The Computer in School: Tutor, Tool, Tutee

by Robert P. Taylor, II, Columbia University Teachers College

Editors' Note:

Since it was published in 1980, a copy of Robert Taylor's book, *The Computer in the School: Tutor, Tool, Tutee*, has had a place of honor on the shelf above my desk. This book of readings shaped the thinking of a generation of educational innovators.

We have previously republished seminal works from this text by Alfred Bork (http://www.citejournal.org/vol2/iss4/seminal/article1.cfm) and by Arthur Luehrmann (http://www.citejournal.org/vol2/iss3/seminal/article1.cfm), who coined the term "computer literacy." We now have the pleasure of republishing the introduction to *Tutor, Tool, Tutee* by Taylor himself, who began in this fashion:

> For the foreseeable future, computing will play an increasingly important role in human learning. However, no one yet knows exactly how great that role will eventually be, or precisely what form it will take. (p. 1)

Dave Moursund, last year's recipient of the SITE *Lifetime Achievement* award, wrote the original introduction to *Tutor, Tool, Tutee*, concluding:

> Anyone wondering either what role computers can play in education or why their incorporation into the curriculum should receive highest priority should read this book from cover to cover, as soon as possible. (Preface, p. viii).

Taylor framed potential uses of the computer as (a) tutor, computer assisted instruction in which the computer teaches the child, (b) tool, in which the computer amplifies ability to address academic tasks, and (c) tutee, in which students learn by programming (tutoring) the computer.

Nearly a quarter-century later, we are well into that future that Taylor envisioned, with a distance to go. This midpoint offers a useful vantage point for consideration of the roots of the discipline, and where we may wish to go in the future.

We also encourage you to read Taylor's companion reflection piece, entitled "Reflections on *The Computer in the School*," by clicking on the "Read related articles" link on this page.

Introduction

FOR THE FORESEEABLE future, computing will play an increasingly important role in human learning. However, no one yet knows exactly how great that role will eventually be, or precisely what form it will finally take.

Few people outside the computing community have anything but the vaguest concept of either that role or its form, for two reasons. First, technical innovation has come so fast in computing that even the expert can barely keep up with it. Second, the effort to apply computing to education is less than 25 years old and, though an immense body of work has already accumulated, it has been poorly publicized to the wider public. The media generally have tended to sporadically overrate a small subset of developments in this field while ignoring or giving only superficial treatment to the rest. Nevertheless, with the advent of microprocessors and the prospect they afford of widely available computing power, thousands of educators and parents are beginning to seriously ponder what the role of computing will be in human learning and what action they can and should take to affect it.

This book is meant to help them. It does so by making readily available a number of articles about the application of computing to education. Their authors, all pioneers in this field, have been directly or indirectly responsible for a great deal of work in this area in the past decade, and the articles included reflect upon and report that work. Despite the extensive innovation in computing, much remains the sameparticularly in the way computer logic structures are related to human thought structures. Thus what has already been examined and implemented can be of surprising relevance. Teachers and other educators now entering this field may imagine they are breaking new ground when in fact they are not. Reading these essays will discourage such fantasies. By presenting past accomplishments, the essays will encourage the new entrant to use them and build upon them rather than to blindly create them anew. The articles are therefore a key to the future as well as a record of the past.

Such writings should be read by anyone interested in computing and education because they suggest what has already been accomplished. To know what has already been accomplished is the first step, whether one merely wishes to find out what the field is like or whether one wishes to determine the point from which to begin one's own work. However, simply plunging into the field and attempting to assimilate the ideas may not work. Initially some conceptual help may be needed.

