4 Multimedia Authoring Paradigms

The construction of a coherent hypermedia presentation composed from its constituent parts is a non-trivial task. To explore the requirements of a hypermedia authoring system designed to aid an author in this task, we describe a selection of both research and commercial authoring systems. These provide examples of the types of support that can be given to authors, and how this support can be provided in practice. We differentiate four authoring paradigms and discuss their advantages and disadvantages for editing features of a multimedia document model.
This chapter is based on work presented in [HaBu95b].

4.1 Introduction

In the previous two chapters we discussed and proposed a hypermedia document model which can be used to record sufficient information for storing a hypermedia presentation. A good model is a necessity for a good tool. A model only, however, is insufficient for solving the problem of how to create such a presentation. Tools are required for the creation, manipulation and deletion for individual parts of the model but in addition to such basic requirements the author needs an environment which supports the complete authoring process.

This and the following two chapters discuss authoring environments for multimedia and hypermedia documents. This chapter illustrates a selection of approaches implemented in existing systems. The following chapter, Chapter 5, states the full requirements for a hypermedia authoring environment and Chapter 6 describes the editing environment CMIFed.

We begin our analysis of the requirements for a hypermedia authoring environment by investigating a number of multimedia authoring systems that exist as either academic prototypes or as popular commercial systems. Each of these systems allows the creation of a multimedia presentation conforming to the system's own proprietary document format and uses its own suite of tools for creating the presentation.

Constructing a presentation consists of three major processes:

- creating and editing the media items comprising the presentation;

- assembling the items into a coherent presentation, where this includes the specification of the temporal and spatial layout of the items; and
- specifying the interaction between the reader and the presentation.

Our analysis of the authoring task concentrates primarily on designing an authoring system that supports the last two of these—the assembly and interaction processes. We are less concerned with the first—the creation of individual media items—which requires the use of specialist data editors for the range of media types used. Neither do we focus on the data formats used, nor the necessary trade-offs for authoring presentations destined to be played over a network.

In some respects, authoring multimedia can be compared with word processing. Both activities require the collection/generation of source material and the placement of these sources within a presentation environment. A generic word processor allows an author to layout information for use on a printed page. Depending on the features supported by the formatter, authors may be able to vary the font and size of the text, they may be able to vary the spatial layout of the information on the page, and they may be able to incorporate higher-level structures, such as chapters and sections, in the document. In the same way, multimedia authoring tools allow an author to integrate several types of information into a composite presentation. Unlike text, however, the temporal dimension often dominates the multimedia authoring process. In many respects, then, multimedia authoring is more akin to movie making. Here an editor is concerned that the individual shots that have been created are assembled into sequences which are in turn are grouped into scenes containing a single coherent thread of the story [RuDa89].

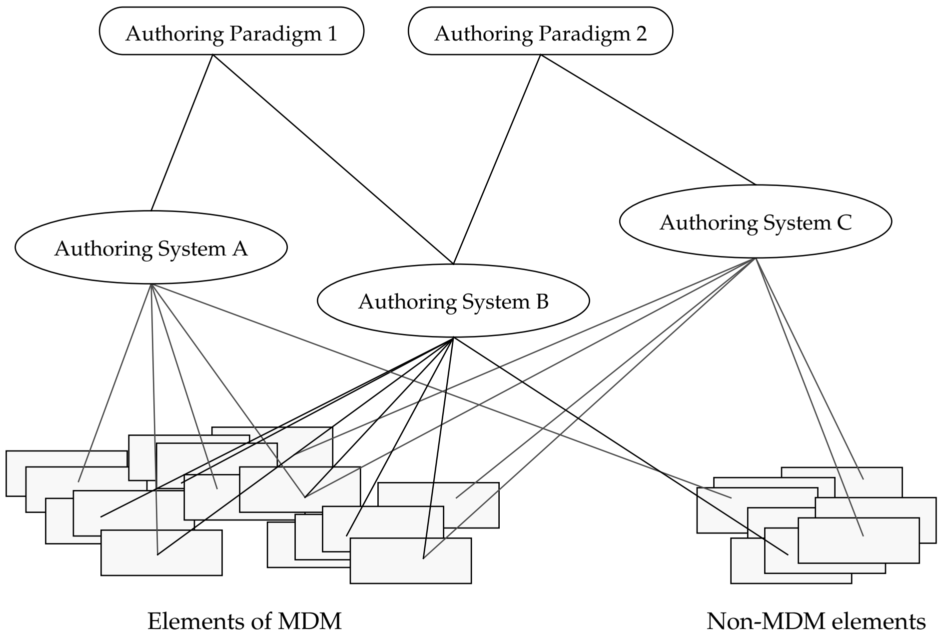
An author of multimedia has the same goal of communicating a message to the reader. In order to achieve this goal, the author is required to specify the individual parts of a multimedia document. To ease the task for the author, these specifications should be as transparent as possible and retain the emphasis on the manipulation of the message rather than on the document parts. This requires the presentation of the document parts to the author in a way that supports higher-level narrative manipulation. We term the different approaches used for this authoring paradigms. An *authoring paradigm* presents the author with a particular view of the document model. For example, in the word processing example a document can be viewed as a sequence of words in a text flow or as a layout of areas on a page. In the movie world the paradigm is the grouping of shots into sequences and scenes.

In this chapter we analyse the authoring approaches and individual functionalities of existing multimedia authoring systems. In order to compare these, however, we discuss them in terms of a multimedia document model. This model is less rich than the Amsterdam Hypermedia Model, defined in the previous chapter, since most multimedia systems do not, for example, include explicit link objects. Also, where objects that do correspond to the AHM are explicit they tend to have a less complex structure, e.g. composition of instances. The multimedia document model is not all encompassing, so some authoring systems may manipulate objects that lie outside the scope of the model, e.g. a user interaction history.

While the document model that is manipulated by these systems is similar, there are a number of distinct authoring paradigms which are used. An impression of the relationships among authoring paradigms, authoring systems and the parts of a document model is given in Fig. 4.1.

Having described a selection of authoring systems, illustrative of the authoring paradigms, we make an analysis of which paradigms are more supportive for editing which parts of the document model.

This chapter is structured as follows. We first give two sets of definitions: the authoring paradigms used for categorising authoring systems; and a multimedia document model used for comparing authoring facilities. In Section 4.3 we use these definitions to discuss a representative selection of academic and com-



An authoring system is able to edit elements of a multimedia document model. The approach used may conform to a single authoring paradigm, but is more often some combination of paradigms.

Figure 4.1. Relationship between authoring paradigms, authoring systems and elements of a multimedia document model (MDM).

mercial multimedia authoring systems. We use this as a basis for analysing the authoring paradigms for their suitability for the authoring tasks. We conclude with a summary of our analysis.

4.2 Definitions

In this section we present terminology for classifying multimedia authoring systems and discussing their features. We first present a multimedia document model to allow the discussion of individual features of the systems in terms of the document structures they manipulate. We then present four authoring paradigms which are used to classify authoring systems in terms of the style of interaction provided to the author.

