THE MENTOR PROJECT: FROM CONTENT TO INSTRUCTION

Terence C. Ahern[1] *and Nancy Van Cleave*[2]

Abstract — *Instructional technology is rapidly expanding into all levels of our society from a wide variety of government and educational institutions to corporate enterprises. We currently have unprecedented access to information and resources, yet with all the advances in technology and profusion of web sites, there is no standard on how to transform this deluge of content into instruction using technology. Specifically, there is a lack of direction on how to effectively sequence individual learning objects (reusable digital resources) that makes sound instructional sense. Before the potential for computer managed instruction can be completely realized, instructional design must be incorporated into instructional technology. The Mentor Project is designed to explore this issue by mediating between learning objects and instructional sequencing. In this paper we present an XML meta-grammar to categorize learning objects in meaningful human terms, an XML meta-grammar for lesson sequencing, and a prototype of the sequence editor, a course authoring system.*

Index Terms — instructional design, learning objects, lesson sequence, learning technologies.

Introduction

Instructional technology is at a crossroads. Rapid development of computer technology has provided unprecedented access to information and resources that only a generation ago was the stuff of science fiction. According to a study by Becker [1], there are over 10 million computers available in K–12 schools, 98% of which have Internet access. Online access for higher education has also mushroomed, with student enrollment in online courses during the years 1995–1998 more than doubling, as the creation and dissemination of information also kept pace [2]. A recent industry report published by A.T. Kerney [3] notes organizations around the globe lose approximately $750 billion annually due to "wasted time spent by knowledge workers seeking and capturing information necessary for them to do their jobs." It further states that the proliferation of content has exploded so that "the number of Web pages, which stood at five billion in 2000, is expected to increase to over 40 billion by 2003—translating to roughly six and a half Web pages per living person on the planet."

[1]Terence C. Ahern, ICST, California State University, Monterey Bay, terence_ahern@csumb.edu or College of Education, Texas Tech University, terenceahern@ttu.edu

[2]Nancy Van Cleave, Eastern Illinois University, cfnkv@eiu.edu

Even with this plethora of available technology and content, Becker (p. 2) reports that the areas of "...acquiring information, analyzing ideas, and demonstrating and communicating content understanding... involves computers significantly in only a small minority of secondary school academic classes." Obviously there are still obstacles to overcome in order to utilize fully the power of computer managed instruction.

A major problem for instructional technology is how to transform all this content into *effective* instruction. Two critical subproblems which must be addressed are:

1. How to quantify, classify, and establish relationships between units of information, and
2. How to utilize **instructional design** to best present these packets of information to the learner for the highest rate of understanding and retention.

The first of these challenges is represented in remarks made recently by a student teacher studying science. "If we find a web site that has great information about both the solar system and living/nonliving things, can we list it more than once? Is there a way to list sites according to both subject matter/TEKS [TX Essential Knowledge and Skill] relation and grade level?" [4]. Learning objects, typically defined as any digital resource that can be reused to support learning, have been suggested as providing a possible solution to this problem. The Learning Technology Standards Committee (LTSC) of IEEE was formed in order to develop the guidelines necessary to support the implementation of the learning objects approach [5]. Their purpose was to establish instructional technology standards with the intent to create environments that "automatically and dynamically compose personalized lessons" [6]. However, there is within the proposed standards a general lack of direction on just how this is to be accomplished, because "no one had considered the role of instructional design in composing and personalizing lessons" [7].

Which leads to the second challenge, involving instructional designers in the development of courseware and other computer facilitated instruction. If we all learned in the same manner with the same rate of understanding and retention, then we could simply make textbooks available online and call it electronic instruction. However, there is a wide variety of instructional strategies and criteria for their application, all of which must be considered if we are to succeed in creating effective electronic educational tools. Learning objects must be combined "in a way that [makes] instructional sense, or in instructional design terminology, 'sequencing' the

0-7803-7444-4/02/$17.00 © 2002 IEEE

learning objects" (ibid., 11).

