USING A HYPERTEXT INSTRUCTIONAL DESIGN METHODOLOGY IN ENGINEERING EDUCATION

Susan A. Mengel
Texas Tech University
Computer Science
Box 43104
Lubbock, TX 79409-3104

William J. Adams
Room 1115, Building 602
D/EECS, USMA
West Point, NY 10996

Marion O. Hagler
Texas Tech University
Electrical Engineering
Box 43102
Lubbock, TX 79409-3102

Abstract -*Inherent in good engineering is the practice of using a design methodology when constructing complex systems, whether they are bridges, circuits, or computer programs. The design methodology can help to work out problems before they make their way into the final product. Also, the design methodology can help to make all of the parts of the final product fit together better and thereby achieve a certain coherence in the design.*

A design methodology can be used to achieve similar coherence in building hypertext instructional systems. The design methodology described in this paper can be easily used by engineering educators and graduate students without special training after reading a description of it and working through an example application. The methodology uses the object-based paradigm, has checks for validity to help the user detect unreachable hypertext nodes, and incorporates sound instructional design principles. The value of the methodology is in making it easier for the user to organize instructional material carefully without having to spend a large amount of time in learning a complex design process and to incorporate components already developed that have been used successfully.

INTRODUCTION

The goal of the hypertext instructional design methodology in this paper is to demistify, simplify, and codify hypermedia product development, since, even with the hypermedia authoring tools available on the market, the task of organizing and assembling a hypermedia training resource can be overwhelming. For example, hypermedia training products usually are created by teams of highly paid specialists working with expensive hardware and software although subject matter experts who have an intimate understanding of the subject at hand, but not necessarily of computer programming or of graphics and multimedia design, may wish to create their own training products.

Because the subject matter experts may not be multimedia experts, they could benefit from the use of a methodology which supports sound instructional and software engineering principles. For instance, the methodology could point them toward previously successful instructional and multimedia techniques for computer-assisted learning, making it unnecessary for them to discover as much on their own. They could learn how to design, implement, and test their system to reduce painful reengineering of previous work to achieve future improvements.

The hypertext instructional design methodology described here and used at the United States Military Academy and Texas Tech University consists of existing techniques integrated into a software engineering life cycle that focuses developers on the issues of reliability, maintainability, extendibility, usability, and reusability. The methodology is purposefully kept relatively easy to learn and use, since multimedia design packages, copyright issues, and subject matter organization present a considerable challenge to authors, as it is.

BACKGROUND

Because of the difficulty in authoring instructional systems, much of the literature in instructional system design presents work in progress (that may not have been applied as of yet), suggestions as to how to utilize authoring systems better, and help on how to approach multimedia design. Related work also occurs in the field of intelligent tutoring systems, usually in the form of instructional design advisors. A complete methodology for helping educators go from the instructional material into the multimedia system has not been given as a software engineering life cycle process although bits and pieces of such a process do occur in the literature and can be combined as suggested in this paper. Typical of the papers are those presented below which have good advice or systems to offer, but do not achieve a full software life cycle process.

Ehrlich and Reynolds [9] give a general discussion of the key issues on which one should concentrate when designing multimedia systems. They focus on learner characteristics, selection of topics and tasks, objectives, performance assessment, instructional activities, selection of media and delivery systems, resources, multimedia system revision, and multimedia system evaluation. They point out that because it is difficult for one person to act both as

programmer and instructional designer when implementing a multimedia system, creating the system should be a team rather than individual effort. Although they do not advocate or develop a specific methodology for developing multimedia systems, they give a thorough set of questions to consider during revision and evaluation. These questions easily could be used to drive the requirements document for the design of a multimedia system. The question categories include needs and goals, learner characteristics, topics, tasks, objectives, performance assessment, instructional activities, media, delivery systems, and resources. The questions focus on assessment of the multimedia system, mastery level required of the learner and geographic location, and objectives of the instructional materials.

Hardman, van Rossum, and Bulterman [11] have developed the CMIF (CWI Multimedia Interchange Format) authoring environment which focuses on the design and implementation of a multimedia system. The authoring environment is essentially a CASE (Computer Aided Software Engineering) tool which facilitates the building of a set of nodes for a hypertext multimedia system. It allows nodes of the system to be executed in serial or parallel format and permits media items, such as text, still images, audio, and video, to be attached to the nodes. It allows the media to be edited by appropriate editors within the authoring environment. It allows two views of the multimedia system, termed hierarchy and channel. The hierarchy view supports editing of the presentation structure in both a bottom-up and top-down manner. The hierarchy view implicitly imposes timing constraints on nodes to be played in serial or parallel fashion. Additional constraints may be added through synchronization arcs in the channel view, which shows both the logical mapping of nodes to media and the timing relationships. In addition to the two views, the player component of the authoring environment controls the running of the presentation and allows the user to select the channels to be played. The authoring environment does not integrate principles of instructional design or suggest how to begin design of the material, but offers a flexible system for creating multimedia presentations.

