

# Introduction to Authoring Tools

## Objectives

Describe the role of authoring software in multimedia development.

List different types of interactivity supported by most authoring tools.

Outline desirable features authoring software should include.

Describe the standard development metaphors used in authoring software, and discuss the advantages and disadvantages of each.

## Introduction

Recall that multimedia in the context of this course (and the program) is an interactive, computer-mediated experience that uses one or more media to communicate something. We have looked briefly at software used for editing images, and a quick perusal of any software catalog will reveal numerous products for manipulating sound, video and animation. However, the creation of digital media is not sufficient for creating multimedia. We must have a means of constructing the interactive interface that allows the computer to act as a mediator through which the user can gain access to the media. We must be able to produce a computer program that runs the show.

One approach to this would be using a powerful programming language such as C++ or java to write all the instructions necessary to create our product. But multimedia titles are large, complex programs, and writing them from scratch would be very time-consuming and expensive. Fortunately, when the possibilities of interactive multimedia first began to emerge, several clever individuals (and companies) realized there would be a need for software tools that would facilitate the production of multimedia titles. These tools are specialized for the purpose—one would not wish to write process-control or payroll programs with them. But *authoring tools* make the development of multimedia titles much simpler.

## Functions of Authoring Tools

The primary job of the authoring tool is to allow us to develop an interface, define the interactions allowed, and place the media we need where we want them. The production of the media elements is not the task of the authoring software, although most authoring tools do include simple drawing tools and may also include tools for creating simple cel-based or path based animation. Some authoring tools also include minimal sound recording and editing capabilities.

### *Interface*

Creating an interface with most authoring tools is fairly simple—we first decide what screen layout(s) we need, and how the users will be allowed to navigate through them. This is mainly a paper-and-pencil task, although many developers use a product like Hypercard to "draft" a prototype. Afterward, we can create the

individual screens according to our design, positioning buttons or some other kind of navigation artifacts to provide links to other screens. Built-in definitions of buttons, etc, and the behaviors they are allowed make this a fairly easy task (compared with programming from scratch).

Authoring tools support a variety of interactive capabilities. Interactivity is in essence a stimulus/response relationship, and one way we can categorize our options is by listing stimuli the user can provide and then the responses we can make happen. Here are some examples--you may be able to think of more.

#### Stimuli:

- click a button, "hot spot," object, or text
- press a key or key combination
- drag something to a target zone
- type something in an entry field
- allow some time to elapse

#### Responses

- Go to another screen (with a specified transition)
- Display an image or movie
- Play a sound
- show an animation
- launch another application

The above tend to be easy to make happen. In addition, we can use the scripting language included in authoring tools to provide a wide variety of other actions.

#### *Navigation Schemes*

There are many elaborate ways to categorize navigation schemes, but a fairly simple list is sufficient to get the general idea. Sequential navigation is like a slide show, allowing the user to move forward or backward through a sequence. Hierarchical navigation provides links to specific sections of the title, and then to subsections, so that the user may choose when to view the material for a given section even though random access to every single screen may not be possible. This type of navigation is suitable for giving access to structures, orderly information. A third type of navigation is associative navigation, which includes links to screens that are related in some way to the current screen, but not as a continuation of the same thought (as the next screen in a sequence would be ) or as a generalization or dissection of the current idea ( as would be the case in hierarchical navigation). Associative links result in web-like networks—like the World-Wide Web.

Most multimedia titles incorporate a combination of navigation, depending on the content.

## **Authoring Tool Features**

### Building the Interface

- Specify screen dimensions and background color
- Create links that go to "next" and "previous" screens
- Create links that go to non-adjacent screens
- Specify transitions
- Define button appearance, icon, style
- Make invisible buttons (hot spots)
- Draw simple screen elements

### Media handling

- import sound, graphics, video so they become part of the executable
- access sound, graphics, and video as external files at run time
- synchronize different media

### Advanced Features

There are some features of authoring tools that frequently go unused, either because they are not obvious or employing them demands more expertise. These features provide the ability to make large, powerful multimedia titles. A list of some such features follows.

Allow scripts to be associated with certain events to support more sophisticated product behavior. An example here is reading and writing text files.

The ability to load media files at runtime and the ability to read and parse text files makes it possible to create relatively small executables that essentially consist of templates for every type of screen. The code loads databases of text that provide details about how to construct each screen, and the necessary media are loaded as needed. Revising this type of product is much simpler than changing one that is a large executable with every screen "hard-coded."

All the authoring software we look at in this class is capable of almost every feature described above. Some make certain things much easier than others, and there are a few tricks unique to each package--such as Authorware's "drag-and-drop" interaction. However, the production metaphors for these products vary widely.

## **Preview of Production Metaphors**

The task of creating interactive multimedia can be looked at as "just writing another program." However, the use of some kind of metaphor for the product helps to reduce brain strain when dealing with the multiple dimensions of activity. Three effective metaphors are the card (or book) metaphor, the flow-line metaphor, and the timeline metaphor.

HyperCard, released in 1987, provided the first use of the card metaphor in authoring. The tool, made for the Macintosh computer, models the product being created as a stack of index cards. Buttons can be placed on a card (or in a

background layer shared by one or more cards) and given tasks. The tasks might include going to another card, playing a sound, showing a video clip or combinations. More sophisticated jobs can assigned using the scripting language, HyperTalk.

On-campus students have looked at few examples of simple Hypercard titles in class. The fact that several other products have since used the card metaphor suggests that is a reasonably effective one. Supercard, Digital Chisel, HyperStudio, ToolBook and Revolution DreamCard all use(d) a similar approach.

Authorware employs a very different approach. Its metaphor is like a programmer's flowchart. The developer drops various icons onto the flow line to diagram the products behavior. Icons may specify that an image be added to teh display, that asound play, that a video play, or that a screen with a set of buttons that dictate various interactions appear. Authorware also includes the ability to attach a "calculation" or script to any icon to increase functionality. A calculation icon can be dropped on the flow line at any point that further instructions are needed.

Director uses a "timeline" metaphor. While HyperCard's output is a stack, and Authorware's output is a "package," director creates "movies." The idea is that the programmer is a "director" telling various "cast members" (buttons, images, etc) when to appear "on stage." The timeline is represented as frames of film.

The metaphor for director is perhaps the least suitable for assisting in the development of non-linear products, since a movie is by nature a linear, non-interactive experience. On the other hand, synchronizing events is easier in Director because the cast members representing the elements to be synchronized need only be brought in at the same frame. Director's domination of the market for authoring tools suggests that it has a great deal going for it, and although *I* am troubled by its inherently linear metaphor it is at present the standard for most multimedia development.

## **Conclusion**

In conclusion, Authoring tools are specialized software packages that are used to create interactive products with graphical user interfaces. They are designed to simplify the task of creating the interface, and although this limits the type of products one can create with them, they are perfect for creating interactive multimedia.

Although the development metaphors adopted by different authoring tools vary widely, the task accomplished by each is essentially the same: provide a simple way to construct an interactive interface and programming tools to give the interface more complex tasks.