The MENTOR Project is designed to explore exactly this issue by mediating between learning objects and instructional sequencing. In this paper we will discuss learning objects, sequencing, and instructional design further, followed by the presentation of:

- An XML meta-grammar to categorize learning objects in meaningful human terms
- An XML meta-grammar for lesson sequencing — the output of the sequence editor
- The prototype for a sequence editor — a course authoring system which creates and links *learned objects* to form a lesson

Instructional Design

Experience is our first teacher. A baby learns the concept of hot through a process of simple trial and error. In contrast, an adult might employ more complex metacognitive strategies that nonetheless rely on previous experience [8]. Experience, although it can be very effective, is usually a very inefficient teacher. Instruction, on the other hand, is a systematic process that attempts to remedy a deficiency or to overcome a barrier in a student. It does not wait for experience but attempts to anticipate the learner's need.

An instructional system has three major elements: student, teacher, and content. The system also manages three major interactions: content–learner, teacher– content, and learner–teacher, as illustrated in Figure 1.

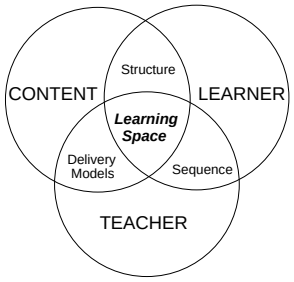

FIGURE 1. Instructional System

Content organization is directly dependent on the student. For example, if we were teaching young children about polygons, a good structure might be to approach the content by using concrete examples that will lead to the more abstract concepts. In other situations, such as with adult learners, a more abstract structure that starts with an abstract definition may be more appropriate.

Within the instructional system, the teacher/ designer has two extremely important roles or functions. The teacher/designer must be able to design the most appropriate delivery model given the content. The choice is not monolithic, but depends on the type and organizational structure of the content. Joyce and Weil [9] have cataloged a variety of teaching models and grouped them into four major families: (1) behavioral, (2) information processing, (3) social interaction, and (4) personal source. They provide developers with a very useful source to categorize delivery methods.

Consider our polygons. If the content starts with the concrete, we may want the children first to find examples of squares or triangles within their immediate environment. Next we might have them define the polygons by naming them. Later on when we want the students to classify different types of polygons by shape, the concept attainment model found in the information processing family would be a better choice.

Additionally, the teacher/designer is also responsible for the creation of an effective sequence of learning experiences that matches the content with the developmental level of the learner. Sequencing is defined here as "the order in which elements of subject matter, including information, skills, and cognitive strategies are taught during instruction" (English, p. 23). The teacher not only selects but also must regulate the sequence by pacing the learning experiences most effectively for the learner. This is a critical factor, and the teacher must constantly evaluate the readiness of a student. Given specific content structure, the teacher must judge whether or not the student is ready to proceed to the next target within the learning sequence.

The individual elements of content, teacher, and student come together to form a learning space, defining the lesson environment. Time is the primary constraint on the ability to deliver instruction within this learning space. The teacher, which could be an automated system or a human, mediates between the learner and the content by managing the delivery of the material in an appropriate sequence. In real time systems, the teacher is free to adapt to the changing environment by speeding up or slowing down the delivery of material and exercises. Within automated environments, this is much more difficult to accomplish.

Designing Mentor

The complete `Mentor` project will incorporate instructional design into a course authoring and management system. Conceptually, the teacher/ designer mediates between the content, learning objects, and students within a learning space. The name `Mentor` was chosen to represent this process. Further, the interface for the sequence editor was developed to maintain the notion of the learning space and mediation as the fundamental metaphor.

The complete system will be composed of several

components:

- Content Editor – to create meta–learning objects
- Learning Space Sequence Editor – to create *learned nodes* (from meta–learning objects) and link them through a *learning space map*
- Mentor – to manage student progress through lessons consisting of learning space maps and learning objects

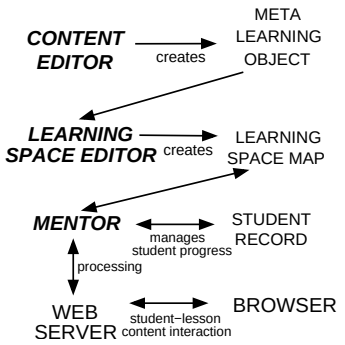

FIGURE 2. The `Mentor` project

An XML document type definition for the meta–learning object will link the content editor output with the sequence editor. Another XML DTD for the lesson sequence map then links the output of the sequence editor with the `Mentor` application.

