KARATINA UNIVERSITY
COM 429
MULTIMEDIA SYSTEMS

Authoring Tools

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

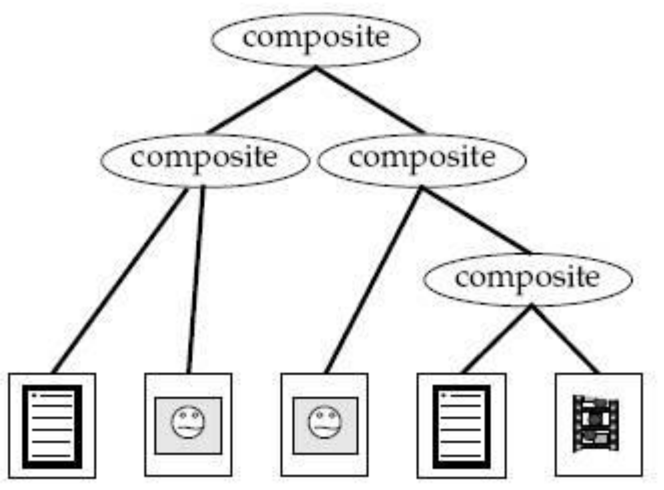


Figure 4.3. Structure-based paradigm

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 manipulated 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 me dia items placed along a time axis, possibly

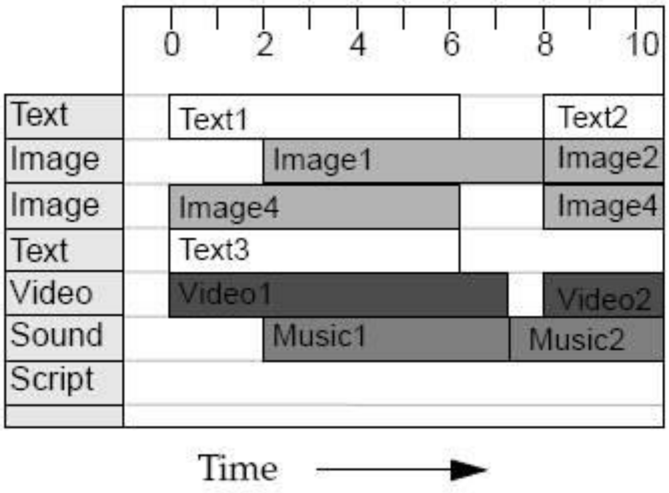


Figure 4.4. Timeline paradigm

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 specification 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.