Approaching the Diverse, Technically Foreign Area of Computing in Education

The application of computing to education encompasses a range of complex activity, formidable in its apparent diversity even for those who are simultaneously both computer specialists and educators. Approaching such a complex area for the first time, especially as a computing novice, can be very confusing. This book attempts to minimize the unnecessary confusion three ways. First, by limiting the number of authors included, it arbitrarily limits the diversity of what is presented. Second, by presenting only articulate spokesmen, the issues and the work discussed are presented in an intelligible fashion. Third, by introducing a succinct framework (tutor/tool/tutee) for classifying all educational computing, the book provides the reader with a simple scheme for intellectually grasping a somewhat chaotic range of activities.

The major role of this introductory essay is to present the tutor/tool/tutee strategic framework. The basic framework and a summary set of comments on each of the five authors are presented. Then, the application of the framework is demonstrated, in terms of the work of those five.

To assist the reader interested in understanding more of the context of the author's work, a brief biographical sketch precedes the presentation of that author's articles. To assist the reader interested in reading more of a particular author's work, a short selected biography for that author can be found at the end of the book

Tutor, Tool, Tutee The Three Modes of Using Computing in Education

The framework suggested for understanding the application of computing in education depends upon seeing all computer use in such application as in one of three modes. In the first, the computer functions as a *tutor*. In the second, the computer functions as a *tool*. In the third, the computer functions as paychecks a *tutee* or student.

The Computer as Tutor

To function as a *tutor* in some subject, the computer must be programmed by "experts" in programming and in that subject. The student is then tutored by the computer executing the program(s). The computer presents some subject material, the student responds, the computer evaluates the response, and, from the results of the evaluation, determines what to present next. At its best, the computer tutor keeps complete records on each student being tutored; it has at its disposal a wide range of subject detail it can present; and it has an extensive and flexible way to test and then lead the student through the material. With appropriately well-designed software, the computer tutor can easily and swiftly tailor its presentation to accommodate a wide range of student differences.

Tutor mode typically requires many hours of expert work to produce one hour of good tutoring, for any or all of several reasons. (a) As intuitive beings, humans are much more flexible than any machine, even a computer. (b) Creating a lesson to be delivered by a human tutor requires less time because it omits much of the detail, relying upon the spontaneous improvisation and performance of the instructor to fill in both strategy and substance at the time of delivery. (c) Computers are still relatively crude devices and the only means we have of programming them are awkward and time-consuming. (d) Human instruction rarely aims to accommodate individual differences because the normal classroom situation prohibits such accommodation; hence lesson preparation and design are simpler and swifter. Because such accommodation is possible with the computer as tutor, the substantive and strategic details needed to individualize the lesson tend to get included, thus often greatly

substantive and strategic details needed to individualize the lesson tend to get included, thus often greatly lengthening lesson design and preparation time.

The Computer as Tool

To function as a *tool*, the classroom computer need only have some useful capability programmed into it such as statistical analysis, super calculation, or word processing. Students can then use it to help them in a variety of subjects. For example, they might use it as a calculator in math and various science assignments, as a map-making tool in geography, as a facile, tireless performer in music, or as a text editor and copyist in English.

Because of their immediate and practical utility, many such tools have been developed for business, science, industry, government, and other application areas, such as higher education. Their use can pay off handsomely in saving time and preserving intellectual energy by transferring necessary but routine clerical tasks of a tedious, mechanical kind to the computer. For example, the burdensome process of producing hundreds or even thousands of employee paychecks can be largely transferred to the computer through the use of accounting software; the tedious recopying of edited manuscripts of texts or even music can be relegated to the computer through word or musical notation processing software; the laborious drawing of numerous intermediate frames for animated cartoons can be turned over to the computer through graphics software; or the fitting of a curve to experimental data can be done by the computer through statistical software.

To use the computer as tutor and tool can both improve and enrich classroom learning, and neither requires student or teacher to learn much about computers. By the same measure, however, neither tutor nor tool mode confers upon the user much of the general educational benefit associated with using the computer in the third mode, as tutee.

