Computational Instructional Design for Construction of Adaptive Tutors in Real Time from Distributed Learning Objects

Kurt Rowley
Command Technologies, Inc.

US Air Force Research Laboratory, HEJ
Brooks AFB, Texas 78235-5352 USA

kurtrowley@gmail.com

ABSTRACT: With the advent of the WWW and recent interoperability and ontology initiatives, there are new possible approaches to generating real-time adaptive instruction by drawing from distributed interoperable learning object resources. In order to assemble adaptive tutoring systems appropriate to the needs of individual students, a computational form of instructional design has been developed with a generic instructional vocabulary. The vocabulary facilitates the automatic construction and management of tutoring environments from distributed interoperable components. A demonstration web-based ITS that generates individualized and adaptive instruction in English Composition through the use of an interoperable instructional vocabulary is described.

KEY WORDS: Intelligent tutoring, Interoperability, Shared ontology, Writing process, Learning objects

TOPICS: Architectures for ITS, Deploying ITS, Educational objects/agents, Web-based ITS

January 14, 1998

Computational Instructional Design for Construction of Adaptive Tutors in Real Time from Distributed Learning Objects

1. Introduction

This paper describes the rationale behind the design of a web-based ITS that generates individualized and adaptive instruction for English Composition through the strategic use of an interoperable instructional vocabulary [1]. The vocabulary is based on an instructional task ontology [2] that is shared among multiple JAVA-based learning object resources for teaching procedural writing skills. The vocabulary facilitates a real-time, computational instructional design process [3] for generating individualized tutoring environments that follow a variety of tutorial strategies, such as Socratic dialogue, coached practice, collaborative learning, constructivistic, and cognitive apprenticeship approaches.

Computational instructional design will be presented as one approach to managing strategic and tactical decision-making in the assembly and adaptation of the tutoring system and its eventual presentation to the learner. Next will be a discussion of the use of a shared task ontology, and instructional vocabulary, to formalize the computational instructional design strategy and facilitate the selection and adaptation of tutoring modules in real-time. The discussion will continue with an example of meeting individual student needs through the use of rich meta-data

to select distributed JAVA-based learning objects, and construct systems with a high level of adaptability. A description of a system using the interoperable instructional vocabulary in the domain of English Composition skills will be included.

2. The Need for Computational Instructional Design

Consistent with the increased focus on object-orientation in systems design, learning objects are an important concept in the sharing and reuse of instructional technologies at the component level. A learning object is an emerging concept, so a universal definition is not available, but will likely come into view in the future. For purposes of this paper, a learning object will be defined as any bounded object that performs a function in a learning environment. By this definition, a learning object in an ITS context could be an independent student model, a knowledge base, or even a JAVA-based instructional module of any type. The characteristics of such learning objects could be communicated through reference information with rich meta-data.

In order to develop an intelligent instructional system capable of drawing from widely distributed interoperable JAVA-based learning object resources, it is necessary to develop an approach for the strategic selection of instructional applets or server-side JAVA modules (servlets) to meet specific needs of students. The approach should also help the selected applets adapt to the needs of students. The use of an instructional design strategy in a computational form is relevant and necessary since traditional tutoring system instructional strategies involve mostly lower-level or tactical decision-making. For example, an ITS may trace a model of expected performance while observing student performances, recognizing when the student has demonstrated a known misconception and should be remediated [4]. This traditional ITS strategy, or instructional model, operates best at a tactical level, or the lower-level tutorial advice and exercises. However, for assembling an entire tutoring system from distributed interoperable objects to meet a specific learning need, a strategy is required that considers the strategic level design of the overall system before collecting resources that can be adapted to solve complete skill-set problems. In other words, the computational instructional design system operates at a level of scale above the traditional instructional model in an ITS, constructing real-time tutoring systems from distributed learning objects [5], as well as facilitating the adaptation of those learning objects to specific needs and requirements of students.