Structure-based paradigm

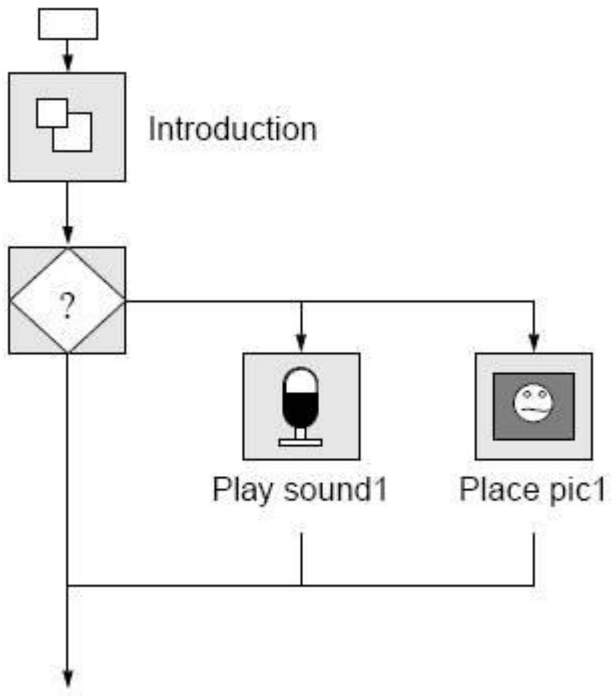


Figure 4.5. Flowchart paradigm

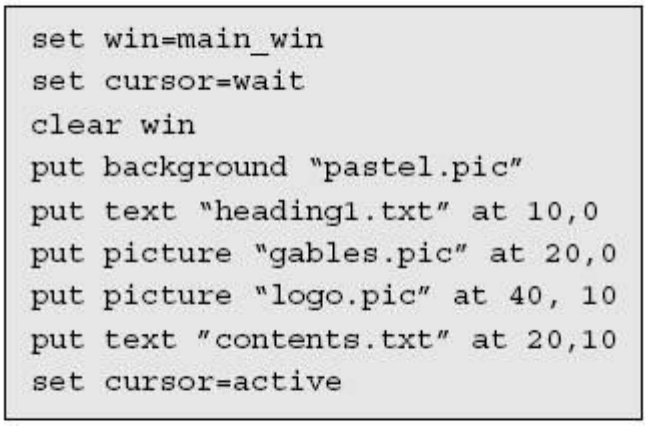
Flowchart

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 specified, 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

Actually authoring is just a speeded-up form of programming without the need to know the intricacies of a programming language but an understanding of how programs work is necessary (Merrill, 1985).

Authoring Interface

The interface of an authoring tool plays an important role in attracting the users to use the tool. An appropriate graphical user interface should be selected for different multimedia applications, which naturally have different roles to extend. In the authoring environment, the authoring interface is referred to as the authoring metaphor. This basically means the way applications are structured. Some of the current examples that are available are given below.

Slide-show Metaphor

Each screen of a presentation is seen as a slide object. This metaphor is an excellent choice for authoring linear presentations. However, it is not suitable for random interactivity. Some examples of current tools that apply this metaphor are Microsoft's Powerpoint by default a linear presentation e.g., PowerPoint, ImageQ

Book Metaphor

In this metaphor, the application is thought of as a book containing many numbers of pages, which can contain any number of multimedia objects and interactivity. Each object has an associated script, which contains any event handler. It is suitable for Encyclopedia, fairy tale story, etc. It also requires programming skills of a certain level. An example of such a tool is the Asymetrix's Toolbook.

Timeline Metaphor

A timeline metaphor works with a time scale diagram and places objects and events on the time scale in correct relationship. It is especially valuable when dealing with a number of dynamic activities. Some authoring tools that apply this metaphor are Macromedia Director and Action!

Icon Metaphor

The application structure is built by dragging icons from an icon palette into the workplace to form a flow-line of icons. These icons represent objects and events that will occur according to the sequence of the arrangement of the icons. Authors can now concentrate on the application logic and content rather than on the details of programming. Developing sophisticated application is still a difficult and complex task. For example, AimTech's IconAuthor and Macromedia Authorware.

WHAT IS AN AUTHORING SYSTEM?

An Authoring System is a program which has pre-programmed elements for the development of interactive multimedia software titles. An authoring system speeds up the development of multimedia applications (e.g. ready-made buttons, etc). They create multimedia applications in a fraction of the time that would be required by script programming tools. Buttons, dialog boxes, etc. can be brought onto the screen, positioned and linked to functions without writing any code. Interface design and screen editing are made much easier and quicker. Animations can be complex and detailed and yet created in a relatively short time. However the same amount of time required to plan, design and create the content. Authoring systems vary widely in orientation, capabilities, and learning curve. There is no such thing (at this time) as a completely point-and-click automated authoring system; some knowledge of heuristic thinking and algorithm design is necessary. Whether you realize it or not, authoring is actually just a speeded-up form of programming; you don't need to know the intricacies of a programming language, or worse, an API, but you do need to understand how programs work. It generally takes about 1/8th the time to develop an interactive multimedia project, such as a CBT (Computer Based Training) program, in an authoring system as opposed to programming it in compiled code. This means 1/8 the cost of programmer time and likely increased re-use of code (assuming that you pass this project's code to the next CBT project, and they use a similar or identical authoring system). However, the content creation (text graphics, video, animation, audio etc.) is not generally affected by the choice of an authoring system; any production time gains here result from accelerated prototyping, not from the choice of an authoring system over a compiled language. The ***authoring paradigm***, or ***authoring metaphor***, is the methodology by which the authoring system accomplishes its task. An authoring system is a set of software tools for creating multimedia applications embedded in an authoring environment. A person who creates applications for multimedia integration, for example, presentation, is called an author. The processes together are called authoring other components which belong to the authoring environment, such as multimedia. Consider an application, which coordinates a multimedia presentation- This application needs to provide a dynamic behavior and support several users' actions to integrate media to a required multimedia presentation. To implement an application with such dynamic support requirements, several processes must be programmed. This kind of application can be either written in a programming language, or implemented using an authoring system. It will usually comprise of

- Hardware
- Firmware (software that is permanently built into the hardware) and
- Assembly tool. (An authoring too! that arranges multimedia objects into a presentation or an application, dealing with their relationships in space and time).

