



## **Simulations Primer**

By Dave Feasey

Friday, August 2, 2002



### e-Learning by Doing

Bill Horton recently observed that “e-Learning changes nothing about the way people actually learn.” Nearly everyone in our industry recommends e-Learning by doing rather than e-Learning by reading. Yet, with notable and expensive exceptions, e-Reading continues to dominate the category. Up to 70% of e-Learners are dropping out as a result.

E-Learning has been fairly easy to cost-justify, compared with classroom facilities, trainers, lost opportunity cost, travel and hotel expenses, especially if the only thing involved is dumping textbooks out to web pages. Despite the real savings, customers aren't seeing the return in performance that e-Learning promised. The reality now is that people actually need to learn in order for e-Learning to be a sustainable concept.

Enter the mighty simulation. According to Brandon Hall, simulations represent the growth area in the e-Learning market. Learning simulations promise rich interactive experiences with better feedback and higher retention. Simulations rank second only to one-on-one mentoring in terms of effectiveness. Simulations are also highly valuable in training scenarios where mistakes in the real world would be dangerous or costly. We learn by failing, as Roger Schank is fond of reminding us. Yet crashing a web server or jetliner as a training exercise is obviously too costly, no matter how instructive the learning experience.

Simulations address a huge range of scientific, military, municipal, industrial and business problems. There is a large literature base around simulation design. Peter Senge's *Fifth Discipline*, for example, uses the systems dynamics model as a tool for creating learning organizations. The scale of simulation types ranges from complex

weather and macroeconomic systems modeling down to the humble spreadsheet. Training buyers will want guidance in determining simulation capabilities, and suitability of tools for meeting their needs.

Many will be familiar with software application simulations, although these don't particularly take advantage of some key benefits of simulation. Schank bluntly calls them a waste of time. Better to build performance aids and job support into the applications themselves. So we won't cover software sims in this article.

Instead, we'll focus here on the use of simulations to model and “run” systems of varying complexity, in order to understand the relationships embodied in these real world systems.

### What's in a Sim?

The generally accepted definition of a simulation is a “representation of the dynamic behavior of the system by moving it from state to state in accordance with well-defined operating rules.” [Pritsker] The words system, model, and simulation are often used interchangeably, but we should be clear about these terms. A system is a real world set of entities that interact in some way. A model is a symbolic representation of part of the system. Models are generally represented as entities or components making up the system and the relationships between these entities. Models are thus limited abstractions of systems. A simulation is simply a model in action.

So simulations are representations of the dynamic behavior of systems. Systems theory and cybernetics, and to some extent communication theory, are thus foundational to our understanding of simulations. Systems are typically non-linear, and exhibit “emergent” behaviors that could



## Simulations Primer

not be predicted from the sum of their parts. Non-linear behavior is difficult for humans to grasp without the direct experience of many interactions with the system or simulation. Another key concept is that systems often generate feedback loops that can lead either to self-regulation or to infinite decrease or increase— “vicious” and “virtuous” cycles. Systems are typically part of larger systems and are composed of smaller subsystems. In other words, they form hierarchies.

Since the simulation represents the model in action, there is a time element that is central to all simulation. One of the biggest benefits of simulation is the ability to play with time, speed it up or slow it down or run it backwards or skip ahead or play back from a certain point in time. The way that a simulation handles time and events is one way of categorizing simulation types.

There are three main temporal models of simulation: Systems Dynamics, Discrete Event and Object-Oriented. The Systems Dynamics model represents a system as a set of “stocks” and “flows.” Stocks represent various measurements, volumes or attribute values, and flows represent the relationships of increase or decrease between stocks. Think of a drainage system in which a series of pipes transfer water from one pond to another. Depending on the flow into and out of a stock pond, a pond may go dry or overflow or reach an equilibrium state. Systems Dynamics models are useful for both for concrete and abstract modeling. They are used for modeling animal populations, business strategy and even character motivation in Shakespeare. Time flow in a Systems Dynamics model is continuous, calculating new values for each stock based on it’s input and output flows, all at once. Systems Dynamics models do a good job of modeling overall function in a

system, but aren’t totally accurate since they don’t usually handle variable flow or the priority or frequency of inputs. However, Systems Dynamics models are easy to use and flexible enough to modify the structure on the fly. They generally give a good representation of the overall structure and behavior of the system. PowerSim is the leading authoring tool, though iThink is adequate for most applications.

The Discrete Event approach models every event in the system and creates an “event queue” to process each event as it comes to each station in a system. Discrete Event models do well at looking at the detailed functioning of a process-oriented system. They are often used for tracking utilization patterns in manufacturing, contact centers and other processes where bottlenecks are problematic. While yielding more accurate time representation in one way, there is a tradeoff. Time between stations in the flow is usually not modeled, and multiple processes are difficult to run concurrently. Interacting with the simulation usually means stopping, making changes and re-running the simulation from the beginning. Simul8 is an authoring tool and scripting language for building Discrete Event sims.