The Computer as Tutee

To use the computer as *tutee* is to tutor the computer; for that, the student or teacher doing the tutoring must learn to program, to talk to the computer in a language it understands. The benefits are several. First, because you can't teach what you don't understand, the human tutor will learn what he or she is trying to teach the computer. Second, by trying to realize broad teaching goals through software constructed from the narrow capabilities of computer logic, the human tutor of the computer will learn something both about how computers work and how his or her own thinking works. Third, because no expensive predesigned tutor software is necessary, no time is lost searching for such software and no money spent acquiring it.

The computer makes a good "tutee" because of its dumbness, its patience, its rigidity, and its capacity for being initialized and started over from scratch. Students "teach" it how to tutor and how to be a tool. For example, they have taught it to tutor younger students in arithmetic operations, to drill students on French verb endings, to play monopoly, to calculate loan interest, to "speak" another computer language, to draw maps, to generate animated pictures, and to invert melodies.

Learners gain new insights into their own thinking through learning to program, and teachers have their understanding of education enriched and broadened as they see how their students can benefit from treating the computer as a tutee. As a result, extended use of the computer as tutee can shift the focus of education in the

classroom from end product to process, from acquiring facts to manipulating and understanding them.

Five Pioneers of the Application of Computing to Education

Though many computer scientists have broad general interests, most have only a few really dominant specific interests and those few are typically shaped and informed by the individual scientist's particular point of view. Before looking at the work of our five authors in terms of the tutor/tool/tutee framework, therefore, it may be helpful to summarize the dominant interests and point of view of each.

Alfred Bork

Bork is a physics professor at the University of California at Irvine where he has directed the Physics Computer Development Project for a number of years. That project produces computer-based material that can serve as the primary source from which first year physics is learned at Irvine. As this implies, Bork's major interest is the application of computing to physics instruction. His work strongly emphasizes concept mastery, self-paced instruction, and computer-resident testing. Though his work beautifully demonstrates how computer/student dialogs can function and how graphics can be carefully and integrally used to enhance these dialogs, he does not argue that all instruction should be computerized, even in a subject like physics. Bork sees stand-alone computers as the major vehicle in the new generation of computer-assisted learning. He is also careful to point out repeatedly that good software in any reasonable quantity is more likely to be developed by software factories or institutes than by individual professors, teachers, or researchers.

Thomas Dwyer

Dwyer is a computer scientist and educator at the University of Pittsburgh, who for a number of years ran a series of projects involving high school and junior high school teachers and students. The projects were characterized by an exploratory approach to using computing, one which tended to depend upon and generate a new way of looking at learning in the school, one which Dwyer himself dubbed "Solo-Mode" learning. In such learning, the framework is provided by the teacher but the pupil must work autonomously, learning to "fly solo." This mode Dwyer contrasts with the more usual classroom situation that keeps the teacher in complete control and has the student "flying dual." Dwyer's work stresses an heuristic, exploratory approach based on principles rather than a closed one based upon a formula of what to do. He places a heavy dependence upon the teacher as a supportive human being, stresses that the teacher is crucial, and addresses teacher education as a major concern of any attempt to use computing broadly and creatively in the schools. Though much of the solo work dealt with math and physical science, Dwyer's work through solo and elsewhere also applied computing to other subjects, including music.

Arthur Luehrmann

Luehrmann is now associated with the Lawrence Hall of Science at Berkeley, where he is directing projects to integrate computing into museum science exhibits to make them interactive, and projects to teach computing to a broad, general public served by the museum. Prior to going to Berkeley, he was a professor at Dartmouth and

was involved in many successful projects there, applying computing to instruction. As several of his article titles suggest, his strongest emphasis is upon the computer as a new and fundamental technology worthy of study on its own. He sees the mass impact of this new technology as very substantial and stresses the need for popular literacy, the need for everyone to acquire programming skills, and the need for a good stand-alone personal computer. Though trained as a physicist, Luehrmann's work has dealt with applying computing in many instructional areas, not simply those related to the physical sciences.