When multimedia application is produced an authoring system is used. The author goes through several stages and like in systems development where possible a users input is important. This might imply some additional work at any previous steps or repetition of some.

I) Concept generation:

This step identifies

i. the application audience,
ii. The application type (presentation) interaction, etc.),
iii. Application purpose (inform, entertain, teach, etc.) and
iv. The general subject matter.

At this stage, the authoring system cannot help.

II) Design

The style and content of the application must be specified. The object should include and generate enough detail so that the following stages of content collection and assembly can be carried out by the authoring system without further, interruptions. However, the authoring system should still be tolerant of some kind of revisions. At this stage, the design parameters are entered into the authoring system. The authoring system can take over the task of documenting the design and keeping the information for the next steps of outlining, story boarding, flow charting, slide sorting, and scripting. The other task in the design stage is to decide which data files will be needed in the application, such as audio, video and image files. A list of the material should be generated. The authoring system is only rarely involved in this task (a few authoring systems include entries for dummy file names).

III) Content Collection

The content material is collected and entered into the authoring system. In general, this includes taking pictures, making a video clip and producing an audio sequence. When the existing content is available either from internal or external sources, no creation tools are needed. It may be necessary to use a conversion tool to convert external source formats into formats with which the authoring system works. If the author creates the content himself/herself creation tools are needed, such as word processing, paint and drawing software, image capture hardware and software, audio capture hardware and software and video animation hard ware/software. Some authoring systems have some of these features but with capabilities compared to stand-alone tools

IV) Assembly

The entire application is put together in the assembly stage. Presentation packages, for example, do their assembly while the author is entering the content for the various screens. Once the screens are defined and placed in order, the presentation is ready to run. The limitation of this approach is that the author has little or no chance to interact with the authoring system when the application includes a lot of interaction and very complex or dynamic screens, most authoring tools require details of the work, sometimes even programming. Most high-end authoring software packages, especially those which have a full authoring language, can be operated in a modular mode. In this case, a programmer can create customized modules that fit the specific needs of the particular application.

V) Testing

The created application must be tested. More sophisticated authoring system provides advanced features such as single-stepping or tracing the program flow.

Authoring tools are still platform-dependent, although they are closer to the goal of independence than editors or hypermedia tools. It is worthwhile to mention Kaleida Labs, a joint venture between Apple Compute and IBM for multimedia technologies. One of their major projects is ScriptX a universal language that will allow multiple digital platforms to play the same digital file without modification. Because different platforms have different features and capabilities, the ScriptX run-time environment includes dynamic adaptation that allows an application to query the environment of the current platform and decide in real-time how it can best present itself. ScriptX is fully object-oriented with the capability for the user to combine objects at run-time. Several authoring products are currently available which help to develop applications such as

• Information delivery applications can be developed using the authoring tools Media script OS/2 Pro (Network Technology Corp.), Icon Author (Unisys), Tool Book (Asymetrix / AOI), Author ware Professional, IBM's infodesigner2 etc

• Professional presentations can be developed using presentation authoring tools such as PowerPoint (Microsoft, Inc.), Freelance Graphics and Harvard Graphics. All the tools provide many features for enhancing the presentation by adding professional styles, images, graphics, audio, video, animation or charts.

• QuickTime movies and interactive projects can be created by movie authoring tools such as Movie Works (from Interactive Solutions. Inc.) Movie Works has several advanced capabilities: first, Movie Works allows the user to create and edit objects in text, sound and paint; second, the user determines the project's look and feel; third, the user uses the Composer component to integrate the objects into a scene, and to add animations and special effects (scaling and transitions); and fourth, scenes can be linked together, either sequentially or interactively, to create a project.

Approaches to authoring

Many approaches have been developed by designers and software engineers have developed *for* multimedia authoring. They are mainly aimed at reducing the development time through the provision design systems that are easily understood coupled with a variety of useful tools. Their methods are designed around metaphors. A metaphor is a figure of speech where the description of one object is used to describe another object. For example the designer can readily understand an authoring package. That uses a *'page turning'* metaphor, even although there are no physical pages that are turned.