HYPERTEXT INSTRUCTIONAL DESIGN

The design methodology presented here (described more completely, with examples, in [1, 12]) is an eclectic blend of several disciplines, but at its base is the object-based paradigm of software engineering. It encourages designers to address, systematically, the educational and cognitive parts of the hypermedia project.

The methodology has three stages: requirements, specifications, and implementation. The requirements stage enables the designer to focus upon the key concepts to be presented in the material and produces a graph of the overall structure of the system. The specifications stage then successively refines the graph into a hyperweb structure. The implementation stage implements the graph in the chosen hypertext system. In each stage, validation and verification are used to ensure the key concepts have been presented effectively.

Requirements Stage

The requirements stage focuses on determining the scope of the system and entails extensive analysis of the subject matter to decide what information should be included in the system and how the concepts will be grouped and linked together. To facilitate this stage, Object-Oriented Text Decomposition (OOTD) [17] is used as a systematic method for decomposing subject material into conceptual components. OOTD uses natural language to communicate the information's structure and relation. It utilizes top-down decomposition methods and concept-mapping applications of Ausbel's learning theories [14] to produce a hierarchy that represents the information in a modular form. The steps are as follows:

1. State the thesis of the tutorial in a single sentence.
2. Expand the sentence into a high level paragraph.
3. Identify key concepts in the paragraph.
4. Identify the relationships between key concepts and the paragraph theme.
5. Review and revise.
6. Repeat steps 1-5 as needed for each new concept.
7. Encapsulate related concepts in a single-object module.
8. Implement relationship links in hypertext.

Each key concept is expanded into its own thesis sentence and paragraph and linked to the parent paragraph via its relationship with the parent. All key concepts are decomposed as far as desired and are connected into an OOTD graph according to their relationships. The OOTD graph gives the overall linking structure for the hyperweb. Each node in the OOTD can itself be a hyperweb as well.

Specifications Stage

The specifications stage refines the OOTD graph into a structure suitable for implementation in a hypertext system. The first step is to determine the first and second order links in the OOTD graph to produce a system flowchart. The first order links represent the links the student ordinarily should take to learn the material and may even correspond to the order of a textbook's chapters and section headings. The second order links, conceptual in nature, connect related material as desired. Next, the system flowchart may be refined several times eventually becoming the conceptual

graph where key concepts are grouped into sets of nodes with each set supporting a single instructional theme. The groups may be tightly coupled within themselves, but should be loosely coupled to other sets of nodes to permit the student some discretion in exploring the environment.

Through consideration of Gagne's Events of Instruction [6], the conceptual graph is augmented with additional nodes to address instructional and administrative issues. To gain attention, a title screen may be used. To inform the student of the lesson's objective, an introductory segment may be used to present an overview, lesson materials, lesson objective, and terminology list. Also, review screens may stimulate recall, information screens present material, and activity screens provide feedback and learning guidance. Practice screens review the material, test screens elicit performance, and performance reports assess performance.

Having been augmented with nodes, the conceptual graph becomes the content flowchart where all links in the flowchart are bi-directional except those to the test nodes which must be one-way to keep the student from exiting the test before finishing. The flowchart should be surveyed to ensure the centrality of key concept nodes [4] by verifying that all nodes can be reached and that concept groupings are tightly coupled. The flowchart may also be given to content experts to check the validity of the material.

Implementation Stage

The instructional designer takes the content flowchart and implements it in the authoring system of choice. The designer also designs and implements the interface screens (If they had been designed and implemented earlier, they might have imposed a premature structure on the hyperweb). The screens should be implemented in a consistent manner with no more than three to six colors per screen and designed to minimize user mouse movement and keyboard effort. For example, instructional material and navigation links should be placed in a common area on the screen.

The completed system should be reviewed by subject matter experts to ensure accuracy and completeness. All links should be checked. Once again, centrality of the key concepts should be verified. At this point the system can be used by a limited number of students to work out minor problems before general usage.

EXPERIENCES

The authoring methodology already has been used to create training products for diverse fields. At the University of Arkansas, it was first used by students to create a course for re-training machinists to operate computer numerically controlled milling machines [1]. Other students have used the methodology to create tutorials on networking, video conferencing, robotics, and object-oriented programming. During the past year, the methodology has been used at the United States Military Academy to create several courses for Army schools in such dissimilar areas as marine diesel engines and self-propelled howitzers.