Content Type Definition

Learning objects are typically viewed as reusable digital instructional resources, but are very widely defined (Wiley, pp. 2–7). One popular conception describes a learning object as containing both *content* and *delivery method* (Figure 3). According to Koper [11], this view of a learning object is consistent with the IEEE definition.

In general, this view of learning objects is insufficient for the purposes of sequencing a lesson since it doesn't provide enough useful information — specifically, where the learning object fits into a particular lesson structure. Furthermore, we need information about the type of delivery model the object requires in order to insert it more effectively into an appropriate lesson sequence.

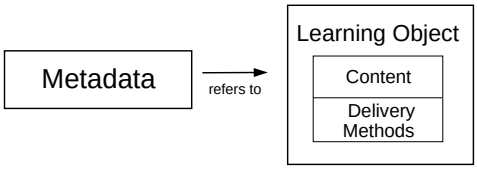

FIGURE 3. (Meta)–Learning Object

According to Forester Research [12], "XML–defined metadata (data that defines data) can embed structured information within data streams that can be used to discern the underlying meaning of the searched material... [It] will facilitate more efficient tagging of content." Therefore, we created our own XML Document Type Definition (DTD), (Figure 4), created by the content editor and used within the sequence editor, as a grammar defining the legal syntax of the content of a lesson.

```
<!ELEMENT source (content)>
<!ELEMENT content (name,models)>
<!ELEMENT models (direct,cognitive)*>
<!ELEMENT direct (method)*>
<!ELEMENT cognitive (method)*>
<!ELEMENT method (type,name,data)*>
<!ELEMENT type (#PCDATA)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT data (#PCDATA)>
```

FIGURE 4. Lesson Content Document Type Definition

The Lesson Content DTD includes information about the type of instructional method (direct or cognitive) embodied in each component of the lesson, as well as the type, name, and location of that component. An individual developer can quickly identify different learning objects from a variety of vendors and insert them in the lesson sequence in the appropriate order. This expedites the inclusion of these objects (e.g., `URL`s, formatted text documents, etc.) within the design space.

```
<!DOCTYPE source SYSTEM "content.dtd">
<source>
 <content>
  <name>Sentence Structure</name>
  <models>
   <direct>
    <method>
     <type>nominal</type>
     <name>Noun definitions</name>
     <data>http://faculty.csumb.edu/noun_defs.html</data>
    </method>
    <method>
     <type> nominal</type>
     <name>Verb definitions</name>
     <data>http://faculty.csumb.edu/verb_defs.html</data>
    </method>
   </direct>
  </models>
 </content>
</source>
```

FIGURE 5. Lesson Content DTD—Example

By utilizing this definition for a lesson, we enable instructional designers to incorporate reusable learning objects from within a content discipline, then create an index file from the Content DTD (produced by the Content Editor). Figure 5 illustrates an example output file resulting from a content editor session. This file contains specific learning objects which the designer wishes to group together and include in a particular lesson. It can then be used by the sequence editor as a building block in a larger lesson.

The example demonstrates the content for an English lesson on sentences, and contains the description of two

basic elements within a sentence—nouns and verbs. Notice the content description describes the delivery method and the links to the location of the learning objects. Mentor needs this information in order to create dynamic delivery information within an appropriate learning sequence. In this example, both of the objects were simple web sites, but the sequence editor is capable of utilizing any type of learning object as long as it is accessible.

Sequence Editor Output

Using the instructional system as our guide, we designed the lesson sequence DTD that defines the resulting output from the sequence editor (Figure 6). The resulting output file associated with this document type definition will in turn be used as input to a course management system which will use the inherent sequencing within the document to automate presentation of the lesson to students.

```
<!ELEMENT StateMap (lesson)>
<!ELEMENT lesson (name, content)>
<!ELEMENT content (node)*>
<!ELEMENT node (ID,location,name,description,
                stategoal,methods,targetStates,
                criteria)*>
<!ELEMENT location (x,y)>
<!ELEMENT x (#PCDATA)>
<!ELEMENT y (#PCDATA)>
<!ELEMENT ID (#PCDATA)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT description (#PCDATA)>
<!ELEMENT stategoal (#PCDATA)>
<!ELEMENT methods (line)*>
<!ELEMENT line (#PCDATA)>
<!ELEMENT targetStates (stateNode?)*>
<!ELEMENT stateNode ((source,sHndl,target,tHndl))>
<!ELEMENT source (#PCDATA)>
<!ELEMENT target (#PCDATA)>
<!ELEMENT sHndl (#PCDATA)>
<!ELEMENT tHndl (#PCDATA)>
<!ELEMENT criteria (line?)*>
```

