Gary G. Bitter

Choosing the proper tools

How does a teacher set about acquiring the software to use with his or her computers? Bitter defines three classes of instructional software, advises on how one should survey the market, and, focusing on the need for teachers to help each other with their evaluations, discusses in detail the kinds of consideration that must be represented on any scheme for evaluating software that a group should design and use. This participational approach is the best guarantee that "drill and practice" software will eventually lose its dominance over the market.

Today teachers of all disciplines and grade levels are facing an electronic revolution in education. Along with the traditional chalkboards and audio-visual equipment common in classrooms, many teachers now find themselves coping with the Computer Age on their own territory. How they choose to handle these sophisticated machines will, in part, determine how adequately prepared their students are to live and work with the computers that will be commonplace in their future. In an attempt to make their students "computer literate" - that is, able to understand the fundamentals of computer usage and to apply this knowledge with relative ease - teachers are attending conferences and seminars on educational computing, taking classes at local colleges and universities, and subscribing to periodicals that help them keep abreast of current trends in classroom computing.

Each year a steadily increasing number of teachers are given the opportunity to incorporate CAI (computer-assisted instruction) into their traditional curricula. Implementing computer education brings with it a mixture of reactions. Many teachers are excited at the prospect of using the latest in educational technology, but their excitement is often

accompanied by confusion over the overwhelming number of applications and the wide variety of equipment (hardware) and programs (software) marketed for use in the classroom. The teacher is, after all, an educator, not a computer scientist.

Fortunately, it is not necessary to be a computer scientist in order to make wise and cost-effective decisions about what types of software to purchase. With a limited understanding of the capabilities of the computer to teach and a few basic facts about the kinds of software on the market, teachers can optimize their use of classroom computers.

Three types of CAI software

Before purchasing software, the teacher needs to know the three basic types of software on the market. These are drill and practice, tutorial, and simulation.

By far the most common type of software on the market is drill and practice. This software presents repetitive exercises that reinforce and test concepts that have usually been presented by the teacher in a lecture. For example, a mathematics teacher may use a drill and practice program that generates numerous multiplication problems in order to determine whether students have mastered the skill of multiplication. Drill-and-practice programs are relatively inexpensive to develop, which accounts for the large number of such programs on the market. However, many educators feel that drill and practice software does not use computer capability to its fullest potential. A major problem with such software is that it often cannot hold the interest of students for more than one usage.

Another type, less common than drill and practice, is tutorial software. Tutorial software aims at presenting concepts to students. In addition to series of questions much like those presented in drill and practice software, tutorial software includes passages of text and graphic representations that introduce and clarify concepts. Still, many educators feel that such software limits the potential for interaction between student and computer.

A third type, both the most promising and the rarest on the market, is simulation. Simulation software makes use of capabilities of the computer that fascinate students and make learning by computer adventurous: graphics and sound. This software creates conditions similar to real conditions that the student does not normally experience, or recreates conditions from the past in the present. For example, airplane pilots now receive extensive training by simulation before risking the dangers of actual flight. Also, simulation software helps history teachers recreate the conditions of historical events that occurred many years ago, thus bringing those remote events to life for their students. Simulation software is the most complicated type of educational software available, and hence

the most expensive and least available. It is likely, however, that publishers of educational software will respond to teachers' requests for more software of this type, particularly since simulation software makes optimal use of computer capability.

Surveying the market

Once teachers have decided what types of educational software are most appropriate for the subjects they wish to teach - and there is software available for virtually every subject conceivably taught in the traditional classroom - they are ready to begin shopping for programs. To reduce the perplexity of this task, several sources of software review information are available to the teacher.

Of course, teachers can approach software publishers directly for information about products on the market as well as about products under development. This requires some caution on the part of the teacher, for the objective of the publisher is to sell the product. Therefore the information provided by the publisher is likely to be biased to some extent. Or a product may be so new that the publisher may be unaware of its deficiencies.

A prime source of software-review information is a professional journal. With the advent of computers in the classrooms, a number of periodicals devoted entirely to software review have appeared on library shelves. In addition, many educational and computing journals regularly feature articles that review new products on the market. Teachers may recommend that school libraries subscribe to such periodicals or may choose to subscribe for their own purposes.

