms are present throughout the language: time-based scheduling, using the `par` and `seq` elements; event-based scheduling, using the `par` and `excl` elements and events; and links, using the `a` element also in conjunction with the `excl` element. The source elements (those paused or discontinued), see Table 1, cannot be specified explicitly, but can be specified through the document structure and the `source/destinationPlaystate` attributes. Elements that are deactivated depend partly on the composition and linking structure of the document, and partly on the applicable interrupt rules. For example, with time-based scheduling, the `seq` element deactivates a child before activating the next child. With the `a` element, deactivation of the source depends on the setting of the `sourcePlaystate` attribute. The `show` attribute also contributes to specifying whether the presentation containing the link should continue to play or be replaced. On replacement, the playing presentation is assumed to be paused, rather than deactivated, so that when the destination elements have finished playing the original presentation will continue from where it left off.

The destination elements (those started up), see Table 1, are those referred to by the value of the `href` attribute. The link destination is guaranteed to be played, so that if the parent of the referred to element is not already active then it is activated.

SMIL links do not allow the creation of multi-ended links with activate/deactivate conditions on them. Instead, the `excl` element, in combination with event-based timing, is specified. This

provides a short-cut mechanism for specifying the source elements. Basically, if any child of an `excl` starts playing (independently of how it is activated), then any playing siblings are stopped or paused. This is similar to the behaviour of the `choice` element in CMIFed (13).

Deactivation and activation behaviour can to some extent be captured in XLink (9). A link may have a `show` attribute with value `replace` which specifies that a destination element (identified using a `from` attribute) should be displayed instead of the source element (identified using a `to` attribute), in other words the source element is deactivated.

An event mechanism deactivates elements by having each element to be deactivated include a reference to the occurrence of the event. For example, a number of elements within the document may have an end condition relating to the event. When the event occurs, those playing will be stopped. Similarly for activating elements, each element to be activated includes a reference to the event. Using events for capturing link behaviour is explicitly mentioned in (14).

Table 1: Link and event structures in SMIL 2.0

	SMIL link: the `a` element	SMIL event: `begin/end` attributes
source elements	The `sourcePlaystate={play, pause, stop}` attribute specifies whether the presentation containing the link should continue to play, pause or stop. The `show={new, replace}` attribute specifies whether or not the playing presentation continues or is paused.	The element(s) in which the `end` attribute is specified. Multiple elements may share the same `event-value` of the `end` attribute.
destination elements	The value of the `href` attribute specifies the (possibly composite) destination element. The `destinationPlaystate={play, pause}` attribute specifies whether the destination element is started up as playing or paused.	The element(s) in which the `begin` attribute is specified. A check is made to ensure that the surrounding presentation is already active. If this is not the case, then the activation does not take place. Multiple elements may share the same `event-value` of the `begin` attribute.
activation condition	The `actuate={onRequest, onLoad}` attribute specifies when the destination element is presented, either on loading the presentation or on some other trigger. One possible trigger is an assigned keystroke, using the `accesskey` attribute.	The event value referred to in the value of the `begin` and `end` attributes. There can be multiple `begin`s within a single element. Multiple elements can share the same `event-value` of the `begin` attribute.
activation source	The `a` element itself is the activation source. Parts of the media item can be specified using `area` and `fragment` attributes.	The element referred to in the value of the `begin` and `end` attributes.
synchronization between source and destination	None can be specified.	None can be specified.

Table 1: Link and event structures in SMIL 2.0

	SMIL link: the `a` element	SMIL event: `begin/end` attributes
transition	Transitions cannot be specified.	The out transition of the `eventbase-element` and the in transition on the element in which the `begin` attribute is specified. Note that while these are specified for the element the use of the out transition during event-based activation is not specified in SMIL 2.0.
interrupt control	Interrupt control cannot be specified as part of a link, but can be specified using a `priorityClass` element as a parent of the destination element in conjunction with an `excl` element.	Interrupt control uses the same mechanism for both a link and an event.
destination display environment	The `target=displayID` attribute allows the destination display environment to be specified. The `displayID` can refer to a SMIL region, an HTML frame or a new window.	The `region=regionID` attribute allows the destination display environment to be specified. In contrast to the `target` attribute, it can only reference a SMIL region.

Seeking

In the AHM the notion of seeking is implicitly assumed: each link-end specifier has not only a destination element (the destination anchor) but also a context—a temporal structure specifying from where the destination presentation should start playing. When the link is activated, any destination presentations are activated and played from the state specified by the destination context.

A SMIL 2.0 link assumes that the destination element will start up and that the player will seek to the appropriate element in the presentation (taking into account the interrupt control rules), activating the appropriate branch of the presentation structure.