4.2.1 Multimedia Document Model

To serve as a base for discussing multimedia authoring systems, we define elements of a multimedia document model. The model, which is in some respects similar to the AHM (defined in Chapter 3), is not dependent on any higher order information structuring. Fig. 4.2 gives an overview of the multimedia document model.

- A *media item* is the data associated with a single playable object in a multimedia presentation, for example a piece of text, an image, a video or a sound fragment. It may also be a piece of program code or a combined video/audio format.

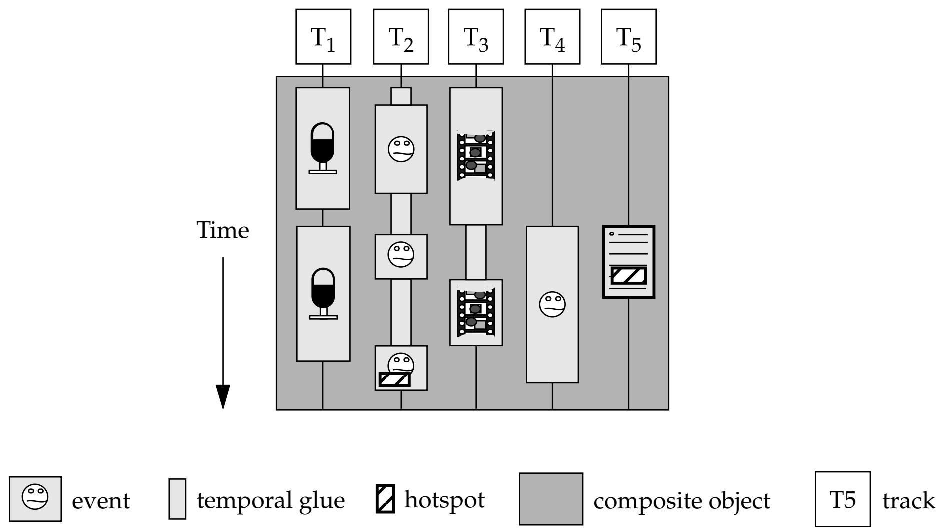


Figure 4.2. Multimedia document model overview

- An *event* is an action that occurs during a multimedia presentation. It may include a reference to a media item, e.g. play the media item for a specified duration, or it may contain more control-oriented information, such as wait one minute then jump to another part of the presentation. An event is an action and a media item is an object. Where only objects are specified in the multimedia document, it is up to a document player to interpret what to do with the objects, i.e. to make the translation from object to event.
- *A composite object* is used to refer to a collection of media items. While it is similar to an AHM composite component it does not possess properties beyond the list of its children. In other words it has no associated presentation specification, attributes or anchors specific to the composite.
- *Tracks* allow media items, events, or control information to be collected together in a single stream.
- *Constraints* are specifications of temporal or spatial relationships between uses of media items. An example of a temporal constraint is the AHM synchronization arc, e.g. start displaying an image 2 seconds after a piece of music begins. An example of a spatial constraint is that a text label should be placed centred at the bottom of an image.
- *Temporal glue* has a duration but no associated media item. This can be captured as an AHM atomic component that has a duration but no associated content.
- A *transition* is a presentation effect used when the system finishes displaying one media item and starts displaying another, e.g. a video item dissolves to the next video item.
- *Hotspot* or *button*. Most multimedia authoring systems have no explicit structures for anchors and links. They often, however, allow the specification of something that corresponds to an anchor value, e.g. an area of an image or a text string. This can be visualised, for example by drawing a border around it or using a different colour, and the reader can click on it. It is this visualisation, rather than the underlying structure, which is referred to as the hotspot or button.

4.2.2 Authoring Paradigms

The majority of multimedia authoring systems can be classified according to a number of different underlying paradigms: *structure*, *timeline*, *flowchart* and *script*. The paradigms provide different approaches to authoring. While we use these to classify the authoring systems discussed in the next section, more than one paradigm may be present in any one system.

Structure-based

Structure-based authoring systems, Fig. 4.3, support the explicit representation and manipulation of the structure of a presentation. The structure groups media items included in the presentation into “sub-presentations” which can be manip-

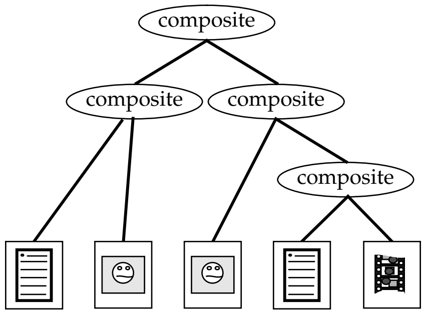


Figure 4.3. Structure-based paradigm

ulated as one entity, and thus can in turn be grouped. Although in principle the same object can belong to one or more groups, in current authoring systems this is not the case. The destinations of choice points in a presentation, that is where the reader is able to select to go to other parts of the presentation, are given in terms of the structure. The structuring may group the media items indirectly, where, for example, higher-level concepts are associated with each other and each concept is associated with one or more (groups of) media items.

Timeline-based

Timelines show the constituent media items placed along a time axis, possibly on different tracks, Fig. 4.4. These give an overview of which objects are playing when during the presentation. Timeline based authoring systems allow the specifi cation of the beginning and end times of display of a media item in relation to a time axis. Manipulation is of individual objects, rather than of groups of objects, so that if the start time or duration of a media item is changed then this change is made independently of any other objects placed on the timeline. The destinations of choice points are given in terms of a new position on the timeline.

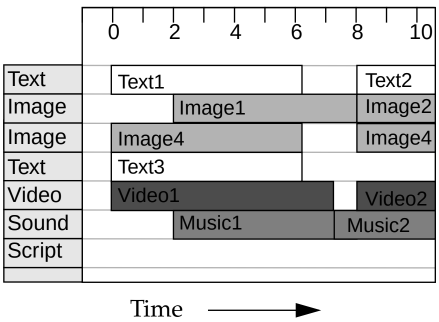


Figure 4.4. Timeline paradigm

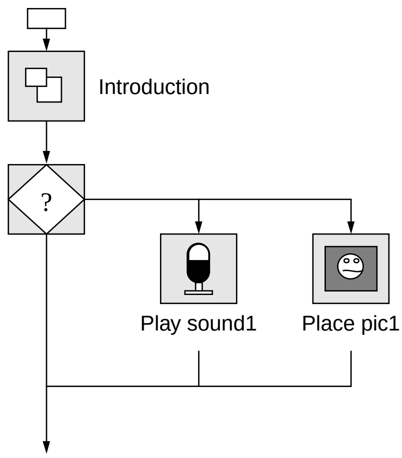


Figure 4.5. Flowchart paradigm

Flowchart
A flowchart gives the author a visual representation of the commands describing a presentation, Fig. 4.5. Authoring with a flowchart is similar to programming the presentation in a procedural way, but with an interface improved by icons for visualising the actions that take place. The narrative of the presentation can be reflected in the routines and subroutines used. The order of displaying or removing objects and other events is shown, but time is not represented explicitly. The destinations of choice points are given in terms of jumping to a new procedure.