Another excellent source of software-review information lies in colleagues who are likely to be using similar programs. Teachers who have implemented given programs can be helpful in explaining those programs' strengths and weaknesses. They are likely to have a unique sensitivity to the concerns of the teacher looking for software, a sensitivity that computer programmers and software publishers may lack. For this reason, many school systems interested in educational computing have designed standard software evaluation forms that are shared by many teachers, thus reducing redundant time-consuming evaluation chores. Teachers who do not have such a form available can easily design evaluation forms for their own use. Software evaluation forms provide an objective basis for purchasing decisions.

Items for a software evaluation form

Although the concerns and goals of teachers will vary, there are several general characteristics of software that should be taken into consideration before purchasing programs for

classroom use. Of primary importance is the physical equipment required to run a given program. Thus, teachers must be familiar with the hardware that is available to them, or must consider the budget available for purchasing necessary hardware. Software designed for one computer system may not be compatible with systems from other manufacturers, a factor that may limit a teacher's choices severely. It is also important to consider the type of storage medium required for a particular software package: tape cassette, disk, or cartridge.

Next, the teacher must take into account what instructional techniques underlie a given program. These may be drill and practice, tutorial, simulation, problem-solving, games, or other techniques, that may be more or less effective for a particular educational setting.

After these general characteristics have been noted, more specific qualities of the software under consideration must be examined. Does the program operate smoothly, or does it contain errors ("bugs") that may prove frustrating to students? Are screen displays attractive and legible? Does the program make effective use of the computer on which it will be run - in other words, does it present colourful displays and interesting audio effects?

Yet another factor to consider is the degree of student involvement that the program encourages. Drill-and-practice programs are criticized for discouraging student involvement, while simulation software is heralded as the most efficient way of bringing students and computers together into an innovative learning experience. It is important to note what sorts of feedback a program provides. Its positive reinforcement must be more interesting than its negative feedback; more than one teacher has been disheartened to find that students intentionally enter erroneous responses because the computer's response to incorrect answers is more entertaining than its positive reinforcement. On the other hand, the negative response should not be demeaning to the student, producing embarrassment or insecurity.

Other aspects of the software to be examined have to do with its overall usefulness. Teachers should try to determine whether a program is sufficiently interesting to encourage its repeated use by students. A program used only once or twice is obviously less effective and cost-efficient than a program to which students will return voluntarily time after time. Good programs provide summaries of performance for the benefit of both teachers and students. This allows teachers to prescribe remediation, when necessary, and to reward progress.

Software created for educational settings usually includes other instructional materials in addition to the programmed instructions for the computer. Taken together, these materials are referred to as "courseware" and most often include teacher's manuals, documentation explaining the way the software operates, student workbooks, and additional learning materials. Each of these materials must be examined individually to

determine its effectiveness and applicability. Good courseware includes clear and thorough documentation, and practical workbooks.

Finally, teachers must ask themselves the larger questions about how appropriate a particular program is to their planned use of it. Is the software under consideration educationally sound? Does it accomplish the objectives for which it is designed? Is it applicable to a given educational setting?

Evaluation forms must include some method of ranking responses to these and other relevant questions. Some may choose a range of responses such as "excellent, good, fair, poor" while others choose to rate characteristics on a numeric basis (e.g., giving each program reviewed a score based on a scale of 1 to 10). Numeric rating has the advantage of providing a clearcut and objective basis for the selection of software.

To repeat, it is helpful and convenient for teachers to share their software reviews with other teachers whenever possible. This decreases the time that must be spent on software evaluation and increases the chances of wise purchasing decisions.

Designing your own software

Dissatisfied with the educational software currently on the market, increasing numbers of teachers are choosing to design custom software for their classrooms. The primary impediment to teacher-designed software is the expertise required to perform extensive programming. Many teachers are enrolling in classes to learn programming languages like Basic that enable them to write software, but this is a slow process. Another problem is the great amount of time required to create innovative educational software.

Several options exist. In some school systems, student programmers are hired to produce software that meets the specific needs of teachers who commission the programs. This not only frees teachers to go about the business of teaching, but it also provides valuable experience to budding computer programmers - and at a relatively low cost to the schools that benefit from developing custom software.(Watt, 1982)

Another exciting possibility for teachers is the advent of authoring languages. Authoring languages, the most common of which is Pilot, are simple programming languages that lead non-expert programmers through the process of writing their own programs. Little time is required to learn to use these languages, so teachers spend most of their time designing the program, and less worrying about the ins and outs of a complex programming language.

The future of educational software

While some educators complain about the dismal lack of effective educational software on the market today, most agree that the future holds promise of better programs for classroom use. It must be noted that the shortage of software has been caused in part by the phenomenal and speedy development of affordable, powerful microcomputers, especially during the last decade. The shortage of educational software, then, is probably quite temporary.

