Some observations on applications of microcomputers in teaching

THOMAS T. HEWETT

Drexel University, Philadelphia, Pennsylvania

Several useful and important observations appear in the papers in this symposium. This paper highlights some of those ideas and some of the underlying themes of the symposium papers. Finally, focusing on some of the problems of either selecting software or developing software, the paper suggests that the fundamental problems associated with developing a computer-based instructional laboratory are the same as those associated with developing any instructional laboratory: setting one's instructional goals and then integrating the equipment and procedures into the instructional context.

The papers in this symposium illustrate several points that should be emphasized. One of the most important is a basic theme of the paper by Butler (1988). In addition to providing some useful criteria for the selection of instructional software, Butler stressed the importance of identifying and selecting software that can be a useful tool for accomplishing a piece of work. It is in its role as tool that the personal computer seems to offer the greatest potential for instructional computing (see Hewett, 1986, 1987).

There is an equally important theme running through the observations made by Eamon (1988). In recent years increased attention has been paid to understanding the factors that make a piece of software both usable and useful. However, Eamon has noted something that can be overlooked when one is grappling with the technical difficulties associated with improving the usability of a program. A continuing focus on the content and objectives of the software is absolutely vital to the development of a successful piece of instructional software. Just as a glossy, empty package is seldom preferable to an ugly old container full of money, a good-quality interface design may remove barriers to the usability of a program, but does not guarantee that it will be a useful program.

One interesting trend suggested by the case study papers in this symposium (Conn, Stafiniak, DiPasquale, & Harper, 1988; Hornby & Anderson, 1988; Leith, Hamm, Duncanson, & Senical, 1988) is that as the costs of obtaining and maintaining traditional laboratory equipment go up, the computer and its software begin to look increasingly more attractive as an alternative for laboratory course development. The papers in this symposium demonstrate that the substitution of the personal computer and its software for more traditional forms of laboratory equipment can succeed; however, regardless of whether one adopts the approach of selecting from previously devel-

oped software or developing one's own, there is no simple way to go.

Although selecting software allows one to avoid the difficult and inherently time-consuming process of programming, the difficulties of selecting software revolve around the work required to find and identify software that works and that fits with one's instructional content and goals. Simply getting a chance to look at some software can be time-consuming and frustratingly difficult. Often what one sees when the software is finally available is not worth the effort expended to locate and obtain it. Even when you have found usable software, you may not want to use what you have found. The software may not fit your style, the content of the course, and so forth.

Although developing one's own software may ensure that the program works and is compatible with one's instructional goals, it often takes a considerable amount of time and work, and one may have to risk becoming a programmer instead of a psychologist. Furthermore, developing one's own software also contains traps for the unwary. It is easy to become enamored with a program over which one has labored intensely, and to lose sight of the fact that the program does something that is basically trivial or not particularly useful. (For some time now, it has been my suspicion that this trap is one reason there is so much badly written courseware. Another reason seems to be that most programmers do not know much about psychology.) In addition, since many programs are similar to icebergs, with most of the work hidden from the casual observer, it can be particularly frustrating to spend hours laboring over a program that the average user may complete in 10 to 15 min. Thus, the "courseware architect" (Hewett, 1986) who also does his/her own programming can easily feel that his/her work is undervalued and misunderstood. Sometimes it is; often it is not.

All of this leads to a warning about the "vaporware" problem. To paraphrase, "Beware of publishers (or colleagues) bearing disks!" It is not yet possible to obtain a turnkey collection of laboratory programs, and one should not expect or want to. Not only is the state of the

The author's address is Department of Psychology and Sociology, Drexel University, Philadelphia, PA 19104.

art not good enough yet, it is not clear that it will be in the near future. Some programs intended for instructional use do little more than turn workbook pages with less efficiency than is possible when using a real workbook. Other programs that might be worth using are similar to the demonstrations in many instructor's manuals. Without a considerable background and context for the courseware, the result of its use will be lots of blank stares or looks of confused panic. (In some cases, there is so little context that even the instructor may not get the point.) The use of any tool in an instructional laboratory, regardless of whether it is computer based or not, requires careful planning and integration into the instructional context.

