AuthorIT & TutorIT: An Intelligent Tutor Authoring & Delivery System You Can Use

JOSEPH M. SCANDURA

Director of Research, TutorITweb, Emeritus Professor, University of Pennsylvania, USA

Intelligent Tutoring Systems (ITS) have a long history, almost as long as the Structural Learning Theory (initially in Scandura, 1971). Although well-funded for many years, neither ITS nor contemporary successors based on BIG DATA (e.g., Knewton) come close to modeling the processes used by good human tutors. AuthorIT & TutorIT rest on a fundamentally different theoretical foundation designed from inceptions explicitly with this in mind. After summarizing approaches, this paper focuses on answering four basic questions that good human tutors must ask, explicitly or implicitly. It shows how AuthorIT authoring and TutorIT delivery technologies have made it possible in a highly cost effective manner to build a broad variety of TutorIT tutorials that interact with students as might a good human tutor. Those interested in field testing or further research are encouraged to contact the author.

Anderson’s ACT-R theory makes a sharp distinction between procedural and declarative knowledge (Anderson, 1996). It also has spawned a large amount of research in Intelligent Tutoring Systems (ITS) (Anderson et al, 1995). Nonetheless, after years of research, ITS still have many limitations (Psotka, 1988). Although a variety of ITS have been built for research purposes, commercial applications have been limited to Carnegie Learning algebra. Moreover, research shows not only that they are extremely expensive and time consuming to build but that ITS miss about 70% of what is important for students to learn (Clark, 2013).

Given sub-stellar results, attention in adaptive learning in recent years has turned to BIG DATA, Learning Analytics (LA) and Data Mining. Much like Factor Analysis in the past, the hope has been that LA will yield a utopian, fully
automated solution. No need for human thought. Knewton has been a prime mover in the commercial space. Knewton makes all pedagogical decisions automatically. When a student gets something wrong, simply find a related problem students on average find easier. When right, give the student an item that students on average find more difficult. Unfortunately, it is not that simple.

Research shows that the best decisions are made by combining automation with human judgement. Google search is a wonderful tool. It almost certainly inspired LA. We all use Google search to find more or less promising prospects. But who makes the final judgment? In the same way, human judgement is essential in pedagogical decision making. The question is, “How?”

Results in both ITS and LA have been a long way from the gold standard for Computer Based Instruction (CBI) set by Ben Bloom many years ago. Bloom found that students aided by good human tutors performed at a level, two-standard deviations above students taught in classrooms – in the top 2% (Bloom, 1984).

This paper introduces an automated solution that has the potential of meeting that standard in predefined content domains. Specifically, I’d like to introduce you to our AuthorIT authoring and TutorIT delivery systems – more specifically, how these systems may have the (to be determined) potential to bring Bloom’s goal within reach. Equally perhaps more important, these systems enable subject matter experts and instructional designers, perhaps even teachers, to build dynamically adaptive (AKA intelligent) tutoring systems in their own areas of expertise. TutorIT tutorials also enable instructional researchers to conduct more definitive research on different pedagogical methods (e.g., didactic/expository versus discovery).

TutorIT and AuthorIT build on the Structural Learning Theory – in turn on decades of basic research in math education, problem solving, AI, cognitive psychology, software engineering and instructional design. Under development for some time, we are now ready to gradually release selected portions of these technologies for beta use.

Let me begin with a short historical overview. Traditional CBI began in the 1960s and 70s with lots of promise, including debates over Crowder’s branching versus Skinnerian reinforcement. Shortly after micro-computers came on the scene, we introduced our MicroTutor II authoring and delivery systems with some success in the early 1980s. Borg Warner Educational Systems, Queue and we ourselves produced 1000s of MicroTutor II tutorials. Royalties continued into the early 2000s– even though these tutorials only ran on the old Apple II computer.

Like other CBI in that era, MicroTutorII CBI systems were essentially one-size fits all – provide instruction and test to see what was learned. We experimented at the time with tutorials based on an early version of my Structural Learning theory

---

2 MicroTutorII came before AuthorWare, which became dominant in the field for several decades.
(SLT), but alas, the old Apple II only had enough computational power to create a tiny sample.