The computational conception of instructional design used for this illustration is derived from classic instructional design theory. For example, an early U.S. Interservice ISD model identified the instructional design task as a step-wise process proceeding through analysis, design, development, implementation, and control (evaluation) stages [6]. Other models of the design process have viewed the design of an effective delivery system as a classification problem to be solved, and focused on using a systematic, scientific process to classify the skills to be taught in order to help organize and evaluate methods appropriate to teach each of those skills [7]. More recent efforts have expanded the views of instructional design to include differentiation between novice and expert design approaches, as well as scaling design and engineering tasks to the needs of the design situation [8,9]. Although the models of design processes used by designers vary in details, most of them share a common ends-before-means, or backwards-chaining approach to the design process. The simple ADDIE model (Analyze, Design, Develop, Implement, Evaluate) is perhaps the most generic task model for designing instruction, presenting a 'waterfall' style process that starts with the definition of terminal instructional objectives, and works backwards from the objectives through a process of analyzing the learner and environment, designing and sequencing instruction so it will lead to the achievement of the

objectives, then developing, formatively evaluating, revising, implementing, and summatively evaluating the instruction.

An expert designer also follows this empirical process and engages in many of the same design tasks as the novice-level designer. However, expert designers rely more on experience-based judgment, specific design activities are often more open-ended, there is usually more use of rapid prototyping and concurrent tasking, and the execution of design tasks may not follow any predictable sequence [8,10]. Although the tasks performed by expert instructional designers are too complex for computational reduction, a simplified approach to instructional design based on the generally accepted paradigm of a backwards-chaining problem-solving process with the goal of the improvement of specific skills to a particular criterion level, similar to the classic view of the instructional objectives [11], can be represented computationally, through the use of a task ontology combined with a problem-solving vocabulary. This approach is consistent with both novice and expert patterns of designing, but without the higher-level expert judgement of the experienced designer.

3. Using an Instructional Vocabulary for Computational Instructional Design

In order to use a shared task ontology and instructional vocabulary to formalize the computational instructional design strategy and facilitate the selection and adaptation of tutoring modules in real-time, a basic task ontology has been developed for instructional design. A vocabulary was designed to use the ontology and provide a standardized, domain-independent representation of an instructional strategy that will facilitate the real-time construction of tutors from distributed learning object resources. Mizoguchi and associates [12] argue that a well-designed common vocabulary can be developed through the engineering of a task ontology, much like the use of knowledge engineering in expert systems. Their view of a task ontology is different from a traditional domain ontology, which describes pure domain knowledge. They suggest that a task ontology be focused on representing a problem-solving process, with the representation broken down into four generic components: nouns representing objects, verbs representing activities, adjectives modifying noun objects, and other concepts specific to the task (see 3.1 in their paper [12]). This type of task ontology is an appropriate construct for computational instructional design of tutoring systems, as it allows for the representation of the components of an instructional system and their interrelationships in an objective-seeking problem solving context, consistent with the problem-solving focus of most instructional design task models [6-11].

In order to develop a generic instructional vocabulary, an ontology has been designed that allows a broad range of instructional strategies to be represented in a sentence-like form. The vocabulary is composed of terms that can beobjects, activities, attributes, goals, or criteria. These terms can be combined using a simple grammar in order to represent instructional strategies in a sentence-like form. The sentences follow the form of: "objects with various attributesperform activities that accomplish instructional objectives as measured by criteria." The proposed vocabulary is designed as a generic tool that will allow instructional strategy information to be shared between diverse learning objects. The following list includes a sample of the terms used for the vocabulary.

Proposed Generic Instructional Terminology

- Structural Objects (showing only top level of the task ontology)

learner (with learning styles, attributes, prerequisite knowledge and skills, history, etc.)
instructor (human)
cohort/peer learner(s) (in the learning environment, or as team members)
instructional resources (interactive software systems, etc.)
reference information (print or electronic indexing)
tools (physical or electronic)
facilities (physical)
environment (human, physical, or electronic)

- Activities of Objects (functions of structural objects and their attributes over time)

learner

observing or otherwise sensing (human, all inputs)
learning declarative information
learning procedural skill
remembering (human, contextual)
deciding (human choice)
reasoning (human cognition)
exploring
organizing
requesting feedback or information
responding to inquiries
practicing or performing (authentic activity)
interacting with teammates or peers
playing (relaxing or recreation)

instructor

presenting information
reminding of information or process steps
showing
assessing performance
controlling learning process
managing learning environment
monitoring work or performance and providing feedback

cohort / peer learner

interacting cooperatively
interacting competitively
interacting socially
instructional resource (interactive software system)
presenting (print, electronic)
connecting (remote, networking)
simulating (kinetic or cognitive performance environment)
reminding (print, electronic)
showing (print, electronic)
assessing (performance)
controlling (print, electronic)
monitoring (electronic)
. . .