Script-based
A script-based system provides the author with a programming language where positions and timings of individual media items, and other events, can be specifi ed, Fig.4.6. Authoring the presentation *is* programming. The destinations of choice points are given in terms of jumping to a new procedure.

```
set win=main_win
set cursor=wait
clear win
put background “pastel.pic”
put text “heading1.txt” at 10,0
put picture “gables.pic” at 20,0
put picture “logo.pic” at 40, 10
put text ”contents.txt” at 20,10
set cursor=active
```

Figure 4.6. Script-based paradigm

Discussion
Each of these paradigm described is based on a particular view of a multimedia presentation. The structure-based paradigm emphasizes the narrative structure of the presentation; the timeline emphasizes the temporal aspects; the flowchart and script emphasize the execution order of displaying and removing objects at runtime. The structure and timeline paradigms are of a more declarative nature, while the flowchart and script paradigms are procedural. The first two specify structural or timing properties of objects which are then interpreted as events by a system at play-back time. The second two list a sequence of actions, including those referring to media items, which are executed by the system.

While all four authoring paradigms include the notion of event, it is more explicit in the flowchart and script based paradigms. The flowchart and script paradigms are based on streams of events, where either an icon or a script statement specify what the event is. The structure-based paradigm manipulates composite and media items which are interpreted by the system at play-time as events. A timeline refers to objects representing media items, but, given that their presence on the timeline implies that they be played for the specified duration, are more correctly events. Other events, such as increasing the tempo of the presentation, can be difficult to visualize on the timeline.

The paradigms themselves are not mutually exclusive, but reflect a difference in emphasis. It is not that any one approach provides the ideal solution to an author's task and more often a combination is appropriate.

4.3 Analysis of multimedia authoring paradigms

The goal of this chapter is to compare the advantages and disadvantages of existing authoring paradigms. This section describes a number of authoring tools illustrating each of the four paradigms discussed in the previous section. The descriptions provide a basis upon which a comparison of different paradigms can be made. Most of the authoring system described are to be found in the academic literature, since they explore the more innovative approaches. We have also chosen to include a number of commercial systems, since these have proven themselves by surviving years of use by real authors. While a large number of authoring systems is commercially available we have selected Director, Authorware, and IconAuthor as being representative of the commercial systems.

Each sub-section describes a number of systems in terms of the paradigms and terminology discussed in the previous section. While a number of the systems described in this section use more than one of the paradigms described above, we have used the predominant paradigm to classify the system. The paradigms should be viewed as descriptive of the approach(es) taken by a system, and are used as a basis of making comparisons among systems. We wish to emphasize the strengths and weaknesses of the paradigms, rather than recommending one authoring system above another. For each paradigm we have selected one system which we consider as typifying the paradigm. Where other systems illustrate extra features or insights we describe these also. The descriptions of the systems highlight authoring features that will be referred to in the discussion of desired features. This section is not a review of the best authoring system to purchase.

4.3.1 Structure-Based Authoring Systems

While our own system CMIFed is a structure-based authoring system, we discuss it in detail in Chapter 6 and omit it here.

4.3.1.1 MAD

MAD (Movie Authoring and Design) [BRFS96] decomposes a multimedia presentation as a nested hierarchy, Fig. 4.7. This hierarchy is able to represent "acts", "scenes" and "shots", although these divisions are not imposed on the author. In a manner similar to a text outliner, the different levels of the hierarchy can be hidden or revealed and the position of items can be moved within the hierarchy. The start time of each item is calculated from the start times and durations of preceeding items and subitems in the hierarchy. The duration of an item can be calculated on the basis of the media item for video and audio, or can be specified by the author. The author also has control over playback of sections of the presentation by playing complete items or skipping forward to following items. MAD lacks any control of synchronization among items so that a single item with its associated parts can be played, but other items cannot start before it has finished. It is unclear to what extent there is control of spatial layout.

4.3.1.2 MET++

In the MET++ authoring system [Acke94] a multimedia presentation is considered to be a hierarchy of serial and parallel compositions of media items. The temporal layout of the constituent items is derived from this composition hierarchy automatically. The building blocks consist of composite objects, called time layout objects, and media objects. Each has a starting time, a duration and an associated virtual timeline. The media object contains a reference to a media item and associated attributes, such as position, which can vary with time. Both object types are incorporated into a hierarchical structure with the media objects as leaf nodes and the time layout objects as intermediate nodes. When the start time or duration of an event is altered all time positions are recalculated. Any object can be stretched or reduced in time. In the case of a composite object, the transformation is applied throughout the descendant hierarchy.

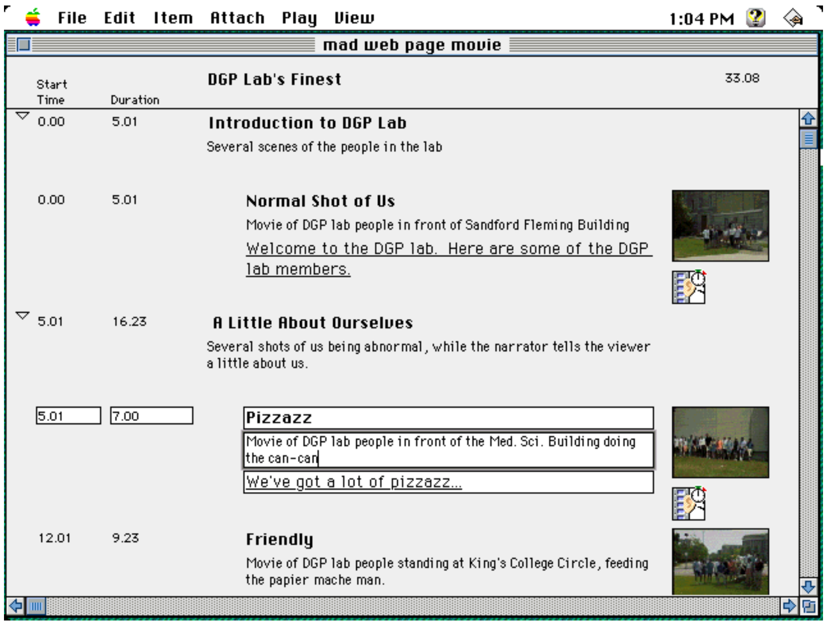
The timeline representation, Fig. 4.8, allows the visualization and manipulation of the hierarchical structure—combining structure and timing information in one representation. The timeline shows the values of attributes that vary over time, e.g. the horizontal and vertical positions in the figure. This representation could also be used for other object parameters, e.g. the volume of an audio object, or the fade-in rate of an image or video.

4.3.1.3 Mbuild

The multimedia authoring system Mbuild [HaRe94], Fig. 4.9, uses a hierarchical structure of composites and multimedia items for editing and reusing composite multimedia data. The timing of the presentation is determined when the highest ranking composite object is determined and is calculated using temporal glue. Authors are able to create empty hierarchical structures, reflecting the narrative of the presentation, and only later fill them with the desired media items.

