Proceedings of the 1983
Winter Simulation Conference
S. Roberts, J. Banks, B. Schmeiser (eds.)

AN APPROACH TO TEACHING SIMULATION TO UNDERGRADUATES

Maurice F. Aburdene
Dept. of Electrical Engineering
Bucknell University
Lewisburg, PA 17837

David Finkel
Dept. of Mathematics
Bucknell University
Lewisburg, PA 17837

Although traditionally a subject for graduate study, simulation has recently been included in many undergraduate curricula. After a discussion of the objectives of a simulation course for undergraduates, this paper recommends an approach to teaching such a course which makes use of a commercially available simulation language, such as GPSS or SLAM. This approach is discussed in relation to the course objectives. Finally, the results of the authors' experience with this approach are discussed.

1. OBJECTIVES OF AN UNDERGRADUATE SIMULATION COURSE

Simulation is a formal name for a natural way of thinking, a way of using analogy to unravel the workings of a complex system. It is the process of developing a simplified model of a complex system, and using the model to analyze and predict the behavior of the original system.

We see three major objectives for an undergraduate course in simulation. The first is to instill in the student the ability to identify the important relationships or laws governing the behavior of a social, engineering, economic, or behavioral system and to understand the appropriate level of approximation. This is the construction of a mathematical or logical model. The second objective is to familiarize the student with the methods of converting the mathematical or logical model into a computer model; that is, implementing a simulation model. The third objective is to enable the student to evaluate, understand, and draw appropriate conclusions from the results of the computer model. These objectives relate to three phases of a simulation study: model construction, model implementation, and the model application and interpretation.

We explicitly exclude as course objectives topics we feel are more appropriate to advanced, graduate level simulation courses. These topics include design of simulation languages, efficiency of simulation programs, and detailed comparison of simulation language characteristics.

2. APPROACHES TO TEACHING AN UNDERGRADUATE SIMULATION COURSE

Many ways of choosing a language for teaching simulation courses have been suggested (Roberts 1982). These can be classified into three main categories: using a general purpose high-level language such as Fortran or PL/I; using a simulation language or library package specially designed for teaching; and using a commercially available simulation language, such as GPSS or SLAM. Superficially, the choice of a language appears to relate only to the second course objective, implementing the model on the computer. In fact, the choice of a language will affect all three objectives. The implications of these three approaches will now be discussed in more detail.

2.1 General Purpose High-Level Language

The use of a general purpose high-level language is probably the most popular approach. The text by Payne (1982) makes use of this method. There are a number of advantages to this approach. It is inexpensive since the academic institutions usually have the high-level language implemented on their computer, and so no special software purchase is required. Also, it is a natural outgrowth of an introductory programming course, and can serve as an advanced programming course as well. Finally, this approach provides the student with an understanding of the fine details of the workings of a simulation program. We note however that we do not consider this to be a primary goal for an undergraduate simulation course.

The major drawback of this approach is the amount of time and effort the student must spend developing computer code. The student views the course objective as getting the program to work, and loses sight of the other course objectives of constructing valid models, and drawing appropriate conclusions from the simulation runs. Because so much time must be spent coding each assignment, fewer assignments can be made, limiting the student's exposure to a variety of modeling techniques. Finally, this approach discourages students with indifferent programming backgrounds from enrolling in the course.

CH1953-9/83/0000-0375 $01.00

2.2 Specially Designed Simulation Language or Library Package

The second approach to teaching simulation provides the student with a limited capability simulation language, or simulation library package, especially designed for teaching simulation. Texts using this approach include Kochenburger (1972), and Law and Kelton (1982).

Like the high-level language approach, this approach is relatively inexpensive, since the necessary software is provided by the textbook authors, or developed by the instructor. When a language is provided with a textbook, there is the advantage of good coordination with the other course teaching materials. This approach also reduces the amount of computer code the student must create, allowing them to concentrate on the other phases of a simulation project.

The major disadvantage of this approach is that these specially designed simulation languages have limited capabilities. They often do not include advanced modelling features and output capabilities included in commercially available simulation languages. Furthermore, some are not documented or supported as well as the commercial languages. A third disadvantage is that the student does not learn a language that will be available to them in an industrial setting.

2.3 Commercially available simulation languages

The approach recommended by the authors is the use of a commercially available simulation language for instruction. The most popular simulation languages are GPSS, SIMSCRIPT, SLAM, DYNAMO, ACLS, and CSMP. The first three of these are designed for the simulation of discrete systems, and scheduling and queueing problems. The remaining three are designed for the simulation of continuous systems, and complex feedback systems.