- Modifying Attributes for activities of objects (modifiers for any level of ontology)

communication between and among
time required for XX (learning, practice, performing, etc.)
storage space (physical and electronic)
capability or capacity (human intelligence and abilities)
interest in (human)
level of mastery of (learning, performance)
use of cognitive tool XX to (software object)
use instructional resource XX to (software object)
looking-up reference information in order to (electronic)

- Instructional Goals of activities (skill representation, the purpose or objective of an activity)

learner

a fact or concept is known
a skill or task has been (or can be) performed independently or interdependently
a choice or decision has been (or can be) reached
personal or group needs have been (or can be) met
goal achieved or progress made toward goals (or rewards)

instructor

student successfully supported or guided during learning and practice
directed or controlled use of technology by student
assessed and certified student performance
managed learning environment for multiple (potentially interacting) students
designed curriculum for multiple students to accomplish educational objectives
domain knowledge and skills improved (instructor's knowledge)

cohort / peer learner

interacted cooperatively/collaboratively with learner during teacher-directed work
socialized learner
provided peer product feedback or review
provided peer tutoring or mentoring
participated in collaborative group learning activities

instructional resources (interactive software system)

capabilities of a human instructor extended
learner coached
dangerous or expensive learning environments simulated
individualized instruction provided (adaptively)
timely instruction or feedback provided (just-in-time)
individualized, self-paced instructional environment provided
capabilities and developing skills of student and adapt instruction addressed
instruction sequenced (adaptively)
ability or proficiency of student assessed
. . .

- Performance Criteria for goal accomplishment

targeted instructional and educational outcomes improve by XX amount
accomplishment of XX objectives & purposes demonstrated by YY (specific deliverables)
XX instructor time (freed for more productive, individualized interaction with students)students, instructors are more satisfied as measured by X

improved performance as evidenced by XX measures of outcomes, attitudes or opinions
XX measures of process/product increase/decrease by YY amount
. . .

A basic format for this problem-solving vocabulary reflects the notion common in instructional systems design that an instructional strategy can be represented as an objective-seeking system [7,11]. To represent an instructional strategy computationally, the concept of an objective-based instructional system will be reduced to goal-seeking equations that can be represented as statements. The general syntax of a statement is that one or more objects are modified byattributes, and combined with an activity in order to eventually accomplish an instructional goal according to a skill improvement criteria for a desired outcome. An example of using this type of statement to represent one part of an instructional strategy is:

LEARNER (object) uses COGNITIVE TOOL XX (attribute) to perform REMEMBERING (activity) until FACT IS KNOWN (goal) as measured by OUTCOMES IMPROVE BY XX (criteria)

The statement can also be represented as an equation in which the object interacts with an attribute, and the goal interacts with the criteria:

LEARNER [uses COGNITIVE TOOL XX] + REMEMBERING = FACT KNOWN [measured by OUTCOMES IMPROVE BY XX]

The equality condition of the equation represents the final state of the instructional strategy. The equality goal is not reached until the objective is accomplished according to the completion criteria. In other words, each instructional strategy statement represents an independent problem-solving process in which an activity is specified that is available in a learning object to accomplish an objective according to a completion criteria. If the process were formed as a generic equation, one possible representation could be the following:

OBJECTS[use ATTRIBUTE] + ACTIVITY = GOAL[measured by CRITERIA]

Or, for a more readable general statement:

OBJECTS use ATTRIBUTE to perform ACTIVITY until the GOAL is achieved as measured by the CRITERIA

This simple example should illustrate the general concept of using an instructional vocabulary to represent the instructional strategy used within interoperable learning objects. The vocabulary can be used at many levels within a given tutor. This may include use of the vocabulary as part of a query to identify candidate modules from distributed resources, or within the learning object itself. The vocabulary may also include variables that link to other objects, or statements. These additional objects would then provide additional representations of activities, and continue to use the vocabulary. Through this type of multi-layered approach, the vocabulary can be used both to assist in the selection of learning objects, and to facilitate representation of the instructional tactics used within the learning objects.

4. Real-Time System Generation from Distributed Components

In order to utilize the generic instructional vocabulary to construct and operate a tutor from distributed learning object resources, such as JAVA applets, domain knowledge bases, student models, or other system resources, all objects must be compliant with a standardized instructional ontology. An example of the real-time generation of a tutoring system for teaching English composition will illustrate how the system utilizes distributed learning objects to generate adaptive tutoring environments.