To address these and other technical issues, we turned our attention to research in software engineering working with the US Air Force, Army, Office of Education, and NSF culminating in a $33M+ 3 year project with NIST (Dept. of Commerce) led by IBM, along with us and a couple other small companies like Boeing and TRW, with the goal of automating the supply chain.\(^3\)

Although our software engineering research had nothing directly to do with education, the technologies we have developed, along with patented methods building on SLT, solved a fundamental issue in knowledge representation in the cognitive sciences. Namely, what level of abstraction is most appropriate for representing to-be-learned knowledge. The short answer: We discovered a way to represent to-be-learned knowledge SIMULTANEOUSLY AT ALL LEVELS OF ABSTRACTION. This solves the aforementioned problem (what is the appropriate level of knowledge representation) and opens a wide range of options.

The rest of this paper deals with the immediate future. Specifically, I will summarize what we have accomplished to date and describe how subject matter experts, potentially even teachers, will soon be able to create dynamically adaptive tutoring systems in their own areas of expertise – without programming. We shall see that TutorIT is a dynamically adaptive tutoring system that interacts with students as would a good human tutor. It makes its decisions based on what individual students do and do not know at each point in time. TutorIT is designed to ensure mastery of the operations and decision making skills required for success – cognitive or otherwise.

AuthorIT refers to a family of authoring platforms. We’ve used these platforms to create a broad variety of TutorIT tutorials. Most are now ready for field testing. Our immediate goal is to make these authoring platforms usable by non-programmers – to enable instructional designers and subject matter experts to create dynamically adaptive TutorIT tutorials in their own areas of expertise.

**Background.**—All instructional designers are familiar with the so-called educational games children and students play. Figure 1 shows a couple sample games.

While entertaining – it is unclear what students are actually learning. Research shows only one thing for certain: The more time students spend playing a game, the better they get at playing the game. The unanswered question is: How much educational value do they receive?

There’s a lot of serious educational software out there of course. Also various theories and biases as to how to use these educational resources: Blended learning,

---

\(^3\) In the interim, Intelligent Tutoring systems (ITS) were introduced in the 1980s. More recent work has been based on Learning Analytics.
Flipped classroom, etc. All involve competency based learning in some form. The problem is the noise – how to distinguish promises from results? Moreover, almost all are of the one-size fits all variety, the best with a pretest to determine student starting point, followed by on-size-fits-all instruction and a posttest (e.g., see Figure 2).

**Intelligent Tutoring Systems** (ITS) attempt more -- to find out what is going on in students minds (see Figure 3). The result is systems that are very complex. ITS
theories typically start with basic assumptions about: a) knowledge that might be in student brains, b) how students use that knowledge. ITS’ decision making is based on these assumptions. The results are complex, prone to error and hard to correct.

ITS make assumptions about what might be in student brains, and also how students use that knowledge. Instruction is based on those assumptions. This is a very complex task. It costs lots of time, money and specialized expertise. Results have been minimal despite millions in Federal Funding (e.g., $6M plus for Rand study of Carnegie Learning ITS) (e.g., Anderson et al, 1995; 1996; Corbett et al, 1997; Koedinger et al, 2009; Psotka et al, 1988).
Major approaches in use today fall generally in one of two categories: **Top-Down** and **Bottom-Up**. There are two general varieties of Top-Down methods:

1. **Task Analysis** is still a widely used art. It starts from the top organizing to-be-learned knowledge hierarchically. The major problem here is that there is a big gap between theory and the resulting instructional systems. As it was put by an old colleague, Norbert Seel, “Why is it that we all start with different ‘theories’ but the resulting systems all look the same?” I refer to this as the implementation gap.

2. **Relational networks** focus on relationships between various concepts and other instructional elements (see Figure 4). The problem here is that there is no uniform way to make instructional decisions. While some systems like ALEX are based on such networks, pedagogical decision making in such systems depends directly on the particular relationships involved. The result is heavy development costs and difficulties in making adjustments based on actual usage.