XLink, being presentation-independent, does not encode any notion of time, nor any notion of scrolling or seeking to the destination.

The SMIL event is dependent on the runtime state of the presentation. If the element listening to the event cannot be scheduled, e.g. its parent is part of a sequence which hasn't yet been reached, then the presentation will not seek forward to the appropriate start time to enable the element to play. In fact, nothing will happen. In this sense the event is, in SMIL, a much "weaker" construct than the link. Its usage, however, is intended for activating small pieces of information relevant to the presentation currently playing, rather than taking the user to a whole new subject.

Events in Labyrinth (14) are used for modelling link contexts, allowing the specification of more powerful links, so that, for example, the result of activating a link depends on the user who selected it.

5.2 Activation conditions and source

Traditionally a link's activation condition is a mouse click, although mouse-over and keyboard control have been used even in early systems, e.g. (16). An event, on the other hand, can specify a whole range of activation conditions. Making the activation condition of a link explicit increases the flexibility of the link, and has indeed already been extended slightly in HTML, where the `a` element can be activated with the "Enter" key after it has been tabbed to. Other examples would include stroke or character recognition for hand-held devices.

Further extensions to activation conditions (for both links and events) could include activating when an item comes into view, e.g. through spatial scrolling. A video presentation could remain concealed until the reader scrolls to the appropriate text (or some prescribed time after having reached the text) and then a video window is opened and the video started up.

The activation source, the element to which the activation condition applies, can be specified by both a link and an event. In the case of a (Dexter or AHM) link, any FROM link-end specifier is an activation source. Similarly, the content of the `a` element in SMIL or (X)HTML is the activation source.

In XLink, a link's activation can be specified using an `actuate` attribute, whose values include `onLoad` and `onRequest`. A value of `onLoad` specifies behaviour similar to that of the `src` attribute of the `img` element in (X)HTML. That is, the destination's presentation is integrated into the source's presentation when the source's presentation is loaded. The `onRequest` value specifies that some other event needs to occur, such as the user clicking on an element with a `from` attribute. Further specification of the effect of the `actuate` attribute is outwith the scope of XLink.

For an event, an activation source can be specified in SMIL 2.0 by including an activation condition within the element. Also, any element can become an activation source through using a DOM (17). Note also that, since multiple elements can contain a reference to the same event, that a single event can cause the activation of multiple elements. This allows a basic mechanism emulating a multi-destination link, without the overhead of an extra structure to be maintained. For complex uses an aggregation mechanism may be useful, resulting in a structure similar to a link. Another mechanism is to remove the event information from the document itself and place it in an "event sheet", in the same way that style information can be separated out from the main document and included in a style sheet.

5.3 Relationship between source and destination

Synchronization

In the AHM there is no explicit synchronization between source and destination elements. There is implicit soft synchronization between any elements that continue to play and the newly started-up destination elements. The temporal/atemporal composition structure is used for specifying groups of source elements that can be deactivated (or paused) independently of one another.

In SMIL 2.0 soft and hard synchronization can be specified explicitly in the composition structure (using the `par` element). This is, however, not used in conjunction with the linking mechanism, so that no checking of breaking hard synchronized groups by stopping or pausing

the source elements is carried out. Similarly, no specification of hard or soft synchronization is given between the elements which continue to play and the destination elements.

XLink does not include the specification of temporal or spatial relationships between the source and destination of the links.

Events can specify synchronization in relationship to the event itself. If the event is timed (e.g. by being tied to a timed offset of a scheduled element) then it can be used for starting up the destination elements with a resolved scheduled relationship with any continuing elements. In SMIL 2.0 the event is not able to specify whether the starting element has a soft or hard synchronization relation with any continuing elements.

Transitions

The AHM allows the specification of transition information on a per-link basis, but does not include out-transition and in-transition information per link-end specifier.

In SMIL 2.0, out-transition and in-transition information can be specified on elements and used during the execution of the time-based and event-based schedules. Transition information is not, however, included within a link structure. For a SMIL 2.0 event the only transition information that can be used is that already existing on the activation source, i.e. the element referring to the event. This cannot be varied for different start (or end) events for the element.

Transition information is outwith the scope of XLink.

Interrupt control

In an AHM link, there is no means of specifying interrupt control between the source and destination elements. The model specifies only that a link-end starts up or deactivates. In the case that it is started up it is assumed that is starts playing (or pausing, depending on the destination play/pause state). If interrupt control were to be added to the model then each destination link-end would need some specification of how it interacts with the other link-ends. In addition, each destination link-end can potentially interfere with playing elements, and so the interactions with the (complete play state of the) document need to be specified.