As the market grows, encouraging publishers to devote larger research and development budgets to its creation, teachers will certainly find their range of choices wider and more attractive. In fact, teachers today have the opportunity to participate in the development of good educational software by keeping publishers informed of their needs and expectations. Most software publishers are receptive to input from teachers.

In the future, drill and practice software will most probably lose some of its dominance of the educational software market. Tutorial and simulation software, programs that transform the computer from machine into instructional tool, will be more widely available. In the years ahead, teachers will be able to select from a wide range of software designed to optimize their classroom computers.

REFERENCES

Bitter, Gary G. and Ruth Camuse, Using the Microcomputer in the Classroom, Reston Publishing, Reston, VA 1984.

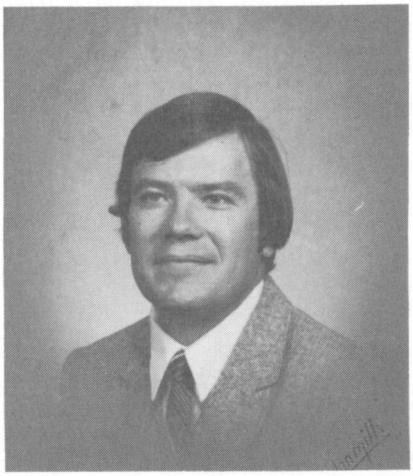
Bitter, Gary G. Computers in Today's World, John Wiley and Sons, New York, NY 1984.

Frenzel, Louis E. "Learning with Micros," in Interface Age, February 1981, p.26.

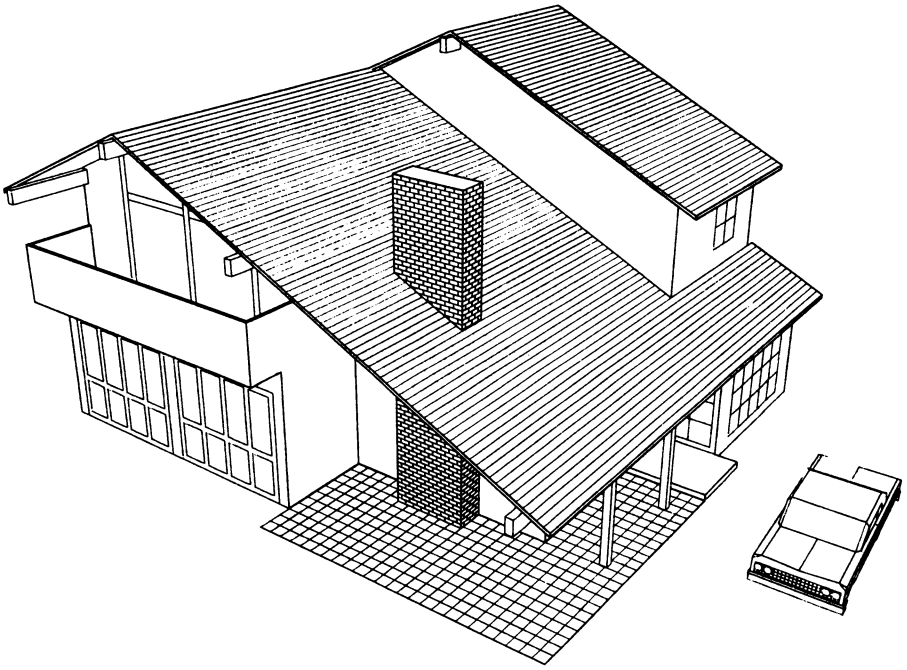
Kleiman, Glenn and Mary Humphrey, "Writing Your Own Software: Authoring Tools Make It Easy," in Electronic Learning, Vol. 1, No. 15, pp. 37-40.

Michael, Frederick W. "Educational Computing at the Crossroads," in Interface Age, October 1981, pp. 68-70, 158.

Watt, Molly, "Making A Case for Software Evaluation," in The Computing Teacher, Vol. 9, No. 9, May 1982, pp. 20-22.



Gary G. Bitter teaches computer literacy courses (including one on television), edits the School Science and Mathematics Journal, and directs the Microcomputer Research Clinic at Arizona State University, Tempe, Arizona, where he is Professor of Computer Education. He is the author of ten books on the subject.



Demonstration drawing produced by a Calcamp 960 Plotter,
to show speed and versatility, 1981.