Other development methods work from the bottom up. Bottom up approaches are shared by ITS researchers and BIG DATA technologists – wherein the computer does the necessary analysis.
3. **Intelligent Tutoring Systems** as above have a rigorous technical foundation, made up of lists of If, then productions (condition-action pairs) and learning mechanisms assumed to govern their use. For example,

\[ C_1, C_2, C_3, \ldots, C_k \implies A_i \]
\[ D_1, D_2, D_3, \ldots, D_n \implies A_j \]

4. **Learning Analytics** approaches store and use captured Learner Data to advise learning. These systems base pedagogical decision making basically on correlations. If a student gets a problem wrong, the system goes out, finds an item that students on average find easier and present that to the student. If he or she gets a problem correct, the adaptive learning system does the reverse: Find a problem that students on average find more difficult. **The problem is the fundamental difference between correlations and causation.** Just because students on average find a problem more or less difficult does not mean that is the case with any particular student. Correlations are very different from causation.

More generally, aside from their high costs, the focus in both ITS and BIG DATA Learning Analytic systems the focus is on **“how”** students learn, not on **“what” students need to know and be able to do to be successful** (Scandura, 2014). This distinction is fundamental and distinguishes what has gone before and what we see as the future. We envision a near term future where authors can guarantee predefined levels of student mastery of any subject assuming that the student enters with assumed prerequisites.

**Introducing AuthorIT authoring and TutorIT delivery systems.** – TutorIT tutorials and associated authoring systems are based on decades of basic and applied research in the Structural Learning Theory (SLT) (cf. Scandura, 1971, 1973, 1977, 2001, 2007). Technical developments based on this research over the past ten years are finally coming to fruition as shown in Figure 5. Specifically, I’d like to introduce you to these recently patented technologies – most important, what they can do for many of you.

Developing dynamically adaptive (AKA “intelligent”) tutoring systems requires answers to four basic questions. These are questions that any good human tutor must ask.

1. **What do I want the student to learn? (What does it mean to know something?)** Instead of worrying about **how** students learn, a good tutor needs to know **what** a student needs to know to be successful. This fact is the conclusion of my very first piece of serious research. During the heyday of the new
math, a major assumption was that students learned better when they discovered something. Deeper analysis revealed that what really matters is what a student knows when he or she receives instruction. Too soon and it falls on deaf ears, too late and it did not matter. Even more important, we later found that one could directly teach by exposition what was learned in discovery – and this could be done more efficiently (Roughead & Scandura, 1968).

2. **How can I find out what the student does and does not know?** Tutors also need a way to determine what any given student does and does know -- at each point in time.

3. **How do students acquire new knowledge? How do they learn?** This question only becomes relevant once we know what the student must learn and what he or she already knows that is relevant. SLT assumes and data supports the idea that students use higher order knowledge to acquire new knowledge (Scandura, 1967).

4. **How would a good human tutor put it all together to help each student learn?** How do good human tutors put this all together in deciding what do next? Let’s take a deeper look!

TutorIT Tutorials, based on Structural Learning Theory (SLT), offer direct answers to each of the above four questions:

1. **What must be learned for success:** In the SLT, to be learned knowledge is represented in terms of hierarchical Abstract Syntax Trees – ASTs. Figure 6
shows how to-be-learned knowledge is represented in AuthorIT. This hierarchical representation is very important. It represents all possible states of knowledge about column subtraction. This particular hierarchy was constructed using SLT’s method of Structural Analysis.

The top level node represents expert knowledge. “Subtract the bottom number from the top number” operates on subtraction problems as wholes. It represents expert knowledge. Given a top number and a bottom number, a student who has mastered column subtraction at this level of expertise can quickly find the difference – potentially in one’s head. It represents “expert knowledge”: It represents a high level operation on complex data.

There are two nodes at the next level, “REPEAT Subtract the current column” and “UNTIL Have you finished subtraction?”. The REPEAT is further refined into “Click …” and “Regroup …”. And so on until we get down to basic skills like “Subtract … the current column”. In general, refinement continues until the skills and decisions required are so simple that they correspond to prerequisites that can reasonably be assumed to be available to every student in the target population.

2. **What a given student knows:** As shown in Figure 7, what any given student knows is measured relative to the initial knowledge representation. Individual nodes are marked “+” (mastered), “−” (known to be unknown

---

**FIGURE 6**
What needs to be learned for success.
and “?” (status yet to be determined). The knowledge representation for this particular student, for example, at this particular point in time, is known to have mastered the nodes in green preceded by a “+” sign. Those is red preceded by a “-” sign are known to be unknown. Notice also that the top level red node in the hierarchy above the “REPEAT Subtract …” node is also is red and marked “-”. Conversely, all nodes below any “+” green node will necessarily be “+”. Prerequisite nodes are at the bottom of the hierarchy.

