Technology in Education
Virginia Society for VSTE
www.vste.org

What Makes Educational Software Educational?*

by Keith E. Polonoli

To understand what features make a piece of educational software effective, it is first necessary to recognize that the underpining premise for using software in the classroom lies in having students learn something (Pillay, Brownlee, &Wilss, 1999). Of course, some software is more successful at helping children achieve learning goals than others. With this in mind, we must ask ourselves what features account for this difference in learning.

Crozier (1999) reports that educational software can be thought of as falling into one of four loosely defined categories:

1. drill and practice, which offers repetition or practice of a particular skill;
2. problem solving, which presents a scenario where a child needs to provide a solution to solve a problem;
3. simulation, which presents events in a number of virtual environments; and
4. tutorial, which presents a lock-step approach to teaching a concept.

It must be noted that much of the software produced today is actually a combination of two or more of the four previously discussed categories; therefore, many of the features will overlap, affecting hybridization.

Thankfully, much research has focused on identifying features that makes software educationally successful. From this body of research, I have identified the following properties: learning theory, gaming features, cultural sensitivity, and eliciting a learner response. Each factor will be discussed in the proceeding paragraphs with the hope that the reader will apply this information when selecting software for classroom use.

** This article first appeared in the Fall 2000 edition of the VSTE Journal (Vol. 15, No. 1). It is reprinted here for the benefit of our readers and to create an electronic archive of an important article.*

Educational Software, continued

Features of Quality Educational Software

Learning Theory

It stands to reason that if software is to be used by a teacher in the classroom to teach content, the software features should have their foundations in some accepted learning theory; otherwise, why bother to discriminate between recreational and educational software. Gray (1990) states learning theory can be viewed as falling into two general categories: behavioral and cognitive. Being professional educators, classroom teachers should possess some knowledge of the major tenets that define both categories. In addition, educators should be able to identify what strategies purported by these two genres will aid in reaching established learning outcomes.

Behavioral Theory

Behavioral theorists advocate that learning is a result of the association of a stimulus and a response. For example, B. F. Skinner's theory of operant conditioning is based on the idea that learning is a function of a change in behavior (Skinner, 1954). Skinner states that changes in behavior are the result of an individual's response to an event (stimuli) that occurs in the environment: A response will produce a consequence. When a particular Stimulus-Response (S-R) pattern is reinforced (rewarded), the individual is conditioned to respond.

Many drill and practice programs successfully use this theory. Software grounded in behaviorist theory is quite effective when continued practice is needed to perfect a specific skill. AlgeBlaster, the popular algebra tutorial, is a software program grounded in behaviorist theory and is popular with math teachers to reinforce basic algebra skills.

Cognitive Theory

Cognitive theory differs from behaviorist theory because it regards learners as sources of plans, goals, and emotions rather than products of incoming environmental stimuli (Woolfolk, 1993). There are many cognitive theorists, but the common theme that runs through all of their work is that learning is an active process in which learners develop new ideas and concepts from interaction with the environment.

Learners will use past knowledge to bridge the gap from what is known to what is to be learned. When viewing educational software in this context, simulation software, which models real-life events, is definitely rooted in cognitive theory. Simulation software requires students to think critically and make decisions based on limited knowledge. MECC's *Oregon Trail* is an excellent example of software grounded in cognitive theory.

Educational Software, continued

One aspect of cognitive theory that should be visible in good educational software is the acknowledgment of learner differences. In general, learning style theory takes into account the way that an individual concentrates on, processes, internalizes, and remembers new academic information and skills (Shaughnessy 1998, p. 141).

Educational software should not only allow the instructor to adjust the software's content to individual student ability levels, but it should also have the capacity to present content based upon the student's learning style.

After applying learning style theory, classroom practitioners have reported statistically significant increases in student test scores and grade point averages (Shaughnessy 1998).

Hence, quality educational software will take into account that student learning will vary with age, gender, and processing preference. Exceptional educational software will not only allow the student to operate within their preferred learning style, but it should also expose children to situations where they are exposed to content delivered in a manner that is outside of their chosen style. This will aid them in flexing their style and develop the ability to use processing strategies that may otherwise never be employed.