4.3.1.4 Discussion

Structure-based systems allow the explicit specification and manipulation of a presentation's structure. The advantage of this is that authors are able to use the structure as a storyboard, i.e. a representation of the narrative, for the presentation. The author is thus able to manipulate the narrative directly. Since the presentation consists of different levels of structure, this can be viewed at different



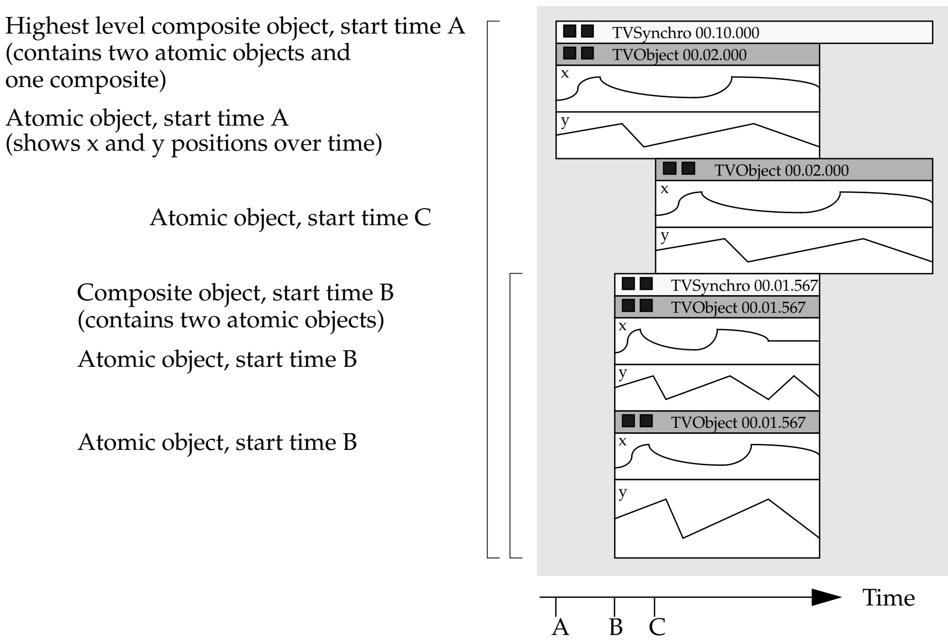
The indentation shows the level in the hierarchy. Each item in the presentation has three fields: title (bold), screen directions (small) and narration or dialogue (underlined) and may also have associated commentary, music, video or storyboard frames—shown to the right of the script. Start times and durations are shown to the left of the script.

Figure 4.7. MAD (Movie Authoring and Design) script view

levels of detail, allowing the author easy navigation of the narrative. Another advantage is that since the structure is able to indicate an ordering it can be used for deriving the timing for the presentation, as demonstrated in MAD, MET++ and Mbuild. The timing can thus be visualized and edited, at least to some extent, in the structure-based view. It may even be possible to have the structure displayed along a timeline, as illustrated by MET++.

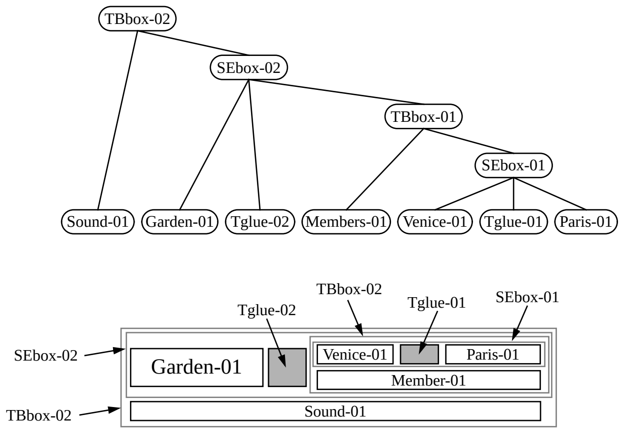
Synchronization constraints can, at least in principle, be defined between media items, between a media item and a scene (or other structure), or between two structures. While the fact that a timing relation exists could be shown in a structure-based view, it requires a time-based view to show the actual influence of the constraints specified. One example is the Firefly system, [BuZe93], where timing constraints, of some complexity, can be defined, but since the view is not time-based the resultant timing is not visualized.

Spatial layout is defined by assigning a position on the screen to the media item. This can be done by positioning the item where it should appear or, as in



Each *TVObject* represents a media item and specifies its x and y positions. *TSynchro* titles are composites of the objects below them. Both types of objects can be cut, copied, and pasted, stretched and shrunk. (Adapted from Fig. 7, [Acke94].)

Figure 4.8. Time composition view in MET++.



The upper part of the figure shows the hierarchical structure of a presentation. The lower part shows the equivalent structure as displayed in *Mbuild*. (Adapted from Fig. 20, [HaRe94].)

Figure 4.9. Hierarchical structure in *Mbuild.*

MET^{++}, by specifying the x and y positions over time. Neither method is specific to the structure-based paradigm, and in both cases it is difficult to get an overview of the position of the object relative to other objects or over time.

Links can be created among structures, allowing, at least in principle, source and destination contexts to be specified along with the source anchor (i.e. the value from which the hotspot is derived).

A problem with the structure-based paradigm is that extra authoring effort has to be expended to create the initial structure. Our hypothesis is that the benefits of understanding and manipulating of the presentation's structure will outweigh the initial effort.

A purely structural view of the presentation gives no understanding of the timing of the presentation. This can, however, be combined with the timing information, as demonstrated in MET^{++}, Fig. 4.8, and Mbuild, Fig. 4.9. Where structural and timing information cannot be combined, multiple views of the presentation can be a solution.

A summary of the properties of the structure-based paradigm is given in Table 4.1.

4.3.2 Timeline-Based Authoring Systems

4.3.2.1 Director

Director [Macr97] is a commercial system designed for creating animation-based presentations. Graphics, text, audio and video media items can be placed on a timeline, or score as it is termed in the system, Fig. 4.10. The timeline is divided into discrete time intervals, called frames, whose speed of playing is determined by the current rate of play, called tempo. The tempo can be changed at any frame. The timeline has a number of associated tracks, where, apart from a number of effects tracks, any media item can be placed on any track. A media item has a position in each frame, and the author can describe a path for the media item to follow through a series of frames. Sections of the timeline can be cut, copied and pasted. Jumps to other parts of the timeline are implemented via a " goto frame" command in the scripting language. Each frame, media item or anchor within a media item, can have an associated script. The script is executed when the end-user interacts with its associated object, normally by clicking with a mouse.

Scene breaks can be recognised by the author by sudden changes of media items on the timeline. These are not automatically marked by the system since there are no special frame types (e.g. a " beginning of scene" frame), nor groupings of frames. An author is, however, able to add explicit markers to frames, allowing jumps to the marked frame, and thus to the marked beginning of a scene.

One of the effects tracks is a transition track, allowing the specification of transitions. The transition is recorded with the first frame of the following sequence, rather than the last frame of the previous one. The transition has a type (for example dissolve or checkerboard), a duration and a choice of whether the whole display area is affected or only the differences between the frames.