The way this inferencing works is very important. The various transitions are illustrated in the schematic below (Figures 8A,B,C,D&E). The initial hierarchy on the left represents a starting point. The learning status of the student at this point in time is assumed to be completely unknown. Hence, the student’s knowledge is unknown to TutorIT. This is illustrated in Figure 8A (with all question “?” marks) in the elements. The center figure (Figure 8B) signifies that the student has demonstrated mastery of tasks associated with the green highlighted node with a “+” sign. Observing this, TutorIT automatically infers that the student also knows the prerequisite nodes below as shown in Figure 8C at the right. In this case, all are made green and marked “+”.

FIGURE 7
What the student knows.
Figures 8D and E show what happens when the student fails on tasks associated with a node in the knowledge representation. This is shown by the red node marked “-” in Figure 8D on the left. Analogous to the inferencing used in the case of success (+), TutorIT in this case also marks higher level nodes (in this case only the top level node) with a “-” – designating unknown (see Figure 8E).

3. How Students Use What they Know to Acquire New knowledge. It is beyond the scope of this paper to detail the precise mechanisms used in SLT to explain the learning process. See Figure 9 for a quick overview. For specifics, please see Scandura (cf. 1971, 1973, 2007). Each of these arrows represents a hierarchy (AKA SLT rule). Think of it as a “chunk” of to-be-learned knowledge. The higher order AST “chunk” in the middle operates on AST chunks and generates new ones (e.g., the combined chunk at the bottom). In the SLT (Structural Learning Theory) students learn or acquire new knowledge using their higher order knowledge to create new knowledge from their existing knowledge. Lower and higher order ASTs (SLT rules) are both derived via Structural Analysis (SA) from the given content domain.

4. How TutorIT Works. Figure 10 puts it all together. The Knowledge Representation at the top represents what students are to learn. TutorIT interacts with the Learner (student) through the Blackboard Interface. TutorIT displays problems and/or partially solved problems on the Blackboard Interface. The Learner responds. TutorIT evaluates each Learner’s response and reacts accordingly.
FIGURE 9
Schematic representation of how learning takes place in SLT. Higher order SLT rules (ASTs) operate on other AST rules under control of SLT’s Universal Control Mechanism (UCM). The results are new SLT rules.

FIGURE 10
Shows how TutorIT interacts with Learners.

As above, paralleling what a human tutor might do, TutorIT marks nodes in each hierarchical AST display as known or unknown as the case may be. Furthermore, hierarchical relationships between the nodes allow TutorIT to infer status on untested nodes. This is essentially what human tutors do instinctively – this type of inferencing is what makes it possible for human tutors to quickly home in
on student needs. The main difference is that TutorIT does this both automatically and more systematically.

There is obviously a lot more to say about how learners learn, how TutorIT makes its decisions and relationships between them. As we shall see, there also is a lot to say about our authoring processes. Toward this end, please review the articles referenced under “GET AuthorIT/TutorIT Publications” at www.TutorITweb.com. Key ideas are unique and recently patented.

Sample TutorIT tutorials.—Figure 11 shows sample entry and introductory screens providing an overview.

The video links are optional. They are used to provide either overviews and/or introductions to the tutorial.

In the screen shot on the left in Figure 12, TutorIT is essentially asking the student where to begin Column Subtract. The one on the right shows what happens in TutorIT after a student correctly answers the question. The bottom screen shot was taken immediately after the student answered correctly, and shows TutorIT’s response. Clicking (in correct locations) is just one type of response accepted by TutorIT.

The Learner Model on the left (Figure 13) shows what is going on behind the scenes. Given the student’s correct response, TutorIT has marked the corresponding

FIGURE 11
Sample videos linked to TutorIT. The first is a Flash file. The second, a YouTube video that provides an introduction to Column Subtraction.
FIGURE 12
Sample screen shots showing TutorIT before and after a student responds.

FIGURE 13
The current state of the Learner Model immediately after a student clicks on the correct column.

node “+” and changed the color of the node green. The Learner Model is normally only visible only to teachers or authors.