Seymour Papert

A professor of mathematics and an educator at M.I.T., Papert is best known for his development of the LOGO language and its application to teaching computing and mathematics to young children. His major thrust definitely is to teach a way of mathematical thinking that young children can intuitively master. By encouraging anthropomorphizing, play, and intuitive guesswork he tries to capitalize upon the existing insights and mental frameworks of children. His strong attention to how and what children are thinking is in part based upon his extended association with Piaget in Switzerland. Papert's work has been exploratory, centering on children's use of computing, emphasizing almost exclusively the child learning to program. It has included imaginative use of robots, graphics, and sound as a child-attractive alternative to traditional textual output. Throughout, the computer tends to be used to create a problem-rich environment, presenting the child with interesting, challenging problems that require a computer for solution.

Patrick Suppes

Suppes is a philosophy and mathematics professor at Stanford, where he pioneered the development of computer-assisted instruction. His work stresses the applicability of the computer to skill areas such as mathematics, logic, and language. It aims to produce complete courses of instruction to be delivered by the computer. He has always stressed how little we know about learning but has carefully used what is available to design a considerable quantity of computer-assisted instruction. Much of this instruction is on the market and in wide use on minicomputers. For example, whole systems stressing mathematics and language arts skills are commercially available through the Suppes-founded Computer Curriculum Corporation. His work stresses individualized learning and increased educational productivity.

Using The Tutor/Tool/Tutee Framework

Now that the framework and the five pioneers have been introduced, let's look at some of their writings in terms of the three modes of that framework. We will only cite a few of the articles included in this book and will discuss none in detail; our aim will be to merely suggest the framework's utility. The authors speak well, even brilliantly, for themselves and do so clearly enough to need no explanation. The framework must be accurate enough for the reader to make the associations between it and the articles for himself or herself.

Examples of the Tutor Mode

Historically, this mode has its roots in programmed instruction. However, when properly deployed it is far more

Historically, this mode has its roots in programmed instruction. However, when properly deployed it is far more flexible than any book- or material- based programmed instruction. For one thing, in tutor mode, the material can be presented interactively, and dynamic graphics and other sophisticated teaching aids can be integrally used. For another thing (as pointed out earlier), in tutor mode the performance history of one or more pupils can be collected and stored, then subsequently used for evaluating the material and as a basis for routing a particular pupil through the material. At the same time, this mode can be designed to move the student at a wide range of speeds and to be interruptible more or less at the student's convenience. Though the label has been applied to broader applications than just this one of using the computer as a tutor, this mode has often been called CAI (Computer-Assisted Instruction), probably because the ancillary tasks it performs are similar to those that could be performed by ideally competent teaching assistants.

The work of Bork and the work of Suppes both exemplify the tutor mode at its best. All of their included articles deal with this model. Bork has concentrated much of his thinking on how best to develop good tutor material for physics instruction, while Suppes has developed material for a wide range of subjects. Both have used the computer to store, analyze, and act upon student results, and both have used such sophisticated peripheral devices as audio or graphics to maintain student involvement and enrich the nature of the tutoring.

The time period encompassed by Suppes' work alone reveals how many years have already gone into work on tutor mode computing applications. The breadth of those applications is suggested by his prepared statement for the Congressional hearings on Computers and the Learning Society, "The Future of Computers in Education." No one who reads either that or his Marseilles conference article, "Impact of Computers on Curriculum in the Schools and Universities," will mistakenly believe that CAI on microprocessors is completely new. One gets a sense of how much work goes into producing good material in this mode by reading Bork's detailed, "Preparing Student-Computer Dialogs: Advice to Teachers," or by reading his very thoughtful "Learning about Graphics." Anyone who would produce good tutor mode material should certainly be thoroughly familiar with both these pieces.