Despite the differences in approach taken by the developers of the laboratories described in these symposium papers, there is a common underlying theme. Regardless of whether one takes the path of selection or development, all of the papers in this symposium illustrate the importance of having software that fits or works in one's instructional context. In and of itself, finding or writing software is actually not a particularly difficult problem when compared with the problem of getting a piece of pedagogically useful software, regardless of whether one writes it or buys it. A major portion of the work in mounting a successful, computer-based laboratory course takes place *after* finding or writing software that will fit. Regardless of whether one writes or buys software, it will probably be necessary to either develop or substantially revise the supporting program documentation. Then it is time to write the handouts and other instructional materials for student use that are needed to tie the software into the instructional context in which it will be used.

For example, in a recent paper (Hewett, 1985), I described some ways an electronic spreadsheet can be used in teaching students to model neural networks. After learning about the capabilities of an electronic spreadsheet, it took me less than 45 min to set up each of the simple models now in use with my cognitive psychology class. Then the real work began. It took days to prepare the supporting documents required to enable students to understand what the programs were about and why the topic of neural modeling was important. In class, it generally takes the best part of a 1-h lecture-demonstration period, with the spreadsheet models actually running on a computer with a large-screen monitor, to explain to the students what they should be looking for in the models. Finally, it is usually necessary to spend half an hour or so

later on in helping the students to understand what they saw and did when they were working with the models. As with many other pieces of instructional software, the availability of the spreadsheet neural models has not changed the fact that some students have trouble understanding the concepts. However, as with some other instructional software, the models do provide another tool that helps some students work through the concepts.

The major problem in developing a computer-based instructional laboratory experience is still much as it has always been in developing any instructional laboratory experience: fitting the equipment and procedures into the instructional context. In fact, one important point illustrated by all the papers in this symposium is that the least important and least difficult part of creating a computer-based instructional laboratory is identifying the computer hardware. Of more importance and greater difficulty is the creation or identification of potentially useful software. Finally, the most important and most difficult part of the process is setting one's instructional goals and integrating the computer-based laboratory into the instructional context of the course.

REFERENCES

- BUTLER, D. (1988). Selection of software in the instructional laboratory. *Behavior Research Methods, Instruments, & Computers*, **20**, 175-177.
- CONN, G., STAFINIAC, P., DIPASQUALE, M. C., & HARPER, L. (1988). Effects of teaching an introductory psychology laboratory using a computerized research tool. *Behavior Research Methods, Instruments, & Computers*, **20**, 184-187.
- EAMON, D. B. (1988). Problems with evaluation of courseware for instruction in psychology. *Behavior Research Methods, Instruments, & Computers*, **20**, 178-179.
- HEWETT, T. T. (1985). Teaching students to model neural circuits and neural networks using an electronic spreadsheet simulator. *Behavior Research Methods, Instruments, & Computers*, **17**, 339-344.
- HEWETT, T. T. (1986). When every student has a computer: A new perspective on courseware and its development. *Behavior Research Methods, Instruments, & Computers*, **18**, 188-195.
- HEWETT, T. T. (1987). The computer as a tool in instructional computing: Students as software architects. *Behavior Research Methods, Instruments, & Computers*, **19**, 123-127.
- HORNBY, P., & ANDERSON, M. (1988). Using computers in introductory psychology. *Behavior Research Methods, Instruments, & Computers*, **20**, 180-183.
- LEITH, C. R., HAMM, H. D., DUNCANSON, C., & SENICAL, T. (1988). A computer-driven undergraduate laboratory curriculum and related support logistics. *Behavior Research Methods, Instruments, & Computers*, **20**, 188-190.