Example of Real-Time Tutor Generation

A good example of real-time tutor generation over the Web would be as follows: imagine that a learner is interested in improving English composition skills. The learner is connected to the web, and loads the home page of an English composition home page. The home page loads an applet or servlet that pursues the following procedure in order to generate an adaptive tutor from learning objects available to it on the web:

1. Survey the learner to determine entry skill levels. This may include asking the learner to write a short paragraph or composition, followed afterwards by a series of questions about how he or she approached the writing task.

2. Analyze the learner survey. Classify the learner and generate a learner profile. This may include information relative to prerequisite skill levels, learning style, interests, etc.

3. Prepare a search string for the identification of candidate learning objects related to writing skills for the learner. For example, if the learner needs help in organizing his or her ideas prior to drafting, the tutor might ask additional questions to determine the needed level of improvement. Next the tutor would compose a search string to locate appropriate learning objects. The search string may be "find all learning objects in domain ENGLISH WRITING SKILLS where LEARNER uses COGNITIVE TOOL (IDEA ORGANIZER) to perform ORGANIZING until SKILL CAN BE PERFORMED INDEPENDENTLY as measured by TARGETED WRITING OUTCOMES IMPROVE BY 50%".

4. Attached to the search string might be meta-data about the terms, including student profile information such as the student's preferred learning style, etc.

5. Search the Web for candidate learning objects based on learning object meta-data. The learning objects should be compliant with the instructional vocabulary and related interoperability standards.

6. Search the Web for candidate knowledge bases or other resources need for the tutor, following a process similar to 3-5 above.

7. Identify appropriate knowledge bases and then configure the learning objects collected. This may include further use of the vocabulary to select appropriate parameters for the learning objects.

8. Assemble the tutor. The tutor should provide an environment with learning strategies that will help the learner improve writing skills.

9. Provide the learner with choices related to how the student prefers to proceed. For example, the student may wish to explore expert performance strategies, or perform practice exercises, or simply hold a discussion with the tutor. The options available to the student at this point may vary depending on the learning objects that have been found.

10. The tutor runs the selected learning objects in the learner's browser's virtual JAVA machine, based on the learner's choices of how to proceed. The learner may also download the tutor or portions of the tutor to an off-line JAVA machine device or palm computer, depending on the specific skills to be worked on.

The following illustration of a system architecture should help further to clarify how the real-time tutoring system components are related (see Fig. 1).

Figure 1 - Real-Time Tutor Architecture

(click on image to expand)

5. A Web-based English Composition Tutor

As illustrated in Fig. 1, the tutoring environment allows multiple learning objects (represented as JAVA/HTML instructional resources) to be connected with knowledge bases and adapted to the needs of the learner according to the tutor definition (using the instructional vocabulary). The design of the interoperable environment is currently being evaluated in the domain of English composition. A tutor has been designed that draws from distributed tutoring resources and assembles a tutoring system on-demand based on the definition of the tutor using the instructional vocabulary. The skills for the tutor are taken from the MAESTRO Writing Process Tutor (see Fig. 2).

Figure 2: MAESTRO Knowledge/Skills

(click on image to expand)

The original MAESTRO Writing Process Tutor was a LAN-based tutor designed as part of the U.S. Air Force's Fundamental Skills Training (FST) Program. FST was a multi-year research effort to transfer advanced, adaptive training technology capabilities developed under Air Force technical training research to public education. MAESTRO teaches the procedural skills of an expert writer, based on cognitive research into the writing process, and several years of evaluative teacher feedback. MAESTRO was implemented successfully over LANs in computer labs at 12 High Schools in the U.S. during the 1996-97 school year in a traditional controlled experiment. Ongoing research is addressing the effectiveness of MAESTRO [13,14].

The instructional resources used for the web-based English composition tutor include MAESTRO's workspace designs and expert performance rules derived from the writing process skills hierarchy (see Fig. 2). The specifics of expert writing process skills were identified through protocol analysis and extensive research by cognitive scientists [15]. The components of the tutor are all compatible with the tutor's instructional vocabulary in the sense that their operation will produce results consistent with the writing process skill goals represented by the tutor's use of the instructional vocabulary.