More generally, “+” means TutorIT knows that the student knows the indicated part of skill. “-” means TutorIT knows that the student does not know the
indicated part. “?” means the status is still unknown. TutorIT still must determine whether the node should be “+” or “-” for that student.

Figure 14 shows some additional screen shots: The top two screen shots show successive steps in solving one problem. The three on the bottom show steps in a more complex problem -- involving regrouping across zeros. The learner model on the right shows the status of a more advanced student. Notice that this student has not made any mistakes – likely because he or she has already been exposed to column subtraction.

Figure 15 shows what happens when the student makes a mistake -- TutorIT immediately shows what was done wrong, and provides instruction on what the student should have done.

Figure 16 shows what happens when a student demonstrates full mastery. TutorIT continues tutoring until a student demonstrates mastery of each and every operation and decision that the author believes students should know about column subtraction. Like a good (tireless) human tutor, TutorIT will continue tutoring until the student either demonstrates mastery or gives up. This is what we mean by guaranteed learning.
TutorIT need not stop just because a student has demonstrated mastery. Some solution methods, or parts thereof, may be judged to be so important that a TutorIT author (using AuthorIT) may require students to perform various combinations of steps in their head -- even after the student has demonstrated mastery. At this point, the student has demonstrated mastery of the full column subtraction.
process, and the author decided to require the student do all the regrouping et al
in his or her head – in this case within 15 seconds (see the left arrow in Figure 17).

Figures 18 and 19 show sample screen shots involving basic algebra. The
screen shots in Figure 20 involve solving quadratic equations. Here, the student
can choose to solve the problem in any of three ways – factoring on the left, using
the quadratic equation on the right and completing the square in the middle.

Tutorials need not be in mathematics, or any other subject. One on the left in
Figure 21 involves critical reading. The screen shots on the right show sample
screen shots in two other TutorIT tutorials. Currently only samples but they give
a broader sense of what can be done. One shows the Periodic Table in Chemistry.
In this tutorial, students can explore properties of various elements before answer-
ing questions.
FIGURE 18
Sample video introductions used in one of our TutorIT algebra tutorials.

FIGURE 19
Two screens shots showing signed numbers, one requiring the student to enter two kinds of answers, one clicking on the number line and one numerical.

The screen shots in figure 21 include TutorIT tutorials for critical reading, the Periodic Table in chemistry and preparing lawyers for the law boards. Law boards are required to practice in most states. If an aspiring lawyer gets a question correct, he is congratulated and told why his answer is correct. If incorrect, he is told why.
FIGURE 20
Two screen shots showing signed numbers, one requiring the student to enter two kinds of answers, one clicking on the number line and one numerical.

FIGURE 21
The screen shots on the left show a TutorIT tutorial on critical reading. The ones on the left show a tutorial on critical reading. The ones on the left show the Periodic Table in chemistry and the Law Boards.
Currently available TutorITweb tutorials include: Basic Facts, Whole Number Algorithms, Fractions, Signed Numbers, Complex Expressions, Math processes, Critical Reading/Logical Analysis, Simplifying Algebraic Expressions, Solving Linear Equations, Solving Simultaneous Linear Equations and Solving Quadratic Equations. Samples include: Word Problems, SAT Prep, Law Boards and the Periodic Table.

**Beta Testing and Classroom Use**— Most of these systems are now ready for beta testing. TutorIT tutorials obviously can be used by individual students. Another use is perhaps less obvious – precisely because TutorIT decision making as well as level of detail is so unique. TutorIT tutorials are not limited to simply presenting problems and/or questions, evaluating student responses and adapting future instruction. Like a good human teacher or tutor, TutorIT tutorials focus on what students must learn to be successful (see Scandura, 2014). TutorIT tutorials pinpoint the cognitive processes and decision making skills students must learn to be successful – at each point in time. At each stage, TutorIT presents precisely the information needed by the student to progress. The process continues until the student demonstrates full mastery.

Perhaps surprisingly, TutorIT does this more systematically than most teachers typically do in class. Irrespective of whether a teacher or instructional designer agrees with all aspects of the solution method being taught, a good deal of thought has gone into each TutorIT tutorial, generally speaking considerably more than most teachers have time to do.