This mode has had both its advocates and its critics. Criticism from those who are deeply involved in computing and education is usually directed at those making extreme claims about the positive benefits to be derived from tutor mode computing. Good criticism of this kind is exemplified by Luehrmann's "Should the Computer Teach the Student or Vice-Versa?" Neither he nor any other pioneer, however, would argue that tutor mode computing should not have a significant place in education.

Examples of the Tool Mode

Tool mode is probably seen to be the major mode of computer use by most people outside computing and education. Because it receives considerable attention and encompasses such a wide range of activities, tool mode computing is popularly seen as synonymous with computer use, period. For example, most business data processing, whether routine accounting or word processing and office automation, uses the computer as a tool. Thus the school's administrative activities use the computer in a tool mode, from payroll and inventory to pupil scheduling and grade reporting. Even library automation involves the computer merely as a tool.

In tool mode, the computer provides a service that the user needs and more or less knows how to use. It is not primarily a teacher or tutor as in the tutor mode; it is not user-alterable and not a set of raw building components as might be provided under tutee mode. Use of the computer in tool mode may teach the user

components as might be provided under tutee mode. Use of the computer in tool mode may teach the user something during use, but any such teaching is most likely accidental and not the result of any design to teach. In tool mode computing, the user can only explore activities and ideas for which the tool at hand is appropriate; one can explore musical inversion with a composition and playback tool, but not a word processing tool or a regression analysis tool.

Most people in computing and education frequently and creatively use the computer in a tool mode, because of their everyday familiarity with computing capabilities. However, possibly because they are familiar with such use, possibly for other reasons, most computing-and-education people do not see this mode as something they want to focus primary energy upon. All five pioneers whose work is included in this book fit this general position. All assume heavy use must be made in education of tool mode computing; none advocates it as most important or focuses his own major interest upon it.

All five have advocated the use of the computer as a calculator and a word processor, and all have advocated various other tool mode uses as well. Bork and Suppes, for example, suggest that the computer as a calculator and record keeper should be available simultaneously to anyone using the computer in the tutor mode. Bork further suggests that various graphic tool capabilities should be similarly available. Dwyer, Luehrmann, and Papert all argue that various tool capabilities should be available to be utilized by anyone who needs them in exploratory problem-solving, writing, or anything else. Dwyer, for example, argues that the student flying solo on the computer would freely utilize many of its tool capabilities as he pursued his overall project.

Examples of the Tutee Mode

The tutee mode is the one upon which Dwyer, Luehrmann, and Papert focus their energies. This book reflects that.

> One of the early and still one of the best arguments for this mode of computer use is Luehrmann's "Should the Computer Teach the Student, or Vice- Versa?" In it, he argues that in teaching the computer, the child learns more deeply and learns more about the process of learning than he or she does from being tutored by software written by others. Papert extends the argument by suggesting how children using the computer as a tutee may learn more of what they should be learning of mathematics than they can in classrooms without computers. This position is clearly articulated in both "Teaching Children Thinking" and "Teaching Children to Be Mathematicians vs. Teaching about Mathematics." Dwyer extends the concept in a different way in "Some Principles for the Human Use of Computers in Education," defining his now well-known concept of solo-mode computing and showing how it relates to the total curriculum question.

In these essays and others, all three suggest that in using the computer as tutee, the learning the child experiences is qualitatively different than he or she might otherwise experience in any school setting. None downgrades the role of the teacher in a tutee mode environment, but all see it as different from the teacher's typical role now. Dwyer states the case very well for all three in his "Some Thoughts on Computers and Greatness in Teaching."

Papert suggests that the computer as tutee can, with appropriate graphic and robotic capabilities, serve as a means to enable the child to link his or her experience to the deep, fundamental mathematical ideas we most want children to learn. In his "Personal Computing and Its Impact on Education," he suggests that this may be the only way to avoid having most children spend most of their time struggling within the dismal reality of dissociate learning. Both in that essay and in "Computer-based Microworlds as Incubators for Powerful Ideas," Papert contrasts dissociate learning, the attempt to somehow internalize great quantities of information apparently of no use in the child's world, with a more natural learning that resonates with the child's experience.