In all cases, these training products were developed without graphics specialists and finely tuned multimedia production stations. The developers were college students familiar with the subject area. Design and production went better than in previous projects in which similar students had been left to their own ideas and impulses about how their multimedia work should be organized. The students successfully used the methodology to assemble and organize material that they wanted to include in their training product without in-depth knowledge of graph or hypertext theory. Instead, they could focus upon the concepts they wanted to teach and upon how their system might be used in the future.

One unique application of the methodology was in the re-engineering of a hypermedia resource for an undergraduate computer science course at the United States Military Academy (USMA) [2]. The course director used the methodology to re-organize and re-link the material in the resource. This material included slide shows, note taking guides, previous student projects, and over 1.2 Gigabytes of text, audio, and video files. This effort concentrated on improving the maintenance of the resource.

For example, the course directory structure previously had reflected 40 lessons that are taught during the academic semester at USMA. Each media type was isolated in its own directory and then connected to the lesson material through media-specific pages. In order to move a lesson, say interchange lessons 5 and 6, the course director had to edit eight files and then carefully review all slide shows and supporting materials to prevent files from becoming orphaned by cut links.

By applying some re-organization of the course directory structure, each directory is now dedicated to a topic covered in the course. All the material supporting that topic is kept in the same directory and linked directly to the slide show or applicable part of the note taking guide. Moving a lesson now requires editing only two files. As a consequence, the files are now directly connected to the instructional presentation and there is less chance that they will become orphaned.

As another example, one graduate student designed a hypertext tutorial on network protocols [12, 13] without using the methodology. The tutorial was hierarchically structured and divided into distinct portions, such as a description of the protocol analyzer written in C for the tutorial, a description of the code, and a description of protocol analyzers. Within the tutorial, some hypertext nodes could be 2.5 pages long or longer.

Although the tutorial was successfully used in class, it became clear that adding to the tutorial would be difficult. The hierarchical structure was not designed for presenting the major concepts to be learned. In addition, nodes were too long and needed to be split up into smaller learning chunks. Had the student started with the hypertext instructional design methodology, later additions could have been made easier by centering the tutorial structure on the concepts to be learned and keeping the nodes to a smaller length.

In contrast, another graduate student designed a videoconferencing tutorial using the methodology [15]. His tutorial screens were kept to a single template type and each hypertext node was only one screen in length. The methodology caused him to focus on keeping the hypertext network shallow so as not to create paths that strayed too far from the key concepts to be presented. He could easily add more material to his tutorial in the form of single screens attached to the others while endeavoring to keep the hypertext network shallow.

Benefits

Use of the methodology has resulted in benefits in two crucial areas of the software life cycle. First, in actual use in an undergraduate course, tests have shown that a hypertext resource that is structured with OOTD actually allows for an increase in information retrieval efficiency [2]; i.e., students spend less time on course readings and items at the lower end of Bloom's Taxonomy and more time on the analysis and synthesis portions of the learning objectives [5]. The second benefit to using the design methodology is found in the maintenance phase of the course. Because the designer has a graphic representation of how the parts of his course are interrelated, changes to the course material are easier to make. The graphic nature of the methodology gives a course maintainer the ability to see where changes need to be made and what effect the changes will have.

Adaptability or Accommodation of Individual Learning Styles

One of the strengths of using a graphical design methodology is that groupings of media become easier to visualize in the hypermedia resource making it easier to choose "guided paths"[7] through the material to cater to individual students' learning styles. Currently two projects at USMA are using the methodology in conjunction with learning styles. The first project is an HTML-based resource which uses an on-line assessment tool [16] to develop a suggested path through the lesson material [3]. The second project, CD-ROM-based and built with a commercially available multimedia authoring system, will offer students the choice of three "learning style paths". Both products use a combination of Gagne's Events of Instruction to structure the lessons [1] and the Felder Learning Model to describe the student [5].

FUTURE WORK

One of the most ambitious applications of the design methodology to this point is the creation of a hypermedia based course resource for an undergraduate core course, Psychology 1, at USMA. Still in the design and media creation phase of development, this course resource will enable students to select one of three learning style approaches to the course. By recording their selections with an on-line assessment tool and correlating their selections with their learning style expressed in terms of Felder's model, the possibility of predicting the media type which best suits each learning style will be explored.