Accordingly, TutorIT tutorials can be thought of (and used) as carefully analyzed lesson plans. At each stage of learning, TutorIT will do automatically what a good teacher might do. TutorIT will present questions and partially solved problems -- systematically step by step. TutorIT will ask students for answers, grade those answers and provide precisely the help needed where needed. The process will continue as long as the teacher desires, or until the class demonstrates full mastery. In the same way that TutorIT tutorials may be supplemented with YouTube videos or other media, the teacher also may intervene during the course of instruction with whatever additional or supplemental information may be desired.

In short, **TutorIT tutorials can be viewed and used as automated lesson plans.** We plan to make TutorIT tutorials available for teachers in class at no cost.

---

4 Those interested in evaluating these tutorials are encouraged to contact us at scandura@scandura.com.

5 For testing purposes, and to reduce the load of these sophisticated systems on our server, we initially are limiting use of TutorIT tutorials for teacher use in intact classrooms.
Teachers, and instructional designers working with teachers, are encouraged to contact us for further information.

**TutorIT Authoring Systems**-- Figure 22 provides an overview of the authoring process. Focusing on the left side shows that Structural Analysis (SA) is used to represent to-be-acquired knowledge in a form that can be used by TutorIT. Our AuthorIT authoring tools make it possible for authors to create TutorIT tutorials of increasing sophistication and scope. In each case, the author must create a representation of the to-be-learned knowledge (the Tutor content), show how problems are to be laid out on the Blackboard interface and specify how the content is to be delivered. Unlike other authoring systems, the same content may be delivered in many different ways, by simply selecting options in a dialog. For example, the same content may serve as the foundation for a dynamically adaptive (AKA intelligent) tutoring system, as an adaptive diagnostic test, as simple test or simple performance aid directing a student or trainee step by step in how to perform some arbitrarily complex task. Our recently patented methods enable TutorIT to perform all of this decision making automatically – without any programming whatsoever.

Figure 23 shows that four authoring systems are available for constructing or customizing TutorIT tutorials, and for integrating same. There are three increasingly sophisticated TutorIT authoring systems: EZauthor, Customizer and AuthorIT (itself) plus a Scope & Sequence tool for putting them together to create entire curricula.

**FIGURE 22**
This schematic shows how our authoring systems fit into the picture.
EZAUTHOR

EZAuthor is designed for use with unstructured content -- like the SAT, foreign language vocabulary, basic facts, etc (Figure 24). Most educators, including busy teachers can learn to use EZAuthor in a very short time. By the time this goes to press, it will be possible for instructional designers, curriculum developers, trainers, tutoring center owners to create their own TutorIT tutorials for use by their students, and also to submit them for broader use.

CUSTOMIZER

Customizer also is very easy to use (Figure 25). Any instructional designer can easily customize any of the professionally developed TutorIT tutorials currently available. For example, it is easy to adjust the wording of Instruction, Questions, Feedback or Corrective feedback. One can even add one’s own voice by loading in a recording or a new language (e.g., using Text-to-Speech). Customizer also enables authors to add one’s favorite YouTube videos, or other media.
FIGURE 24
Sample EZauthor screen.

FIGURE 25
Sample Customizer screen.
AUTHORIT

Whereas Customizer can be used to customize professionally developed TutorIT tutorials, AuthorIT goes much further. Designed by and for use by experts, AuthorIT currently is a full-powered desktop application. Figure 26 shows web-interfaces illustrating inherent capabilities. Given any content domain, AuthorIT empowers authors to perform arbitrarily detailed Structural Analysis to represent the knowledge or skill to be acquired.

In the case of structured knowledge, knowledge is cumulative in nature (as in column subtraction, solving quadratic equations, etc.). This is illustrated in the knowledge representation shown in Figure 26.

In short, AuthorIT is a full featured professional authoring system. AuthorIT is used to create hierarchical knowledge representations for essentially any content. It

FIGURE 26
Sample AuthorIT screen.
also is used to layout problems on the Blackboard through which TutorIT interacts with students. Figure 27 shows the blackboard itself. Problems (or problem schemas) are laid out in the center frame. This “Blackboard” interface (not the Blackboard LMS) represents the interface through which TutorIT and students are to interact. The left side represents the problem structure. To its right are properties of each problems element. These are used by TutorIT in deciding how to interact with students. The web interface in Figure 27 is illustrative.