Interrupt control is determined by the document structure rather than via a link element. A SMIL 2.0 link is thus unable to specify interrupt control explicitly, but the structure surrounding the destination of the link can include an `excl` element, which in turn allows the specification of interrupt priorities of its children using a `priorityClass` element. The same mechanism can also be used for children of a `priorityClass` element being activated by an event.

Specification of interrupt control is outwith the scope of XLink.

Destination display environment

The AHM provides control over the resources used through the channel mechanism. Each media item is played through a channel, specifying screen display information for spatial items (text, images, video) and audio channel information for audio items. Multiple use of the same channel is not addressed in the model, but is assumed to be taken into account in the authoring and playback system (in particular in CMIFed (13) or *GRiNS* (18)).

SMIL 2.0 allows the specification of a display environment for the destination elements by means of a `target` attribute. This allows a particular region (in SMIL) or frame (in HTML) to be specified. If the named display environment is not already active then (a new) one is opened with the specified name. In addition to the `target` attribute, the `show` attribute allows an author to specify whether the destination elements should be presented in a new display environment or not. The value of the `show` attribute is ignored if the `target` attribute has been specified.

The display environment is outwith the scope of XLink, although the `show` attribute has values of `replace` and `new` which specify whether the destination element should appear in the same or new window, respectively.

While the target attribute allows the specification of the display environment on the anchor, the `region` attribute allows the display environment to be specified for the element itself. This is used when the element is activated by means of the `begin` attribute and, in the case of linking, as a default when no `target` attribute has been specified.

5.4 Discussion of Link and Event Structures

A link uses a complex structure to record in detail exactly what state changes should happen in a playing presentation. Links bind the source elements and destination elements together in a single structure. This is a useful higher-level grouping, and allows properties to be attached.

While links can be modelled as separate objects, often they are stored with the source of the link in the document itself, as in (X)HTML. In an environment where the link information is stored separately from the content of the presentation, then multiple links can be created referring to the document content rather than being embedded within it. The question as to where linking information should be stored is not new, but the issues remain relevant.

An event is atomic, and thus more flexible, but at the same time less expressive. A SMIL event has more the notion that the information is stored at its destination. The information relating to any particular event is distributed around the document. It can be thought of as a multi-ended link, but with no central object storing the information. This makes maintenance of any but the most simple use cases difficult. On the other hand, it is precisely for the simple cases that the event is useful. In a system that supplies only document structure and links there is a large barrier for making minor adjustments to the presentation, since it requires the creation of extra structures and objects. Both (14) and (15) include events in their models of hypermedia.

While “typical” link and event structures already exist, they can both be altered to more or less of the information discussed in the previous sections.

6. Conclusion

The paper discusses two types of interaction constructs, the link and the event, as encountered separately in hypertext and in multimedia. Both are required when dealing with hypermedia. We showed how they relate to each other and discussed under what circumstances each of them is a preferred approach for modeling required presentation behaviour. An important factor is the presence of resolved, static time-based schedules and how the required behaviour relates to the schedules. To this end, we discussed the three notions of time-based scheduling, event-based

scheduling and link-based document traversal. These are united in terms of the activation and deactivation of the elements they refer to. Therefore, a single model can be created which specifies how each construct influences the activation and deactivation of the elements in a presentation. Often, useful presentation behaviour can be expressed using either construct and we discussed when to apply each.

"Traditional" hypertext link structures play an important role in computer-based information but are sometimes unwieldy for expressing diverse types of interactions in complex time-based presentations combining multiple media. A link gives the author a self-contained structure for recording useful information on what happens when a reader activates the link. Existing link structures can be usefully extended to include, for example, activation condition information, in- and out-transitions and hard and soft synchronization information. At the same time, where only slight alterations in the detail of the progression of the presentation are desired, the presence of this structure can be an authoring barrier. Events allow such small corrections to be included with comparatively little effort. Both links and events can be used to describe all the various aspects of activation that occur during the run-time presentation of a document.

The development of SMIL 2.0 is an important step forward in integrating time-based and event-based scheduling along with links in a Web environment. Another approach to associating activation information with a document is to include it in an external "activation sheet", in the same manner as style information can be captured in style-sheets. A fruitful direction for future work is to investigate all the types of information that can be associated with a document, such as spatial layout, style and timed and event-based activation of elements, and to develop strategies for deciding on the most appropriate form for capturing the information for a particular (XML) document language. Preliminary work in this direction already exists (19), (20), (21).

ACKNOWLEDGEMENTS

The authors would like to thank Frank Nack for his insightful comments on earlier versions of the paper. We would also like to thank other members of the W3C SYMM working group for many useful discussions related to these and other issues. This research was funded in part by ITEA project 99011, Real Time Internet Platform Architectures (RTIPA).