4.3.2.2 The Integrator

An explicit goal of the authors [SFHS91] is to create a development environment for interactive multimedia that relieves a developer of programming work. The central tool in the environment is the high-level Integrator which is used to assemble various media items into a multimedia application and to specify ways of interacting with the application.

Property / Paradigm	show narrative structure	edit narrative structure	navigate narrative	show timing	edit timing	show synchronization	edit synchronization w.r.t. objects	edit synchronization w.r.t. structure	show layout	edit layout	show interaction	edit interaction	interaction diversity
Structure-based	++	++	++	0	+	0	+	++	--	0	-	0	-

Key: ++ very good, + good, 0 neutral, - bad, -- very bad/not possible

TABLE 4.1. Properties of structure-based authoring paradigm

The basic paradigm used in the Integrator is the timeline, where a pool of media items can be sequenced in virtual time, and then mapped for display onto real time. In the Integrator the multimedia presentation is represented as a set of tracks. Timing and synchronization of multimedia items within a single track are determined by their horizontal positions on that track. Timing and synchronization of multimedia items across different tracks are determined by vertical relationships of objects across tracks (similar to AHM synchronization arcs). As well as media item tracks, the Integrator allows input or control tracks and a timing track which allows the tempo of the tracks to be altered, similar to Director.

Authoring is carried out by placing an icon representing a media item on one of the tracks at a specifi c time. A static item, such as an image, will remain on display until another item occurs on the same track.

While the main authoring metaphor is the timeline, composite objects can also be created, e.g. an image with a graphic overlay, or a slide show. A composite object can be placed on the control track of the timeline, and can be opened to view the layout of objects on its own timeline. The composite object appears as one object on the timeline, which makes it diffi cult to get an overview of all the

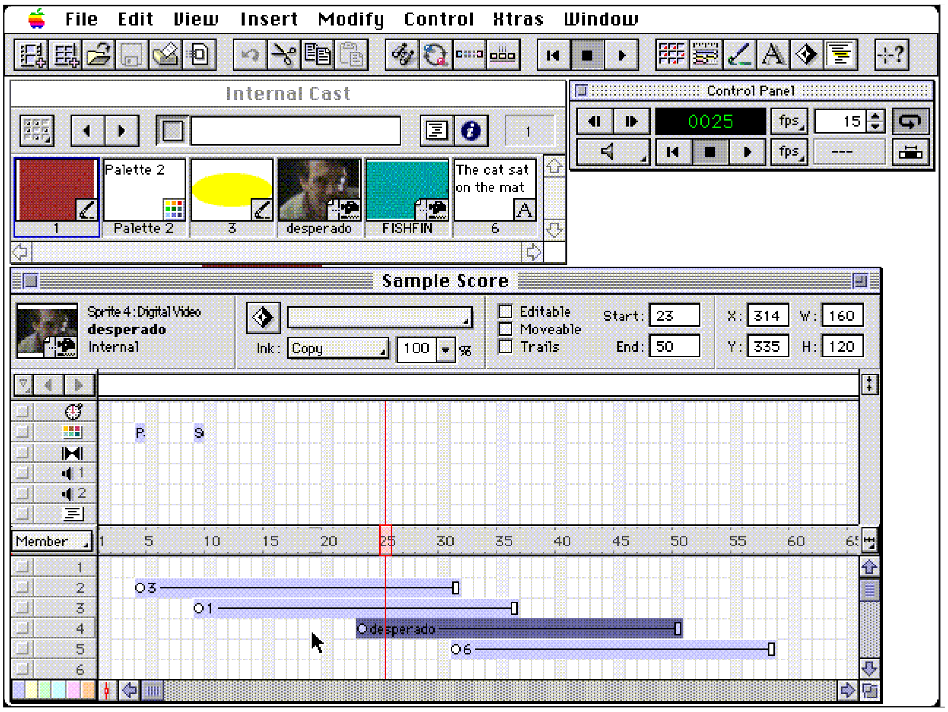


Figure 4.10. Director

objects making up the presentation. Also, time dependencies between objects at different levels of the hierarchy are impossible to specify.

The authors observe that a timeline represents the parallel nature of multimedia applications better than a flowchart. They have, however, included several "flow" operations that can be added to the control track of the timeline, including iteration and conditionally branching constructs. These add to the power of the specification language, but make the visualisation of the presentation on the timeline difficult to interpret.

Transitions can be specified in the system. These are associated with the media item events rather than being separate events themselves, and can occur at the beginning of an event (e.g. fade up from black), at the end of an event, or can join two events (e.g. a video dissolves to another video).

4.3.2.3 Discussion

Timeline-based authoring centres around presentation over time. It uses a time-axis as the main method of organising the (temporal) positioning of media items in the presentation. It is visualized as a line with marked-off time intervals. The advantage of this approach is that the start times and durations of the media items in the presentation are displayed explicitly, and in principle can also be manipulated directly. The timeline can also be used to show the values of properties of the media items that vary over time (as demonstrated in MET++).

A further advantage of the timeline is that synchronization constraints, where these exist, can also be shown and in principle manipulated directly. These constraints can be between media items or between a media item and the timeline. We, however, feel that the synchronization conditions should be expressed directly between (parts of) the media items themselves, so that if other durations are changed the system, rather than the author, can resolve the specified constraints. For example, if a video sequence is shortened or lengthened the corresponding subtitles stay synchronized with the correct parts of the video.

Spatial layout is specified by assigning a position on the screen to the media item. This can be done by positioning the item where it should appear or by specifying the *x* and *y* positions over time. Neither method is specific to the timeline-based paradigm. It is difficult to get an overview of the position of the object compared with other objects and over time. Although no overview is available, the timeline does provide the author with easy access to a screen view from any point along the timeline.

Where links are specified, these are via scripts associated with an object being displayed on the screen. The script defines the destination of the link and may also contain some sort of transition information, such as "dissolve to next scene in 2 seconds".

The main problem with only a time-based representation is that for long presentations it is difficult to navigate around. Also scene breaks, or any overview of the narrative, need to be recognised implicitly, for example, by an abrupt change in the objects on the timeline. Because scenes are not represented explicitly it is also not possible to create synchronization constraints in relation to a scene. Control flow can be added to a presentation as an object on the timeline but its effect cannot be visualized using the timeline.

A summary of the properties of the timeline-based paradigm is given in Table 4.2.

4.3.3 Flowchart-Based Authoring Systems

4.3.3.1 Authorware

Authorware (chapter 12 of [Bufo94], [BuHe93][1], [Macr97]) is a commercial system for creating interactive multimedia presentations for computer based training and kiosk applications, Fig. 4.11. To create a presentation, icons representing actions are selected and incorporated into a flowchart defining the sequence of events in the presentation. Flowcharts can be grouped into subroutines and nested to arbitrary levels. This is often necessary, since there is a limit to the display area for any one flowchart. The hierarchy of subroutines can be used by the author as an outline, or storyboard, for working on the presentation top down—first by stating the sections in the presentation and then filling them in. The flowcharts remain procedural however, and there is no way of getting an overview (via a timeline) of which media items will be played on the screen when. Interactions, on the other hand, can be fairly complex and go far beyond links, which are implemented as "jump to there" commands.