This paper is not to tout the particulars of one learning theory over the other. Both schools of thought offer a sound framework in which to deliver instruction. A problem arises, however, when an instructor cannot identify the paradigm from which the software was developed. Consequently, it probably was not developed from an educational perspective; therefore, it would be best to avoid it.

Gaming Features

Discussing gaming features in a paper hoping to define the principle factors that define good educational software may seem a little strange, but according to a study done by Pillay Brownlee, & Wilss (1999), gaming offers positive learning benefits. The Pillay, et al. study concentrated on investigating the cognitive process as children played *Pilot Wings,* a helicopter flight simulation game. Each child was paired with an expert analyst who was familiar with the game and was trained in qualitative data gathering techniques.The results of this study indicate that children engaged in recreational game play exhibit the same cognitive processes that are found in other problem solving systems using technology. It was found that with the limited instructions given to the participants, inductive reasoning was the primary method of decision making while playing the same. Hence, many strategies that are employed in recreational gaming software have strong positive cosnitive effects.

Educational Software, continued

Roblyer, Edwards, & Havrilik (1997) make the statement that a classroom without games and fun would be a very boring classroom. In a review of the effectiveness of games for instructional purposes, it was found that games are more interesting than traditional instruction (Randel, Morris, Wetzel & Whitehill, 1992). With this in mind, it is reasonable to assume that educational software that use gaming strategies will foster more fervent pupil interaction. More intense involvement and longer contact periods with a learning activity is something that good classroom instructors are constantly trying to accomplish, and gaming is one way of achieving this goal.

Cultural Sensitivity

Many of today's software packages lack accuracy and sensitivity to non-mainstream cultures (Miller-Lachmann, 1994). Because of the power of multimedia software to convey sounds, pictures, movies, and animation, it is imperative that educators pay particular attention to the manner in which cultures are presented to students (see the chart with 10 questions Miller-Lachmann suggest educators ask when assessing the cultural sensitivity of software).

Mei-Yen, Walker, & Huang (1999) examined several educational software packages that were produced for the global market. American and Taiwanese educators were recruited to evaluate software packages produced both in the United States and Taiwan that were marked for global distribution. Both groups were given an identical 21-item scale to assess the software packages; the instrument also had a series of open-ended questions for personal responses. The results of the study purported that only the Asian products were truly developed fora global audience. In comparison, the American products were developed for a Euro-American market.

One particular piece of evidence reported by the authors that substantiate

Questions that Miller-Lachmann (1994) suggest educators ask when assessing the cultural sensitivity of software:

- What is the purpose of presenting other cultures?
- Do people of color and their cultures receive as much attention as people of European descent?
- How accurate is the presentation?
- Are the language and terms used in the package appropriate?
- Do the illustrations or sounds distort or ridicule members of other cultures?
- Does the program present a culture's diversity and complexity?
- Who are the characters, and what roles do they play?
- From whose perspective is the story presented?
- Does the documentation allow instructors to go beyond the program itself?
- Should some simulations not be played because of the lack of cultural sensitivity?

Educational Software, continued

this was the fact that all of the Asian software had the option of the user choosing Chinese, English, or French; the software produced in the United States had no such feature, English was the only language choice. In addition, very little evidence was seen regarding the referencing of non-Euro-American characters in the software produced in American.

Good software should only propagate truth — truth in content and truth in the portrayal of the culture and characters represented in the software. Using the suggestions previously mentioned will aid the educator in selecting software that is culturally sensitive. Choosing culturally sensitive software will aid in halting the perpetuation of pejorative stereotypes that currently exist regarding non-mainstream cultures.

Eliciting maximum usability from a piece of educational software*:

- Be certain that the software is designed for the appropriate grade level.
- Look for software that has a high level of interactivity and learner feedback.
- Look for game-like features in the software.
- The software should represent the child's world, not the adult's world.
- Be sure that the software portrays characters in a respectful truthful manner; that is, be certain that the software is free of racial or gender prejudices and stereotypes.
- Look for software that has a friendly interface; uses simple, easy-to-understand dialogue; exhibits consistency in navigation buttons, program exits, provides shortcuts and ready access to help; and allows learner customization.

* Amended from Robertson's suggestion (p. 261)

Emotional Response

Weinstein (1997) tells us that frustration—such as having our hard drives freeze or our software crash—is to be expected when using the computer. Nevertheless, great joys such as solving complex statistical problems with a mouse click or connecting to the Internet to access boundless sources of information are also to be expected. From an emotional standpoint, educators should look for software that has the capacity to frustrate the conventional problem-solving mind-set of students.