REFERENCES

1. Synchronized Multimedia Integration Language (SMIL 2.0) Specification. W3C Working Draft 21 September 2000. `http://www.w3.org/TR/smil20/`

2. L. Hardman, J. van Ossenbruggen, L. Rutledge, K.S. Mullender, and D.C.A. Bulterman. Do You Have the Time? Composition and Linking in Time-based Hypermedia. In: Proceedings of the 10th ACM conference on Hypertext and Hypermedia, Darmstadt, Germany pp. 189-196, February 21-25, 1999

3. L. Hardman, J van Ossenbruggen, L. Rutledge, D.C.A. Bulterman (1999). Hypermedia: The Link with Time. ACM Computing Surveys. December 1999, http://www.acm.org/pubs/articles/journals/surveys/1999-31-4es/a23-hardman/a23-hardman.pdf

4. M.C. Buchanan and P.T. Zellweger. Automatically Generating Consistent Schedules for Multimedia Documents. Multimedia Systems 1(2), pp. 55-67 1993

5. M. Jourdan, N. Layaida, C. Roisin, L. Sabry-Ismail and L. Tardif (1998). Madeus, an Authoring Environment for Interactive Multimedia Documents. In: Proceedings of ACM Multimedia '98, Bristol UK, 1998

6. L. Hardman. Modelling and Authoring Hypermedia Documents, PhD thesis, University of Amsterdam 1998. http://www.cwi.nl/~lynda/thesis

7. P.L. Schmitz. Unifying Scheduled Time Models with Interactive Event-based Timing. Microsoft Technical Report MSR-TR-2000-114, November 29 2000. Available at: ftp://ftp.research.microsoft.com/pub/tr/tr-2000-114.{html,doc,ps}

8. F. Halasz and M. Schwartz (1994). The Dexter Hypertext Reference Model. Communications of the ACM, 37 (2), Feb, 30 - 39. Also NIST Hypertext Standardization Workshop, Gaithersburg, MD, January 16-18 1990

9. XML Linking Language (XLink) Version 1.0. W3C Proposed Recommendation 20 December 2000. http://www.w3.org/TR/xlink/

10. L. Hardman, D.C.A. Bulterman, and G. van Rossum. Links in hypermedia: the requirement for context. In: Fifth ACM Conference on Hypertext Proceedings (Hypertext '93), Seattle, Washington pp. 183-191, ACM Press, November 14-18, 1993

11. SMIL 2.0 Transition Effects Module, http://www.w3.org/TR/smil20/smil-transitions.html

12. Transfer of Edit Decision Lists. ANSI/SMPTE 258M/1993

13. G. van Rossum, J. Jansen, K.S. Mullender, and D.C.A. Bulterman. CMIFed: A Presentation Environment for Portable Hypermedia Documents. In: Proceedings of the First ACM International Conference on Multimedia pp. 183-188 (1993). August 1993

14. J. Nanard, M. Nanard and P. King (2000). Media Construction Formalism: Specifying Abstractions for Multimedia Scenario Design. New Review of Hypermedia and Multimedia, this issue.

15. P. Diaz, I. Aedo and F. Panetsos. Modeling the dynamic behavior of hypermedia applications. IEEE Transactions on Software Engineering, in press.

16. B. Shneiderman. User Interface Design for the Hyperties Electronic Encyclopedia. In: Hypertext '87 Proceedings, Chaper Hill, North Carolina pp. 189-194, ACM Press, November 13-15, 1987

17. Document Object Model (DOM) Level 2 Events Specification, Version 1.0, W3C Recommendation 13 November, 2000. http://www.w3.org/TR/DOM-Level-2-Events/

18. D.C.A. Bulterman, L. Hardman, J. Jansen, K.S. Mullender, and L. Rutledge. GRiNS: A GRaphical INterface for Creating and Playing SMIL Documents. In: Seventh International World Wide Web Conference, Brisbane, Australia, April 14-18, 1998

19. J. van Ossenbruggen, L. Hardman, and L. Rutledge. Integrating Multimedia Characteristics in Web-based Document Languages. CWI technical report INS-R0024, December 2000, http://www.cwi.nl/static/publications/reports/INS-2000.html, http://www.cwi.nl/ftp/CWIreports/INS/INS-R0024.ps.Z

20. P.L. Schmitz (2000). A Unified Model for Representing Timing in XML Documents. Presented at WWW9 Workshop: Multimedia on the Web, May 15, 2000, Amsterdam. http://www.cwi.nl/~media/www9/TimingIntegrationPositionPaper.html

21. W. ten Kate, P. Deunhouwer, and R. Clout (2000). Timesheets - Integrating Timing in XML.Presented at WWW9 Workshop: Multimedia on the Web, May 15, 2000, Amsterdam. http://www.cwi.nl/~media/www9/Timesheets.www9.html