Longer term projects will exploit the independence of the methodology on any particular learning model to design learning materials that will permit authentication and comparison of various learning models. One aspect will involve using the methodology to design materials that lead students "around the cycle," as suggested by some learning models familiar to engineering educators [10]. During the course of the design, these same learning models will be evaluated according to the degree of insight, utility, and convenience they afford for the design of hypermedia materials. A second aspect will involve using on-line assessment tools to evaluate the concept of teaching "around the cycle." More specifically, teaching around the cycle will be evaluated by creating guided paths that lead students through materials designed to accommodate several different styles and then comparing the performance assessments for those students who have learned "around the cycle" with those for students who have used mainly the materials designed to accommodate their preferred learning styles. Future plans also involve evaluation of cognitive learning theories, particularly constructivism [8], according to the degree of insight, utility and convenience they afford for the design of hypermedia materials.

REFERENCES

[1] W.J. Adams. *Application of Object Oriented Design Principles and Graphical Design Techniques to Computer Aided Hypertext Based Instruction*. Master's Thesis, Computer Systems Engineering, University of Arkansas, December 1994.

[2] W.J. Adams and C.A. Carver, Jr. "The Effects of Structure on Hypertext Design." *Proceedings of ED-*

Media 97, Conference on Educational Multimedia and Hypermedia, Calgary, Canada, June 15 - 19, 1997.

[3] Beumont and P. Brusilousky. "Adaptive Educational Hypermedia." *Proceedings of ED-MEDIA Conference on Educational Multimedia and Hypermedia,* Graz, Austria, June 17-23, 1995, pp. 93-98.

[4] R.A. Botafogo, E. Rivlin, and B. Schneiderman. "Structural Analysis of Hypertexts: Identifying Hierarchies and Useful Metrics." *ACM Transactions on Information Systems*, vol. 10, no. 2, April 1992, pp. 142-180.

[5] C.A. Carver, R. A. Howard, and E. Lavelle. "Enhancing Student Learning by Incorporating Student Learning Styles into Adaptive Hypermedia." *Proceedings of ED-MEDIA-96 Conference on Educational Multimedia and Hypermedia.* Boston, MA, June 17-22, pp. 118-123.

[6] E.K. Cook and E.J. Kazlauskas. "The Cognitive and Behavioral Basis of an Instructional Design: Using CBT to Teach Technical Information and Learning Strategies." *The Journal of Educational Technology Systems*, vol. 21, no. 4, 1993, pp. 287-302.

[7] D. Dee-Lucas. 1995. "Study Strategies for Instructional Hypertext: Effects of Text Segmentation and Task Compatibility." *Proceedings of ED-MEDIA-95 Conference on Educational Multimedia and Hypermedia,* Graz, Austria June 17-23, pp. 175-180.

[8] T.M. Duffy and D.H. Jonassen, Eds. *Constructivism and the Technology of Instruction.* Hillsdale, NJ*:* Lawrence Erlbaum Associates, 1992.

[9] D. Ehrlich and L. Reynolds. Integrating Instructional Design and Technology: A Model and Process for Multimedia Design." *Interactive Learning International*, vol. 8, no. 4, October-December 1992, pp. 281-289.

[10] R. Feldman, "Matters of Style," *ASEE Prism*, December 1996, pp. 18-23.

[11] L. Hardman, G. van Rossum, D.C.A. Bulterman. "Structured Multimedia Authoring." *ACM Multimedia 93*, Anaheim, CA, August 1993, pp. 283-289.

[12] S. Mengel and W.J. Adams. "The Need for a Hypertext Instructional Design Methodology." *IEEE Transactions on Education Special Issue on the Application of Information Technologies to Engineering and Science Education*, vol. 39, no. 3, August 1996, pp. 375-380.

[13] S. Mengel and S. Ali. "A Network Protocol Analyzer with Tutorial." *Applied Computing 1996*, New York: NY: ACM Press, 1996, pp. 115-119.

[14] J. Novak and D. Gowin. *Learning How to Learn*. Cambridge, UK: Cambridge University Press, 1986.

[15] J. Parikh. A Multimedia Tutorial for Videoconferencing. Master's Thesis, Electrical Engineering, University of Arkansas, December 1996.

[16] Syang and N. Dale. "Computerized Adaptive Testing in Computer Science." *Proceedings of the 1993 SIGCSE Technical Symposium on Computer Science Education,* February 1993, pp. 53-57.

[17] M.L. Talbert and D.A. Umphress. "Object-Oriented Text Decomposition: A Methodology for Creating CAI Using Hypertext," in H. Maurer, ed. *Lecture Notes in Computer Science #360, Computer Assisted Learning, Proceedings of the 2nd International Conference, ICCAL '89.* New York, NY: Springer-Verlag, 1989, pp. 560-578.