The software should cause a mild level of frustration in the learner, not so much as to turn the learner off, but just enough to cause a mild state of cognitive dissonance that will make the content challenging. When using a computer, Weinstein states, the joy lays not so much in accomplishing a task, but in transcending to the point of achieving a new level of understanding. Whether the instruction is computer-based or delivered in a traditional didactic mode, transcending beyond simple task completion to an intimate level of understanding should be the central tenet of education.

Educational Software, continued

Instructor Responsibility

Instructional software may include all of the previously discussed features, but if it is not integrated into the curriculum in a purposeful manner, it is worthless. With so much pressure being put on classroom educators to use technology in their teaching, it is important for them not to succumb to using computers/computer software in the classroom just for the sake of using technology—it loses purpose.

Usability is the term used to describe the quality of user-interface (Robertson, 1994 of a system. It is a measure of how well a technology is used for some purpose by humans. Although the term comes from the field of industrial engineering, Eason(1988) extends the meaning to define how well planners (teachers) institute the technology for the users (students) to gain the most learning without undue strain on their capacities. See sidebar list of six suggestions for eliciting maximum usability from a piece of educational software.

Conclusion

Effective educational software packages all share four elements:

1. their conception is grounded in accepted learning theory,
2. they employ gaming features;
3. they are culturally sensitive, and
4. they possess the ability to elicit an emotional response from the learner. The instructor should carefully review the package to be sure these details are present before adopting software for classroom use.

However, even if the software contains all of the previously mentioned features that define it as educationally sound, the idea of usability must be addressed; the human element cannot be ignored. As educators, we must acknowledge the fact that usins an effective educational software package will not compensate for poor instructional planning. Quality planning will, however, allow a savvy teacher to rise above the sea of mediocrity and become a better instructor when pedagogically sound software is used in an appropriate manner in the classroom.

Educational Software, continued

References

- Crozier, J. (1999, March/April). Ways to evaluate educational software. *Media & Methods, 35*(4), 50-52.
- Eason, K. (1988). *Information technology and organizational change*. Philadelphia: Taylor & Francis.
- Gray, R. (1990). Microcomputer educational software design and development: Lessons from learning theory. *International Journal of Instructional Media, 17*(2), 109-119.
- Mei-Yan, L, Walker, D. F., Huang, J. (1999). Do they look at educational multimedia differently than we do? A study of software evaluation in Taiwan and the United States. *International Journal of Instructional Media, 26*(1), 31-43.
- Miller-Lachmann, L. (1994, November). Bytes & bias: Eliminating cultural stereotypes from educational software. *School Library Journal* 23-28.
- Pillay, H., Brownlee, J., & Wilss, L. (Fall, 1999). Cognition and recreational computer games: Implications for educational technology. J*ournal of Research on Computing in Education, 32*(1), 203-217.
- Randel, J., Morris, B., Wetzel, C. & Whitehill, B. (1992). The effectiveness of games for educational purposes: A review of recent research. *Simulations and Games. 23*(3). 261-276.
- Robertson, J. W. (1994). Usability and children's software: A user-centered design method. *Journal of Computing in Childhood Education, 5* (3/4), 257-251.
- Roblyer, M. D., Edwards, J., & Havrilik, M. A. (1997). *Integrating educational technology into teaching*. Upper Saddle River, NJ: Prentice-Hall, Inc.
- Shaughnessy, M. F. (1998, January/ February). An interview with Rita Dunn about learning styles. *Clearing House, 71*(3), 141-146.
- Skinner, B. F. (1954). The science of learning and the art of teaching. *Harvard Educational Review, 24*(2), 86-97.
- Weinstein.N. (1997, November/ December). Socrates at the terminal. *Educom Review, 32*(6), 52-55.
- Woolfolk, A. (1993). *Educational psychology* (5th ed.). New York: Simon & Schuster, Inc.

Educational Software, continued

About the Author

At the time of writing this article, Keith E. Polonoli was an elementary technology resource teacher for Frederick County Public Schools, Winchester, VA. Keith is now an instructional technologist for the University of Pittsburgh at Greensburg, PA. You can reach him at: polonoli@pitt.edu