Property / Paradigm	show narrative structure	edit narrative structure	navigate narrative	show timing	edit timing	show synchronization	edit synchronization w.r.t. objects	edit synchronization w.r.t. structure	show layout	edit layout	show interaction	edit interaction	interaction diversity
Timeline	-	--	-	++	+	++	++	--	+	0	--	--	-

Key: ++ very good, + good, 0 neutral, - bad, -- very bad/not possible

TABLE 4.2. Properties of timeline-based authoring paradigm

1. The packages Authorware and IconAuthor were compared in [BuHe93] in 1993. Note, however, that both packages have newer versions on the market which may differ substantially from those reviewed.

4.3.3.2 IconAuthor

IconAuthor ([Aimt97], chapter 12 of [Bufo94], [BuHe93][1]) is a commercial package providing a suite of editors for different data types. Objects created by these, and other external editors, can be assembled in the central application builder for inclusion in the presentation. This is icon-based with flowcharts constructed from a library of icons representing actions comparable to those found in conventional programming languages and other more specialist icons for media-presentation and interaction. There is no enforcement of any programming discipline, so that large, unstructured graphs can be created. The author is given some help with flowchart navigation through being able to zoom in and out of the flowchart representation, and being able to simplify the display by collapsing or expanding collections of icons. Previewing the presentation is possible from the beginning or from a selected starting point.

4.3.3.3 Eventor

The creators of Eventor (Event Editor), [ENKY94], argue that authoring facilities should apply a divide and conquer approach, which they support by providing three different views of the presentation—temporal synchronizer, spatial synchronizer and user interaction builder. They distinguish timeline-based and

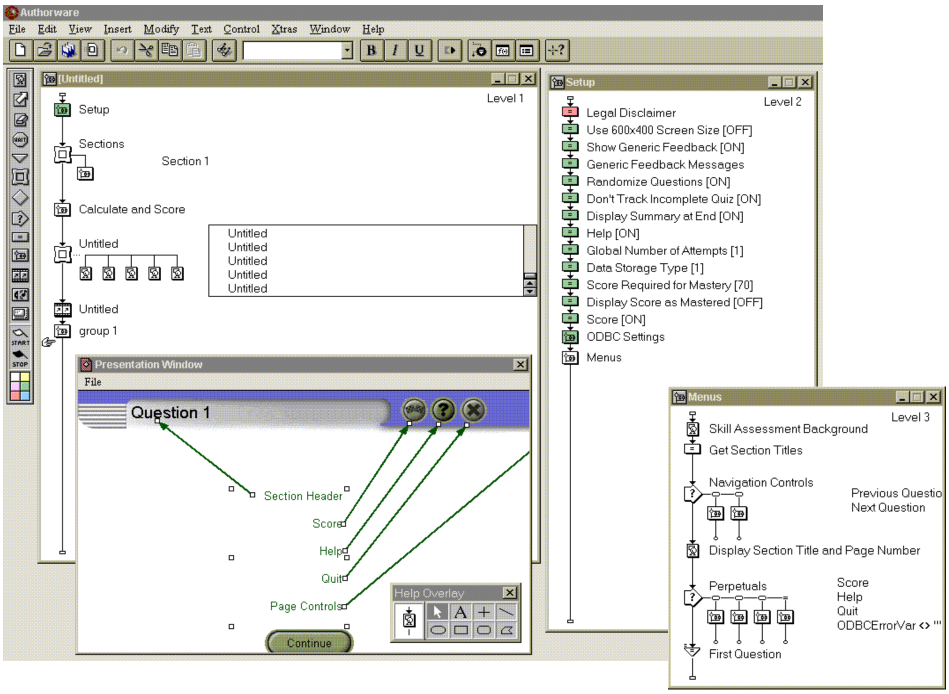


Figure 4.11. Authorware

flowchart paradigms (which they refer to as event-based), and aim to incorporate the advantages of both in an authoring system. Eventor is based on CCS, Calculus of Communicating Systems, a formal specification mechanism. This allows a formal specification of the behaviour of the presentation, which can be used for checking, for example, syntactic correctness. In addition, they provide automatic aids for validating, for example, temporal constraints.

Basic units of programming in Eventor are media items (called basic objects). Temporal synchronization can be specified among media items and composite objects. Synchronization points (similar to AHM anchor values) can be marked in video. The temporal synchronizer visualises the composite objects in a flowchart style structure. The spatial synchronizer allows the author to specify paths and scaling transformations by demonstration—these are tightly coupled to the temporal synchronizations. (While this provides a more direct method of interaction than in MET^{++}, section 4.3.1.2, the latter visualises the spatial movements in time more explicitly, Fig. 4.8.)

4.3.3.4 Discussion

In a flowchart, control is the emphasized view: events are executed in turn, determined by the surrounding control structure. The advantage of the flowchart paradigm is that it incorporates more powerful interaction commands. For example, standard multiple choice question formats are often provided which support the creation of links to a different section from each answer.

The paradigm provides some form of abstraction, but this is a grouping of commands in the form of nested flowcharts rather than relations among (groups of) events. This allows the narrative structure of the presentation to be reflected by the different levels of flowcharts. Although this is a useful view of the presentation's structure, it is difficult to find which items are displayed on the screen in the middle of a flowchart, since, e.g., background images may have been displayed before the flowchart was executed. This means that a sub-scene cannot be played independently—since the state of the presentation is known only by playing the presentation.[2]

The only form of timing specification is through the use of "display item" and "erase item" commands in the flowchart. This leads to three disadvantages. Firstly, if a number of items are to be displayed simultaneously then this cannot be specified, but only approximated by using a number of "display item" commands one after the other. Secondly, synchronization relations among items cannot be specified. Thirdly, it is not clear from the script which objects are on display at any particular time. This could, however, in principle be calculated from the scripts and displayed in a different view (as in, e.g., Eventor). A timing overview is sorely needed in this paradigm, since an object may be displayed at

2. Authorware provides a choice of playing a single item or playing from a prespecified flag. IconAuthor plays from a selected starting point.

Property / Paradigm	show narrative structure	edit narrative structure	navigate narrative	show timing	edit timing	show synchronization	edit synchronization w.r.t. objects	edit synchronization w.r.t. structure	show layout	edit layout	show interaction	edit interaction	interaction diversity
Flowchart	+	0	+	-	--	--	--	--	--	-	-	+	+

Key: ++ very good, + good, 0 neutral, - bad, -- very bad/not possible

TABLE 4.3. Properties of flowchart-based authoring paradigm

the beginning of a long script and erased only at the end. For the same reason, an author may forget to erase an object when it is no longer needed.

Just as timing is given through the use of commands, so is the spatial information for an object. Where the other objects are placed on the screen is only to be found by looking through the flowchart. Any overview of the objects in time, or their relations with respect to other objects is thus not available. The Eventor system introduced the spatial synchronizer to help with this problem.