FIGURE 6. Lesson Sequence DTD

The sequence editor output DTD defines the lesson sequence map for engaging the student. A major features of the DTD is a *state* or *learned node* which defines some general metadata available only to other instructional developers. This metadata includes such items as the node's name, a general description of the content, and the goal for this particular state node. The state node describes all the necessary learning experiences required for a learner within the specific content sequence.

The state node also provides a mechanism for linking the specific learning objects together in a particular order. These objects can be web sites, local documents, or other types of learning objects. Finally, the state node also identifies its own exit criteria, which is the required student performance established by the teacher/designer for the particular lesson. The sequence editor allows multiple target nodes with different performance requirements.

The Sequence Editor Prototype

Educational authoring software developed to facilitate the creation of electronic lessons should be based on the following simple design principles [10]:

1. Utilize a cross–platform, open system model, combining minimum cost with maximum flexibility
2. Provide an easy to learn and use graphical interface
3. Follow the instructional model system discussed in a previous section

To incorporate these principles in the lesson sequencing component of the `Mentor` system, we selected `Java` as our programming language. Current versions of the `Java` virtual machine and development environments are stable and run on multiple platforms with little or no need for revision.

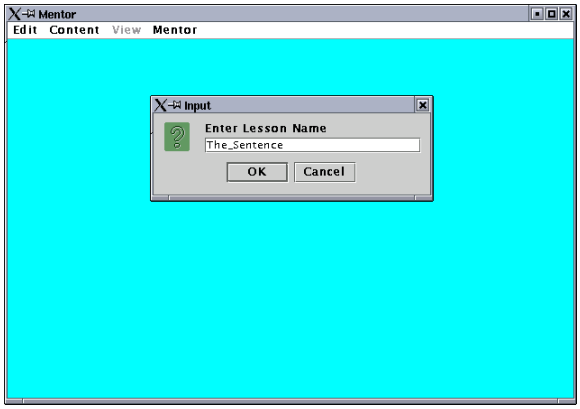

FIGURE 7. Creating and Naming a Lesson

The prototype was designed with the underlying instructional model in mind. When a developer starts up the sequence editor, they are faced with a blank desktop. To begin a new lesson, the designer pulls down the first menu item under `Edit` and selects `New Lesson`. A dialog box pops up and prompts the designer for a lesson name as illustrated in Figure 7.

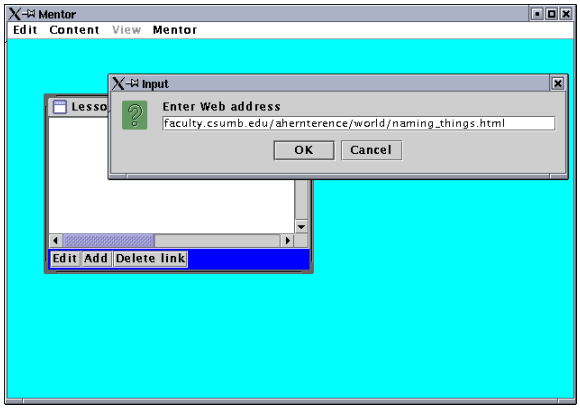

FIGURE 8. Learning Space Window and Content Dialog

Once the designer presses the enter key, the learning space window appears. After the lesson has been initialized, the designer/teacher/developer can register learning objects that are both content and vendor specific. To register content specific learning objects, the designer selects the pull–down `Content` Menu. The system prompts the user for a file name or `URL` address (Figure 8). The sequence editor will read the file and parse it according to the name and delivery type. Additional information can be entered and associated with this node (Figure 9).

