Simulation and Gaming in IT Education

Norman Pendegraft
University of Idaho, USA

INTRODUCTION

The rapid change in information technology presents several serious problems to IS educators. In particular, the number of basic ideas that must be mastered is constantly increasing while the time available is not. This makes it essential to use class time efficiently as well as effectively. Simulations and simulation games provide an interesting and useful tool to help in this effort.

BACKGROUND

Simulation, case studies, role playing, and gaming are related teaching methods based on experiential learning. They permit experience or experimentation with a situation modeling the real world (Senge, 1990). On a deeper level, simulation is claimed by some to be a fundamentally new way of studying the world (Pagels, 1998). The idea that students learn better by doing goes back at least to Dewey (1938). The key idea underscoring this approach is that people learn better from experience than from reading or listening (Corbeil, Laveault & Saint-Germain, 1989). By compressing time, the simulation allows the students to experience the consequences of their own actions or to see how a system operates.

Simulation as a teaching tool suggests several approaches. Perhaps most obvious in an information systems curriculum is computer simulation, a tool well known in the information systems community. Using this technique, a computer program is written which exhibits behavior that models the behavior of the system under study. Butterfield and Pendegraft (1998) described a spreadsheet simulation of a Fourier Series, adding sine wave to construct a square wave, thus demonstrating how bandwidth limits data rates. Simulation can be easily used in the information systems classroom by having students operate or create simulations of relevant technology. Campbell (1996) created a simulation of a computer and had his students write assembly language programs to execute on the simulation. Englander (2003) uses Little Man Computer, a simple paper simulation of a CPU, as an example to explain basic CPU architecture, CPU operation, and machine language. In an extension of that idea, Pendegraft and Stone (2003) had their students develop a Visual Basic simulation of a Little Man central processing unit on which they ran programs mandated by the instructor. In addition to having to execute simple programs written in Little Man's machine language, their simulation had to deal with other architectural issues like input and output.

Case studies are a time-honored approach of instruction in strategy courses (see for example, Burgelman, Maidique & Wheelwright, 2001). Barker (2002) suggests that they can also be very valuable for teaching technical skills such as software development. In some sense, a case study is a role play, with the student acting the part of an analyst examining the case situation.

Role playing and simulation gaming are similar approaches in that they use simulated worlds, but instead of creating or observing or analyzing that world, students are immersed in it. Role playing is a method in which students are presented a scenario simulating some real situation, and assigned roles in that scenario. The scenario can be based on real or simulated situations (Barker, 2003). Participants then assume the roles of relevant persons in the scenario and act out the situation to see what happens. Role playing is a commonly and successfully used tool in IS education (for example, Christozov, 2003).

According to Greenblat (1988), simulation gaming includes role playing as an element, Whereas role playing allows participants to play the roles as they please, simulation gaming emphasizes the interactions of the roles and constraints of various types on the players. In some sense, a simulation game strives to teach about a specific situation, while a role play or game may have a more general lesson. The additional structure allows us to focus the students' attention on key issues. Dennis (2002) and Pendegraft (2002, 2003) both developed classroom gaming exercises to help students understand TCP/IP from the inside. Their exercises will be examined in more detail later in this article.

After many years of using such exercises at all levels (undergraduate, graduate, and executive), it is the author's opinion that they are very useful and that major benefit accrues to the instructor in preparing the simulation as well as to the students when they play the game. Simulation and gaming are student-centered learning, that is, the student is actively involved in the learning rather than passively observing the instructor (Greenblat, 1998). The student does the work, makes decisions, and sees the impact of the decisions. Role playing and simulation gaming attempt to take advantage of this by creating a situation in which a student may "play a game" in which time is compressed and attention can be focused on a few key ideas. Finally, these kinds of exercises are fun. The class gets to move around, talk, and frequently laugh. Simulations and games epitomize the idea that learning should be fun.

USING A GAME TO TEACH TCP/IP

TCP/IP is a layered protocol for controlling data flow in a packet switched network. For a more complete discussion of TCP/IP see, for example, Hunt (1998). Figure 1 lists the layers. TCP/IP uses packet switching, which means that each message is broken into pieces (called packets), each of which contains part of the message. Each packet is augmented with a header that contains (among other things) the addresses of the source and destination machines. At each step along the way, a router looks at the header, determines the address of the next host along the way, and forwards the packet to that next host.

Understanding the addressing presents a major hurdle to understanding TCP/IP. There are two types of addresses in each packet. The IP header includes the IP addresses of the source and destination machines. These do not change as the packet moves through the network. DLL layer header includes the MAC addresses of the source and destination machines for the current hop. These change at each hop. Since there are two destination addresses and two source addresses, some students have difficulty understanding how the addressing works. One way to help explain the addresses is to have the students play a game.

As examples of this approach, consider two similar games, one designed by Dennis (2002) and one by Pendegraft (2002, 2003), to help teach how TCP/IP works. Both are published elsewhere and so will not be described in detail here. Both games are designed to be run in one class session in a course on telecommunications. Both seem suitable for undergraduates or graduate students.

Figure 1. TCP/IP layers

Layer		Address
Application		
Transport Control	TCP	Port
Internet Protocol	IP	IP
Data Link Layer	DLL	MAC
Physical Layer		
TCP/IP Layers		

Both could be adapted to class sizes ranging from a dozen to more than 40.

The games have similar structure. The class is divided into teams, each team representing a host. Each player represents one layer on that host. A network map and instructions for each layer are given to the players. Figure 2 shows a sample network, and Figure 3 gives an excerpt from Dennis's instructions for the Data Link Layer.

In the play of the simulation, an application layer player writes a message to another application layer player on a paper form. The form is then handed to the TCP player of the sending host. The TCP player adds the TCP header and hands the packet to the IP player. The IP player adds the IP header and hands the packet to the DLL player. The message is then passed to another host where each player strips off the header for that layer and hands the message upward, or in the case of IP forwards it as necessary.

There are several structural differences. Pendegraft gives the IP layer a routing table (see Figure 4 for an example), while Dennis has the IP layer use the network map to determine first hop IP addresses. Pendegraft uses a separate envelope for each layer and its header, while Dennis has different forms for each layer's header data which are taped to the message form. Another difference is that in Pendegraft's game, there is no physical layer player. Instead, the DLL players hold the ends of pieces of wire representing the physical layer. When a DLL player finishes adding the MAC addresses to the packet, he strings the packet on the wire and by raising his hand, sends the packet to the next machine. Such silliness introduces some humor into an otherwise dry subject, and helps keep the students interested and involved.

The games offer different points of view. Dennis's game allows many messages on the network at one time; consequently, it offers a more complete (and hence more complex) model of TCP/IP. In Pendegraft's game only one message is sent at a time and the entire class follows it along the way, immediately discussing problems that may occur such as a player incorrectly addressing a packet. In Dennis's game that sort of error is handled in discussion between the affected players. One result of these two points of view is that in Dennis's game, each player has a different experience that can be shared in the post-game discussion, while in Pendegraft's game, there is a more shared experience.

Both games simplify TCP/IP, ignoring some issues like handshaking or error detection. This is not to say that these issues are unimportant, but that these games focus attention on a limited set of issues of paramount importance, primarily addressing. Pendegraft's game is designed to allow including such complexities "notionally." That is, the instructor can intervene with an external complexity. For example, after a couple of messages have

Related Content