AuthorIT is designed for use by authors with some degree of technical experience. The desktop version of AuthorIT was used to build our existing TutorIT tutorials. It is a powerful full featured authoring system -- but requires some degree of technical sophistication. The screen on the right (Figure 27) shows the beginnings of an easier to use web based version currently under development. Our ultimate goal is to enable professional instructional designers and other subject matter experts with minimal programming expertise to learn and use AuthorIT to create dynamically adaptive TutorIT tutorials with all of the features previously described.

SCOPE & SEQUENCE

The Scope & Sequence authoring system is used to define entire curricula composed of existing TutorIT tutorials (Figure 28). The author simply selects and specifies hierarchical relationships between available TutorIT tutorials. This is
done irrespective of how they were created – with EZauthor, Customizer or AuthorIT.

Scope & Sequence hierarchies may be run in either diagnostic or curriculum mode. Running in Diagnostic mode is used to determine where in a curriculum, any given student should begin (and progress). Run in curriculum mode will automatically take a student thorough and entire integrated courses or curricula. What makes Scope & Sequence tool particularly useful is that it is particularly easy to create custom Scopes & Sequences, made up of “official” and/or custom TutorIT tutorials. Run as a diagnostic, Scope & Sequence quickly determines which Tutorials a student has mastered and which not. Run in curriculum mode, it provides a general road map for entire curricula, ensuring that the student masters each tutorial in a curriculum before moving to the next. In all cases, TutorIT makes inferences about what individual students do and do not know, ensuring optimal efficiency in the learning process. Each student gets precisely the tutoring needed to progress in optimal fashion – moving ahead at the rate required to ensure prescribed levels of mastery throughout the process.

MANAGING DELIVERY MODES

AuthorIT also includes a broad variety of management options. These options enable teachers to customize delivery of individual tutorials (e.g., see Figure 29).
TutorIT delivery options, for example, include automatically enabling any given tutorial to serve as an efficient diagnostic engine or as a dynamically adaptive tutorial. These options also enable researchers to compare alternative instructional methods using the same content with unprecedented precision. Such research, for example, could help put to rest such age old questions as to whether and why it is better for students to discover knowledge on their own rather than being told.

**TICL On-Line.**-- Courtesy of Old City Publishing (formerly Gordon & Breach Science Publishers) researchers now have Free Access to ALL articles published in Technology, Instruction, Cognition & Learning (TICL). Just log in directly at www.TutorITweb.com. This feature also is directly available to members from the Association for Education Communications and Technology AECT and TICL SIG (of the American Educational Research Association (AERA) websites. Readers can also get background research publications on which TutorIT and AuthorIT rest.

**Concluding Statements and Opportunities.**-- TutorIT & AuthorIT represent a major advance in automated tutoring processes. These technologies build on: a) Decades of basic research by the author, his former students and colleagues in the cognitive sciences, AI, mathematics education, instructional design and software engineering, in all eight books, well over 200 refereed articles, along with decades
of technical development in both software engineering and computer based learning systems. It all began with the first educational software systems for the old the first Apple II computer, sold by MicroTutorII, Borg Warner Educational Systems and Queue, Inc. winner of the Inc. 50 fastest growing companies. Software engineering research in the interim made it possible to create the current AuthorIT and TutorIT systems.

TutorIT and AuthorIT technologies are based on newly patented methods (June 10, 2014). To optimize broad based benefits, these technologies are now being made available in a carefully measured way to instructional designers and subject matter experts in a broad range of disciplines – limited only by available resources. Those who are interested and/or positioned to help are encouraged to contact us for specifics at scandura@scandura.com:

1. If interested in field testing TutorIT, please send a short description of planned research. Please include content area, research interests and whether the research is funded or unfunded. How do you think TutorIT or AuthorIT would help?
2. If interested in developing TutorIT tutorials in your own areas of expertise, please send a short description of the TutorIT tutorial you would like to create. Please include how you plan to use the tutorials and your preference as to authoring tool: EZauthor, Customizer, AuthorIT or Scope&Sequence? Please also include an assessment of your (or team’s) computer skills (programming skills, etc.)

REFERENCES

Potskta, Joseph; Massey, L. Dan; Mutter, Sharon, eds. (1988). Intelligent Tutoring Systems: Lessons Learned. Lawrence Erlbaum