Chapter 17
Visual Support for Authoring

Rob Wright
Barry Harper
John Hedberg
University of Wollongong

Keywords: Authoring tools, Constructivism

Abstract

Development of educational software has had a long history of use of authoring environments that have enabled instructional technologists rather than programmers to design and develop applications. The advantage of these tools has been that the designer did not need to be highly skilled in high-level languages, but could use a simpler construction set of pre-programmed modules, often supported by a simple scripting language. The disadvantage was that the developer was limited to the pre-programmed modules available and to the underlying assumptions of the structured instructional design models adopted by the tool. The designer also had to work within the visual and procedural structures employed by the tool designers to represent the design process within which learning activities could be constructed.

To simplify terminology amongst authoring environments, in this paper we will refer to the authoring application as a "tool" and the resources, which are collected together for display as a "project". Several authoring tools can construct standalone "projects" (for example, Director creates a "projector") which can be distributed to learners individually. Some tools also focus upon record keeping aspects of the system and create student files that are the student’s responses to the embedded questions. Some tools work directly with the visual display, such as Digital Chisel and Oracle Media Objects, all changes and design decisions are represented as visual changes to the display. Other tools provide a more comprehensive meta-view, for example, Authorware, shows the flowchart of the algorithm being employed as well as the runtime display window. Apple Media Tool shows a graphic view of the screens and the associated links, and mTropolis represents a meta-view as a hierarchical "finder"-type structure.

1. VISUAL METAPHORS IN PRACTICE

No matter what authoring tool is used, software designers have to work within the prevailing metaphor of the tool. For example, if a designer employs Director, the prevailing metaphor is a theatrical pageant and is conceived as a time dependent display. Events occur on a stage and are managed by a score, which dictates the depth and movement of each object over time. By contrast, HyperCard employs a card metaphor where links between discrete representations of objects are embedded in an Hypertext relationship. Icon Author and Authorware create a complex flow chart
(algorithmic) structure, which is used to design the experience but is not displayed to
the learner as they work through the pre-determined paths that are reminiscent of the
traditional concepts embodied in programmed instruction.

From these brief examples it is clear that the visual metaphor will create a vehicle
in which the instructional designer must work and conceive their project. It follows
that there will be a symbiotic relationship between the instructional strategies and the
way the tool enables the designer to think about the task. Of the many common tools,
the older have taken a more structured approach borne out of behavioural learning
theory. The more recent tools have striven to reduce the need for time or procedural
structure to create an environment of intelligent objects. To represent this process the
tool authors have created different ways of changing and visualising the relationship
between creation of a learning environment and using these environments. Thus
options which will enable time, hierarchical and spatial display are all possibly needed
by designers as the learning tasks and project demand.

2. DEVELOPING AN AUTHORING ENVIRONMENT

The designers of an authoring environment must make assumptions about the
instructional design models that the tool will support and the potential end users.
Some tools are designed for developers and others for learners to construct their own
ideas. Several key writers have called for a reassessment of instructional design
models used for the development of technology-supported learning environments that
assume constructivist views of learning. Hannafin and Land (1997) have suggested
that we should be aiming for open-ended learning environments, Jonassen and
Tessmer (1996-1997) argue that we should be aiming at new learning outcomes, and
Duffy and Cunningham (1996) have described a range of metaphors which structure
how we teach. Additionally, Savery and Duffy (1996) have elucidated several
principles that characterise this philosophical view in technology-based learning
environments.

Jonassen and Tessmer (1996-1997) have also questioned the commonly-used
taxonomies of learning that are the basis of our instructional design models, proposing
that engagement with a greater range of learning outcomes is essential for meaningful
learning. They have suggested a new framework for specifying the types of learning
outcomes modern learning environments should be developing.

In response to these design suggestions and the changes that have occurred in
hardware technology options, we have proposed a design model (Figure 1) that is
cognisant of constructivist approaches to instructional processes and addresses many
of the above suggestions for reassessment of instructional design models (Hedberg et
al., 1994).

Given the changes to hardware technology options, we wanted to provide design
frameworks and visual metaphors that could be employed by highly skilled designers
to use the tool for project creation. We also wanted a tool that could support young
learners, with access to rich resources, to construct representations of their own ideas
using such a tool.

Phase one of the model takes the basic information derived from an assessment of
needs and describes the parameters of the Project space. This is the information
which is to be included in the materials, how it is structured, what the target audience
understands about the information and how it might be structured for the audience. A