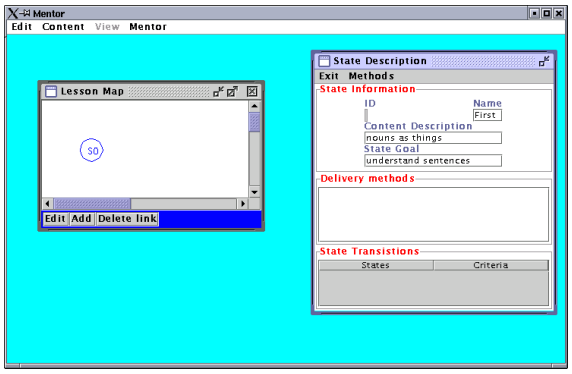

FIGURE 9. Creating a Learned Node

Once the editor has interpreted the content file, the pull–down `View` Menu becomes active. If it is selected, the designer is presented with a cascading menu consisting of registered learning objects. In addition, the `View` Menu allows the user, if they select an item, to view the specific learning object in another window.

By registering the learning objects within the `Content` Menu, `Mentor` provides the designer with access to the learning objects within each of the individual state nodes of the lesson.

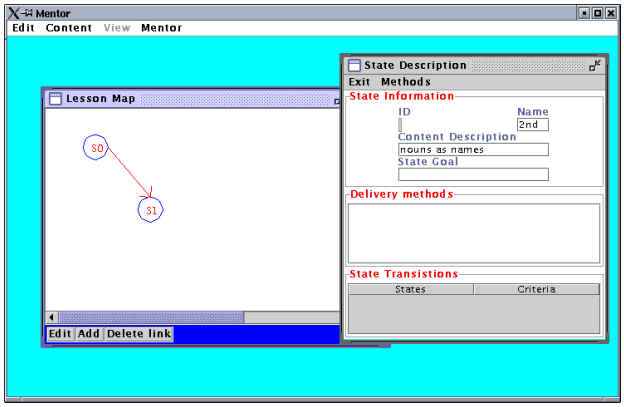

FIGURE 10. Sequencing Learned Nodes

Now the user can proceed to develop the lesson sequence and tailor it to individual learner performance. To create a state node, the designer clicks the `Add` Button located on the button bar within the lesson map window. The designer can position the state node by double clicking the mouse within the learning space.

The system then draws a node represented by a circle within the space and pops up the `State Description` window, as illustrated within Figures 10 and 11.

Within the `State Description` window, the sequence editor automatically enters the node number assigned to the learning object just entered into the lesson map. The designer names this particular `State Description` and enters information concerning the vendor or content for the specific learning objects.

Further, the lesson designer indicates the intended general outcomes for this particular state node. This information may be used by other lesson developers or teachers when this lesson is made accessible or shared.

The user has total control over how the learning objects are sequenced within the `State Description`. Based upon the learning objects available under the `Methods` Menu within the `State Description` window, the designer creates a list of learning objects and experiences within the lesson methods (see Figure 11). Other objects that are not represented under the `Methods` Menu can also be accessed and inserted.

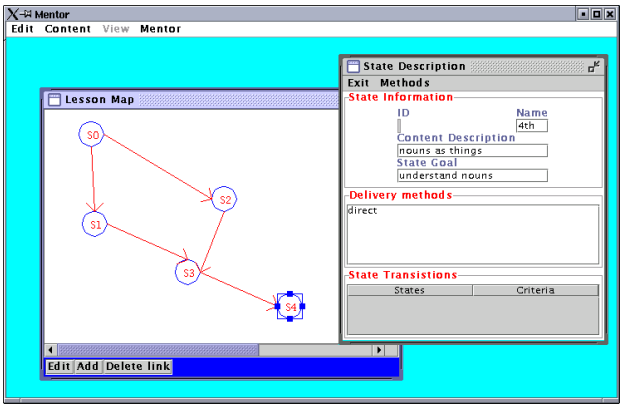

FIGURE 11. Sequence Map

The developer provides a sequence of nodes built around individual student performance within each state node. As illustrated in Figure 11, the designer has created five state nodes with various links connecting *source* to *target* nodes.

The learning state sequence editor allows the designer to easily develop and provide multiple learning paths through the specific content structure of a lesson. The system tracks the number of state criteria required for each of the target links within the learning space. These alternative paths can provide the learner with additional content or exercises, if warranted by the individual student's performance. Thus, if the student's performance in a state is less than satisfactory, the designer can

provide automatic alternatives for the learner. A specific student would then be directed to any one of possibly several `Review` states based upon their level of achievement. These `Review` states are intended to be remedial in nature, providing a transition for the student to the desired level of understanding. In this way, the system can accommodate multiple learners and multiple learning styles and still remain based upon learner performance.