1. Linear/ frame-by-frame metaphor

It is the simplest metaphor of all. Projects are designed to run in a linear fashion his means that a project has a beginning and an end and the viewer is intended to watch the project's contents unfold a page at a time, there is little or nothing the user control of the project, apart from perhaps pausing or exiting. This method is best suited to unattended presentations at exhibitions and in supermarkets and for some product promotions. Since the flow determined for the viewer, there is only a single route to be followed and every piece of screen activity follow the previous one in orderly fashion. The method is simple; the products are quickly assembled using basic authoring tools. There are a large of freeware and shareware packages available for producing these kinds of projects and PowerPoint is the most common commercial product used for linear productions (although it can also of some limited interactivity).

2. Programming

Programming requires scripts of code to be written by the developer. Early languages required complex scripts draw a box on the screen, while modem programming languages such as Delphi. Visual Basic are visual, blurring the boundaries between conventional programming and multimedia authoring GUI-based development tools, such as Delphi and Visual Basic can bring in buttons, display graphics, and call up video and sound clip without resorting to heavy scripting. Visual Basic, for example uses *a* screen form which have a graphic image as a background on which objects and controls placed. The developer then sets the properties that should apply to each of the controls. Code is attached to a screen object and is only run when the object is activated. This makes it possible for some practical projects to be created without any coding. Some of the package provide for compilation facilities and comprehensive error checking and debugging facilities.

There are three types of scripting:

Scripting type	Example
Scripts to create me entire project	GL Pro is an authoring package that is entirely script Based, with non-screen authoring.
Scripts that supplement	Director's Lingo scrap, Flash's Action Script

authoring packages	
Internet advanced scripts.	JavaScript, Perl. etc.

3. Hypermedia linkage

This method covers the linking <of any number of separate resources that allow user access. The most common implementation is the web site, although CD's can use hyperlinks for navigation and for linking to wider resources through me Internet. This system produces the widest possible navigation facilities and the must likely methods for implementing these systems are HTML scripts, JavaScript, CGI scripts, and Perl scripts, etc

4. Card /scripting

These are also sometimes called *'page-based* systems. The project data is organized to simulate pages of a book or a stack of cards. Each page has its own individual set of objects and its own layout. It can display graphics or play sound animations and video clips on any page. It uses a book metaphor as its presentation format. Using navigation buttons, viewers can flick through the pages of the book in any order. This presentation method is best used when a number of page sequences are use, although viewers can allow jump between pages. Some application that use this metaphor include Tool book, Mediator and illuminatus used this method.

5. Iconic/flow control

This method makes the project's structure and links highly visible to the developer, as the main screens are represented on a flow chart. These flow diagrams display the navigational links and flow of ideas. This aids the development of projects with complicated structures, as a top-down approach starts from the higher-level tasks and breaks them into lower levels tasks. A flow chart plots the possible routes between activities providing a *navigational map*. An icon represents an activity, which could be a decision to be made e.g.

- a user entry to be required,
- a new screen of graphics to be shown,
- a video or sound clip to be run,

At the design and implementation stages, groups of icons can be grouped together under a single icon - implementing a top-down design of sub-modules. Major points can be added to the flow line, and these can be resumed to later, to flesh out their sub-units and contents.

Author ware and Icon Author are applications that use this met

6. Hierarchical Object

The Hierarchical Object paradigm uses an object metaphor (like OOP) which is visually represented by embedded objects and iconic properties. Although the learning curve is non-trivial, the visual representation of objects can make very complicated constructions possible.

7. Tagging

The Tagging paradigm uses tags in text files (for instance, SGML/HTML, SMIL (Synchronized Media Integration Language), VRML, 3DML and Win Help) to link pages, provide interactivity and integration of multimedia elements.

MULTIMEDIA PROGRAMMING VS MULTIMEDIA AUTHORING

It should be noted that a distinction should be made between Programming and Authoring. Authoring involves the assembly and bringing together of Multimedia with possibly high level graphical interface design and some high level scripting. Programming involves low level assembly and construction and control of Multimedia and involves real languages like C and Java.