The use of such simulation languages answers most of the disadvantages identified for the other two approaches. These languages are designed to be relatively easy to use. At least a subset of the language can be used by students with minimal programming backgrounds, and hence a diverse group of students can be served by the course. These languages also allow the efficient development of complex models, with only a modest amount of computer code required, reducing the burden on the student.

These languages provide a general modeling technique for designing simulation models, such as the blocks in GPSS or the network nodes in SLAM. Furthermore, these languages are well-documented and maintained by professional concerns, and they offer a wide variety of utility functions, and output reporting options, to encourage the student to make proper use of the simulation results. Finally, the student learns how to use a completely transportable tool which they will be able to use in the workplace.

Some of the objections to using such languages which might have been raised a few years ago are no longer valid. For example, these languages were once expensive to purchase and required extensive computer facilities to run. Now, most of the major languages are available free or at nominal cost to academic institutions. Language developers have recognized the advantage to them of having students trained in their languages. Furthermore, the languages no longer require the extensive computer resources they once did. One powerful new language, SIMAN, can even be run on a personal computer (Pegden 1982).

Another objection to using simulation languages for instruction was a shortage of good teaching materials and resources. Now, however, there are textbooks based on several of these languages, for example Gordon (1975), Pritsker and Pedgen (1979), and Russell (1983). Some of these books in fact provide an excellent introduction to simulation in general, and can serve as the primary text for a simulation course. In addition, several organizations offer short courses in these languages, which may be taken at reduced cost by faculty members. Thus the prospective teacher of one of these languages will find ample support.

3. RELATIONSHIP TO COURSE OBJECTIVES

Using commercially available simulation languages makes it easier for the instructor to meet the objectives stated earlier for an undergraduate simulation course. For example, the block diagram approach of GPSS, SLAM, SIMAN, CSMP and DYNAMO provide a convenient framework for conceiving of and developing a model and converting the model into code. In fact, with SIMAN the student can develop a block diagram graphically using an interactive terminal, and have the diagram automatically converted into computer code.

With the commercial simulation language approach, the student spends much less time generating computer code (see Gayeski 1983), leaving more time to understand the simulation process, and concentrate on the objectives of the simulation. The student is also able to perform sensitivity analyses, try alternative strategies and alternate system designs. The student is also able to develop and run simulation programs much earlier in the semester, and thus gain wider exposure to simulation modeling techniques.

Most of the commercial simulation languages have excellent output facilities, providing automatic statistics collection, tabular summaries, and in some cases graphic output. These capabilities make it easier for the student to validate the model against actual data, and draw appropriate conclusions about the behavior of the system under study.

4. EXPERIENCE

We have used SLAM, GPSS, DYNAMO, and CSMP in our simulation courses.

The use of a commercial simulation language in the teaching of an undergraduate simulation course opens the course to a wide variety of students, and deemphasizes the computer programming aspects of the course, and the need for the student to be an expert programmer. This provides the oppor-

tunity to attract students from diverse disciplines, and to model and simulate complex, large scale systems of an interdisciplinary nature.

Students were able to execute a simulation program from the first day of class and were able to understand the model and evaluate the results. As the course progressed, more complex and realistic systems were addressed. Finally, students were able to do a large scale system simulation term project by the end of the semester.

After the students became familiar with the simulation language, it was easier to discuss the various features of the construction of simulation programs, such as sorting, event scheduling, integration, and random number generation. It was also easier to discuss statistical aspects of output analysis when the tools to perform the statistical analysis are immediately at hand, as features of the simulation language.

The students were enthusiastic about the courses, and were able to concentrate on the important concepts of simulation rather than the routine of programming. The instructors were gratified by indications that the students came away from the course with a good understanding of fundamental ideas appropriate to an undergraduate introduction to simulation.

REFERENCES

Gayeski, P. (1983), Towards better computer science education, Comm. A.C.M., Vol. 26, p. 464.

Gordon, G. (1975), The application of GPSS V to discrete systems simulation, Prentice-Hall, Englewood Cliffs, N.J.

Kochenburger, R. (1972), Computer simulation of dynamic systems, Prentice-Hall, Englewood Cliffs, N.J.

Law, A.M., Kelton, W.D. (1982), Simulation modeling and analysis, McGraw Hill Book Co., New York.

Payne, J.A. (1982), Introduction to simulation, McGraw Hill Book Co., New York.

Pegden, C.D. (1982), Introduction to SIMAN, Systems Modeling Corporation, State College, Pa.

Pritsker, A.A.B., Pegden, C.D. (1979), Introduction to simulation and SLAM, Halstead Press, New York.

Roberts, S.D. (1982), Teaching simulation to undergraduates, Proc. of the 1982 Winter Simulation Conference, IEEE, New York, N.Y.

Russell, E.C. (1983), Building simulation models with SIMSCRIPT II.5, CACI, Inc., Los Angeles.