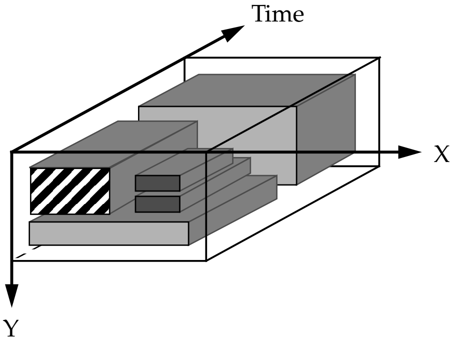
Links are specified via commands associated with hotspots which define which playing objects should be erased and which new objects should be displayed.

A summary of the properties of the flowchart-based paradigm is given in Table 4.3.

4.3.4 Script-Based Authoring Systems

4.3.4.1 Videobook

The Videobook system, [OgHK90], was designed to incorporate a time-based, media-composite data sequence with the hypertext node and link concept, allowing the construction of composite multimedia nodes. The system presents media items and hotspots according to a script specifying their presentation parameters—timing and layout. The script is visualised as a three-dimensional display showing the layout of each object along a timeline (Fig. 4.12). The system thus provides a low-level scripting language for the author to specify a presentation, which is then given a higher-level visualization along a timeline. The author is provided with some amount of structuring support, since each scene is defined in a separate script and scripts for scenes can contain nested sub-scenes. Synchronization of events is specified by giving the start time of an event with respect to the scene. Although the multimedia document has an underlying structure-based paradigm, the structure is interpreted from the author-defined



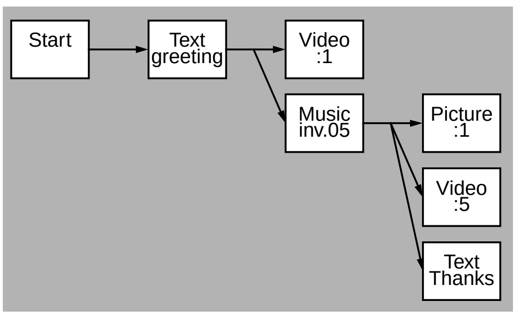
The layout and start times of the media items, defined by the author in a script, are given a three-dimensional visualisation.

Figure 4.12. *Videobook* scene.

script, rather than being used as the basis of editing and for generating the script.

4.3.4.2 Harmony

Harmony [FSMN91] has goals similar to those of Videobook, integrating dynamic media items into a hypertext system. Each object is considered as a node and there are links between nodes. Links are used for specifying the timing relations between nodes, using expressions such as <*aVideo, started: 30, aMusic, play*> which specifies that the piece of music starts 30 seconds after the video finishes. The notion of a composite object, or object group, is introduced, where, if a composite is the destination of a link, a message is broadcast to all members of the composite when the link is traversed. A scenario viewer displays the (derived) structure of a scenario. This is illustrated in Fig. 4.13, showing the time ordering or relations among media items.



The presentation structure as shown in the Harmony scenario viewer.

Figure 4.13. *Harmony* scenario structure.

4.3.4.3 Command Streams

Some authors have taken the script form of specification even further[HeKo95]. Here, the authors view a presentation as a sequence of (possibly synchronized) command streams, where each stream consists of an ordered collection of commands, each of which is assigned its own execution time. If a command stream starts to fall behind, then commands can be skipped to allow the stream to catch-up. The command stream contains sufficient information to allow it to be played not only backwards, but also in both directions at a higher speed (to allow the presentation to be scanned until the reader finds the appropriate part).

It is interesting to note that deficiencies the authors plan to resolve in a revised model are grouping the commands into logical groups (i.e. the introduction of hierarchical structure) and knowing the duration of an object (derived from its first appearance on the screen and its later removal) so that it can be skipped when large jumps in the presentation are made.

4.3.4.4 Discussion

The script-based systems are in essence similar to the flowchart in terms of their flexibility and power of expression, but through the direct use of the scripting language are likely to be more flexible.

In terms of authoring support, however, they lack tools for viewing the procedure calls in any structured way. This in turn leads to more likely program structure errors. Even if the narrative structure of the presentation has been reflected in the script structure, then it remains difficult to manipulate at a high level. As with the flowchart, timing information for the presentation is embedded in the lines of code (with the exception of command streams [HeKo95] where the lines of code have explicit times). Spatial layout information is also given via the lines of code. Since the structure, timing and spatial layout information is present in the code it is possible to derive a structure or time-based visualisation. For example, in Videobook the space and time coordinates of items are shown in a 3D time-based representation and in Harmony the structure can be viewed.

With no further support, this is a tedious, low-level method for specifying a multimedia presentation. It can, however, be the most flexible, since with a more general language the author is not restricted to the actions supplied by an authoring system for manipulating a document model.

A summary of the properties of the script-based paradigm is given in Table 4.4.

4.4 Conclusions

Events and Links

All systems described in this section are in principle capable of creating similar presentations, through specifying events and links. A complete event consists of a media item that can be played as part of the presentation, its (possibly derived)

Paradigm \ Property	show narrative structure	edit narrative structure	navigate narrative	show timing	edit timing	show synchronization	edit synchronization w.r.t. objects	edit synchronization w.r.t. structure	show layout	edit layout	show interaction	edit interaction	interaction diversity
Script	-	-	-	--	--	--	--	--	--	-	-	0	++

Key: ++ very good, + good, 0 neutral, - bad, -- very bad/not possible

TABLE 4.4. Properties of script-based authoring paradigm

start time, its (possibly derived) duration or end time and its position on the screen (for non-audio events). The way the information specifying an event is expressed in each of the paradigms is illustrated in Fig. 4.14.

- In the structure-based paradigm, the timing and position information are explicitly recorded together in the structure with (a reference to) a media item.
- In the timeline paradigm a media item is selected and assigned a position on the screen by the author (often by direct manipulation) and its start and end times specified via the timeline.
- In the flowchart and script paradigms a command refers to an object to be placed on the screen and at some later point in the script another command removes the object from the screen (where an audio item is generally only started).

Links are specified in different ways in each of the paradigms. This is summarised in Table 4.5.

- In the structure-based paradigm, the source component and anchor of the link, along with its associated context, can be specified. Similarly the destination component, anchor and context can be specified. In either case there may be multiple sets of these (although this adds yet another degree of complexity to the authoring process). The transition information can be recorded with the link structure.
- In the timeline paradigm the source and destination contexts of the link are restricted to being everything playing at some point on the timeline. The source of the link is either an anchor or a component on the screen which the end-user can select to follow the link. The source anchor can be defined as lasting for some extent along the timeline. The destination is specified via script command along with any transition information, such as duration or visual effects.