The goal of the editor prototype was to mimic as closely as possible the mediating aspect of lesson design. The user interface approaches this as a design metaphor. The design space allows great flexibility in how the designer will lay out the lesson in terms of each state description, as well as the global issue of learner state transitions and sequences. The designer has control over the sequencing of the lesson, but leaves the pace up to the individual learners to decide through lesson performance.

Future Refinements

The learning state sequence editor represents a single application within the `Mentor` Project. One component yet to be developed is the content editor, which will allow content vendors to categorize specific learning objects by the intended delivery method. Along these same lines, issues of middleware and course management need to be addressed. `Mentor` itself, the presentation and management application also has not been implemented at this time.

Formal usability studies are planned for the `Mentor` Project. Anecdotal feedback from educators indicates that a simple mechanism for creating transition objects within the state descriptions appear to be very useful.

Summary

The growth of content development and dissemination has outstripped our ability to design and implement computer managed instruction. The IEEE has recognized this and created the LTSC to address this new need. Learning objects have been suggested as a possible solution. However, specific implementation guidelines are lacking. The `Mentor` Project was conceived as one way to incorporate instructional design into instructional technology using the open source paradigm. The learning state sequence editor presented in this paper investigated the feasibility automating the sequencing of automated instruction. This editor follows a specific instructional system and attempts to mediate the development of an automated lesson composed of learning objects and their relationship to one another within the lesson.

The result is a state diagram, where nodes represent one or more learning objectives and include one or more learning objects to help the student attain the particular objective. One or more nodes will be designated as `final` states, and they will contain the criteria for exiting from the lesson with satisfactory achievement. There may be multiple paths from the `start` node to `final` nodes, allowing the learner some flexibility by selecting their own path through the diagram. If at any node the student fails to meet that node's criteria (as set by the lesson designer), they can be routed through alternative nodes in order to review or reinforce their understanding of background material. This provides for a transition in the student from unsatisfactory to satisfactory performance.

References

[1] Becker, H.J., "How Are Teachers Using Computers in Instruction," presented at the American Educational Research Association (AERA) Annual Meeting, Seattle, WA, 2001.

[2] Lewis, L., E. Farris, K. Snow, and D. Levin (1999), "Distance Education at Postsecondary Education Institutions: 1997-98," National Center for Education Statistics, Office of Educational Research & Improvement, U.S. Department of Education, Washington DC [Online]. Available: `http://nces.ed.gov/pubs2000/2000013.pdf`

[3] Kearney, A.T. (2001), "Network Publishing: Creating Value through Digital Content," A.T.Kearney: Santa Clara, CA [Online]. Available: `atkearney.com/pdf/eng/Network_Publishing_Study_S.pdf`

[4] Personal Correspondence

[5] Learning Technology Standards Committee (2001, April), IEEE Standards Board: Project Authorization Request (PAR) form, [Online]. Available: `http://standards.ieee.org/guides/par/`

[6] LTSC, Learning Technology Standards Committee (2001, April), web site, [Online]. Available: `http://ltsc.ieee.org/`

[7] Wiley, D.A., (2001) "Connecting learning objects to instructional design theory: a definition, a metaphor, and a taxonomy," Utah State University, Digital Learning Environments Research Group, The Edumetrics Institute. [Online]. Available: `http://reusability.org/read/chapters/wiley.doc`

[8] English, R.E., and C.M. Reigeluth, "Formative research on sequencing instruction with elaboration theory," *Educational Technology Research and Development* (ETRD), **44**(1), 1996, p. 23-42.

[9] Joyce, B., M. Weils, with E. Calhoun, *Models of Teaching*, Allyn & Bacon Publishing Company, 2000.

[10] Ahern, T., N. Van Cleave, and B. York, "Open Protocols for Web-based Educational Materials," *Frontiers in Education Conference*, 2001.

[11] Koper, R., (2001, June) "Modeling units of study from a pedagogical perspective: the pedagogical meta-model behind EML," Educational Technology Expertise Centre, Open University of the Netherlands. [Online]. Available: `http://eml.ou.nl/introduction/articles.htm`

[12] Forester Research (1999, January), "Guided Search for eCommerce," referenced in [3].