The tutor further demonstrates the flexibility of the instructional vocabulary and the interoperable design through the use of multiple instructional strategies, each constructed from

the same domain and student knowledge bases, but utilizing a variety of JAVA or HTML-based instructional resources. For example, the main menu (the left frame on the web page) includes several tutoring approaches from which the student may select the approach most relevant or interesting. The options available to the student (after taking an initial performance assessment survey) include open discussion of the student's writing approach in an 'Eliza' style environment (Socratic), exploration of the writing process in a hyper-text environment (constructivistic), practice of specific steps in the writing process in a coached environment (tutored environment), help information based on actual student writing tasks and problems (cognitive apprenticeship), and collaborative work on joint or group writing projects (CSCL). The following screen is the home page from the user interface of the tutor:

(click on image to expand)

6. Conclusions

The feasibility of an ontology-based, computational form of instructional design as a mechanism for the real-time construction of adaptive tutoring systems from distributed learning objects has been demonstrated by the design of the English Composition tutoring system. The English tutor is currently a prototype that is providing a useful proof-of-concept for the value of interoperability in facilitating efficient tutoring system development and implementation. The completed tutor will demonstrate the value of computational instructional design in the automated construction of new tutoring systems from component parts. This should include the ability to reliably produce an effective tutoring system from components that will work together from distributed resources and allow an integrated tutoring system like the English tutor to select from alternate instructional strategies in order to meet needs of individual students. Harnassing the WWW for the construction of ITS systems from broad-based resources should be important for the practical success of adaptive learning objects.

References

1. Rowley, Kurt. A generic instructional vocabulary for an interoperable distributed learning environment: Adapting MAESTRO the Writing Process Tutor. Proceedings of the AI-ED 97 Workshop on Interoperability, Kobe, Japan, August 19, 1997.

2. Mizoguchi, R, Ikeda, M, & Sinitsa, K. Roles of Shared Ontology in AI-ED Research: Intelligence, Conceptualization, Standardization and Reusability. Proceedings of AI-ED 97 World Conference on Artificial Intelligence in Education, Kobe, Japan, August, 1997.

3. Rowley, Kurt. A design approach for the engineering and construction of CSCL-ITS environments. Proceedings of the AI-ED 97 Workshop on Computer-supported collaborative learning environments and AI, Kobe, Japan, August 20, 1997.

4. Anderson, J., Corbett, A., Fincham, J., Hoffman, d., & Pelletier, R. General principles for an intelligent tutoring architecture. In Regian, J.W. & Shute, V. (Eds.) Cognitive approaches to automated instruction (pp.81-106). Hillsdale, NJ: Lawrence Erlbaum Associates, Publishers, 1992.

5. Rowley, Kurt. The challenge of constructing a mega-tutor over the web. Proceedings of the AI-ED 97 Workshop on Web-based ITS, Kobe, Japan, August 20, 1997.

6. Branson, R., Rayner, G., Cox, J., Furman, J., King, F., and Hannum, W. Interservice procedures for instructional systems development (5 Vols.) (TRADOC Pam 350-30 and NAVEDTRA 106A). Ft. Monroe, VA: U.S. Army Training and Doctrine Command, 1975.

7. Gagne, R., Briggs, L. & Wager, W. Principles of instructional design. Fort Worth, TX: Harcourt Brace Jovanovich College Publishers, 1992.

8. Rowland, G. What do instructional designers actually do? An initial investigation of expert practice. Performance improvement quarterly, 5(2) pp.65-86, 1992.

9. Tessmer, M. & Wedman, J. F. A layers of necessity instructional development model. Educational technology research and development, 38(2), pp.77-85, 1990.

10. Tripp, S. & Bichelmeyer, B. Rapid prototyping: An alternative instructional design strategy. Educational technology research and development, 38(1), pp.31-44, 1990.

11. Mager, R. Preparing instructional objectives. Belmont, Calif.: Pitman Management and Training, 1984.

12. Mizoguchi, R., Sinitsa, K, and Ikeda, Mitsura. Task ontology design for intelligent educational/training systems. ITS'96 Workshop on Architectures and Methods for Designing Cost-Effective and Reusable ITSs. 1996.

13. Rowley, K. & Crevoisier, M. MAESTRO: Guiding students to skillful performance of the writing process. Proceedings of ED-Media 97, Calgary, June, 1997.

14. Rowley, K. Re-engineering existing learning technologies: designing MAESTRO, a second generation writing tutor. Technical Report of U.S. Air Force Research Laboratory, HEJT, Brooks AFB, Texas.

15. Flower, L., and Hayes, J. R. The dynamics of composing: Making plans and juggling constraints. In L. Gregg, & E. Steinberg (Eds.), Cognitive processes in writing (31-50). Hillsdale, NJ: Lawrence Erlbaum Associates, 1980.