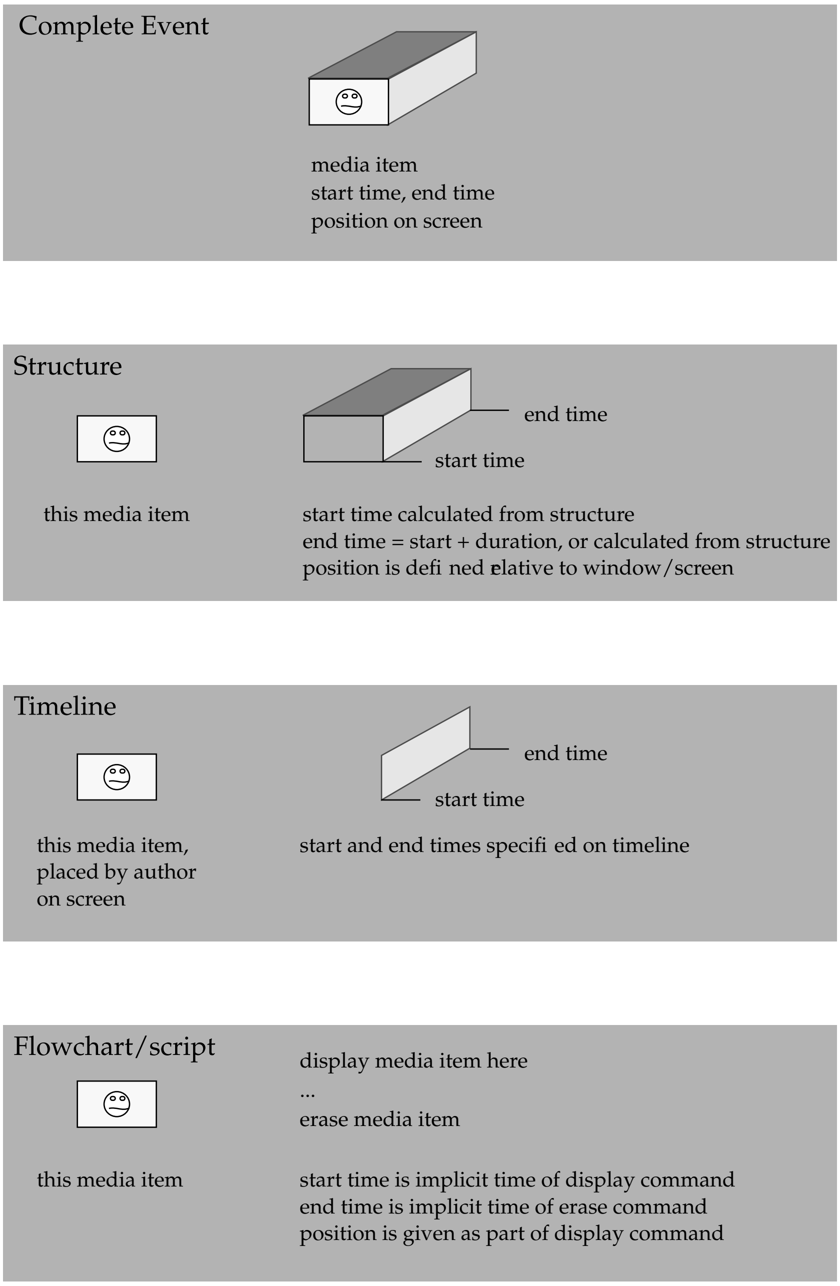


Figure 4.14. Specifying an event in each of the paradigms

- In the flowchart and script paradigms the source anchor is a hotspot object which has an associated script. The script specifies the source context implicitly by erasing some or all of the playing objects. The transition is also part of the script. The destination context is again implicitly defined as the objects that are displayed. There is no special object that can play the role of a destination anchor.

	Source anchor	Source context	Transition	Destination anchor	Destination context
Structure	yes	yes	yes	yes	yes
Timeline	hotspot	all events at point on timeline	yes	no	all events at point on timeline
Flowchart/ Script	hotspot	explicitly erase playing items	yes	no	explicitly display new items

TABLE 4.5. Specifying a link in each of the paradigms

Paradigms

Each paradigm emphasizes a different aspect of the creation or visualization of a multimedia presentation. In general, the richer the language the more precise the specification, but the more difficult it is to use. Also, each paradigm has its own emphasis, for example in the structure-based paradigm it is difficult to create scene changes where events at the end of one scene overlap with events at the beginning of the next. This can be perceived as an advantage or as a disadvantage. The trade-offs between the different paradigms are how flexible the behaviour is that can be specified, how easy it is to specify the behaviour, and how easy it is to view the specified behaviour (other than by playing the presentation). This is summarised in Table 4.6.

Paradigm \ Property	show narrative structure	edit narrative structure	navigate narrative	show timing	edit timing	show synchronization	edit synchronization w.r.t. objects	edit synchronization w.r.t. structure	show layout	edit layout	show interaction	edit interaction	interaction diversity
Structure-based	++	++	++	0	+	0	+	++	--	0	-	0	-
Timeline	-	--	-	++	+	++	++	--	+	0	--	--	-
Flowchart	+	0	+	-	--	--	--	--	--	-	-	+	+
Script	-	-	-	--	--	--	--	--	--	-	-	0	++

Key: ++ very good, + good, 0 neutral, - bad, -- very bad/not possible

TABLE 4.6. Properties of authoring paradigms

- Structure based systems are good for viewing, editing and navigating the narrative structure of a presentation, allowing different levels of detail to be shown as appropriate. While not ideal for viewing the timing of the presentation, the structure can be used for editing the timing by allowing (some) timing relations to be derived from the structure. The structure itself gives an ordering of the display of media items. Layout information is specified per event, and an overview of the layout at a particular time is possible only by playing the presentation. Interaction, other than playing the presentation, is restricted to specifying and following links. Links, however, can in principle be defined using source and destination anchors and contexts.
- Timeline based systems have no direct means of editing the narrative structure of the presentation directly, although it can be perceived and navigated as discontinuities of groups of objects along the timeline. The timeline is, however, the best way of showing when objects are displayed on the screen and synchronization relationships among events. It is not necessarily the best way of editing the timing, since although timing of individual objects can be changed, every object has to be manipulated individually, unless some form of structuring is present. Layout is specified per object per time unit, so an overview of all objects at a certain time is possible. The layout, and other properties of an event, can also be shown as a function of time, e.g. in MET^{++}. Interaction specification is even more restricted than in structure based systems, since links are often only jumps to some other point on the timeline.
- The flowchart and script paradigms are comparable in power of expression, where editing and viewing the result tend to be more cumbersome with scripts. Reflecting the narrative structure in the structure of the flowchart, or script procedure calls, is possible but not compulsory. Similarly, navigation of the narrative is only as easy as the procedural correspondence maintained during editing. Timing information, on the other hand, cannot be shown (although as mentioned previously, this may be derived for viewing in a timeline). Layout information is specified per object, generally as part of the command to display the object. An overview of the layout at a particular time is possible only by playing the presentation. The flexibility of the interaction that can be specified is high, and the flowchart tools can help with its specification. Viewing the interaction remains a problem, where the only methods are to run the presentation to check the various possible paths or to navigate the flowchart/script specification.

While each paradigm has its own strengths and weaknesses, we do not wish to choose one paradigm above another, but seek to combine them in a way that takes advantage of their complimentary strengths. The paradigms illustrate different ways of providing similar functionality in a hypermedia authoring environment. They do not, however, provide a solution to the problem of which functionality should be provided. We tackle this question in the following chapter.