Using the Framework Without Becoming Blinded by It

The tutor-tutee-tool framework has been presented to help those who would like to get an organized initial grasp on an apparently complex field. It will serve to overcome hesitation and initial trauma. Of course it can and should be used later, so long as it conveniently provides insight. It is a reasonably broad framework and suffers from no more shortcomings than any other schema or typology. As such a schema though, it can divert attention from relevant insights when used too slavishly.

Reasonable alternatives to this framework certainly can be advanced. They might be entirely different or be simple extensions of it. For example, I seriously considered extending the tutor, tool, tutee framework to include a fourth mode, making it tutor, tool, tutee, toy. There are numerous games, simulations, and models of many sorts that one spontaneously classifies as toysthey appear to have been created above all to play with and enjoy, whatever other merits they might possess. I finally decided against that extension, though, believing that such software is just as well subsumed under one or more of the earlier three modes. The point is, one need not be bound by this framework. If modifying it or replacing it by an alternative framework helps with the process of internalizing the ideas advanced in the various articles, then such modification or replacement is in order.

The articles are the main thing. They provide a good introduction to computing and education. If you prefer a different order, follow it. If the framework gets in your way, disregard it. But do read the articlesall 19.

Submit a Formal Commentary

Archive
Issue
Archives
Volume 18
Issue 1
Issue 2
Issue 3
Issue 4
Volume 17
Issue 1
Issue 2
Issue 3
Issue 4

More Archives

The Computer in the School – Tutor, Tool and Tutee Taylor. This article looks a the framework for classifying educational computing. Tutor – the computer acts as a tutor – it has been programmed in a subject, presents material and then the student responds and the computer evaluates the response. Tool – the computer is sued as a tool such as calculations, text editor, map making, applying graphs etc Tutee – training the computer – students must talk to the computer in language that the computer understands - the student must learn how to program the computer and how to logically describe the t Tutor, Tool, Tutee. Tutor software is designed to help the student acquire a specific skill—number facts, for example. The content and instructional style of this type of software covers the range from simulation of a task (running a lemonade stand, for example) to rote drill and practice. Tool software includes word processors, databases, spreadsheets, graphics programs, music composition tools, etc. The remaining one or two computers in the school are often put on movable carts, like movie projectors, to be wheeled from room to room on an "as needed" basis. To see how limiting this is, imagine what impact the pencil would have on students if they could only use one for 20 minutes a week—and they would have to go to a "pencil lab" to find one. Download The Computer in the school : tutor, tool, tutee edited by Robert Taylor leave here couple of words about this book: Tags: Art Early Renaissance.

The Computer in the School: Tutor, Tool, Tutee. New York: with Microcomputers. Leicester: Council for Educational Technology. processor.Â A framework is presented for considering computers in education which identifies three functions of a computer: as a tutor, as a tool, or as a student (tutee). A computer's tutor function requires expert programming so that flexible computer-assisted instruction can be provided to students. A computer's tool function requires only that some useful capability (such as statistical analysis) be programmed into the computer. In the student or tutee function of a computer, a human tutor teaches the computer, thereby enhancing human learning and reducing software costs. COMPUTER AS A TOOL, TUTOR AND TUTEE By: Nur Fatihah binti Abdul Rahman, Faculty of Education, B. Ed TESL (2014/2018), National University of Malaysia, Bangi, Malaysia.Â Computer provides some functionality that makes the learnersâ€™ task easier. Besides, it also helps in the teaching profession; as in it saves the learner time and allows them to focus on her intellectual energy on higher order tasks. The teachers also do not have to spend their money on tuition centers because computers includes the software to help teachers to teach; for examples, Adobe Reader, Microsoft Office and such. The advantages that I came across during my research are that, computer being a tool provides practice in the skills at inquiry and problem saving.