Object-Oriented (OO) simulations can use either continuous or discrete time manipulation, and additionally can manage many processes occurring simultaneously or concurrently. With OO, we can combine the best of both SD and DE models. The OO approach has the distinct advantage that we can interact with objects and the environment in time as the simulation plays. On the down side, OO simulations require a unique set of design and programming skills and are best used in situations where the system is well understood. Almost any programming language, including Flash and



Director can be used to produce OO simulations.

The Object-Oriented (OO) model includes two interesting subtypes: agent-based and SWARM models. An agent is an independent decision-making program that interacts with a user, other agents and the environment to negotiate, create and retrieve information. SWARM sims take inspiration from the seeming group intelligence in insect colonies. They use the emergent complex behavior resulting from the combined simple behaviors of large numbers of independent bots to solve problems such as automated construction in robotics, pattern matching and machine learning.

A final type of simulation that deserves mention is role play simulation. Everyone is familiar with this model. Many board games such as Risk and Dungeons & Dragons are based on the simple premise. In this model, either human learners or computerized functions play roles within the system. Humans can even play abstract roles, such as the bank in Monopoly. By combining our various models, we can envisage a next generation of collaborative role-play sims where digital objects can be assigned to play roles, or forced to yield as new human players join the simulation or switch roles. Many interactive "first person shooter" games such as Unreal Tournament and Quake can now be programmed, offering potential role-play platforms for learning.

## Instructional Design with Simulations

Designing simulation-based learning can be challenging, especially since it reverses some deeply held beliefs and methods of traditional Instructional Design. Most profound of these changes is the move to learner-centricity. The designer is more a

facilitator and resource provider than a director of the learning. It is impossible to control or even predict the learned content, or path through the content.

Ideally, the simulation acts as a sort of front end navigation interface to learning resources that are presented to the learner based on her behavior and the state of the system. The simulation should be designed to give the user a manageable set of relevant learning resources as soon as she experiences some "expectation failure" within the system. These resources can be in the form of traditional e-Learning content, case studies, war stories, animations, online coaches, job aids and so on. What's both powerful and challenging about the simulation approach is that users naturally self-select the material that is relevant to them, even if they don't know what it is ahead of time.

This is not to say that learning objectives cannot be specified and measured. One way to try to control the specific desired outcomes is to reduce the size and complexity of any given simulation. There are drawbacks to this approach, though. Players in good simulations get into flow states, and the interruption of continuously starting and stopping short simulations lessen enthusiasm and deter learning. The simulation should be big enough that you can specify several related objectives that users have a realistic chance of covering, without having to specify the way in which they achieve the objectives. As you can see, we are really talking about a design layer above, and additional to the simulation itself. Consider limiting the simulation to the full set of entities or objects that a hypothetical user might need to access in order to accomplish the objectives. Remember too that we want to create interesting failures, obstacles and challenges for users to stumble over in order to learn.



## Simulations Primer

This is similar to creating good distractors in assessment questions, only much more complex. If a simulation is becoming too complex, try reducing the number of objectives gradually until you achieve the right balance of interest and manageability.

An opposite approach is also possible, one which seeks to build the most comprehensive representation of the system that can be devised. The advantage here is that you only have to design the system once. Even with a very complex system this can be advantageous if you have large numbers of learners at many different levels of expertise. The problem is if the system is incorrectly modeled, or is likely to change the result is an expensive mistake.

The key to designing a very complex simulation is to program the system in such a way that Objectives, Roles and Scenarios can be imported and exported allowing an administrator to “program” the sim to a certain start state (scenario) and a target end state (objective), and a user can save state at any point. If carefully designed, the simulation can even be extended with custom objectives, problems, traps, puzzles, resources, feedback mechanisms and more.

All simulations should offer users a chance to reflect on and discuss their performance with other players. Self-assessment and group “post-mortems” are effective after-events. Like the market in the games world for clues and tips, a simulation can generate its own knowledge market.

Like good games, simulations should be easy to learn, difficult to master. This concept is what makes good games so addictive. This gets into the internal marketing and selling of simulations, which is beyond the scope of the present discussion. In general, if people are having fun with the sim, they’ll be more motivated to spend time there and learn.

In some cases, learning can be built around the design of the simulation, or modeling the system itself. This is very powerful and useful especially in strategy or operations applications where there may be ambiguity or uncertainty about what is causing some behavior in a real system. Senge’s *Fifth Discipline* discusses this approach at some length. The Systems Dynamics model is particularly well suited to this approach since the model itself can be edited, and the simulation run again.

Reflection and De-briefing are vital to simulation-based learning. These can be built into the design of the simulation or as after-events, either as group debriefings or self-assessments. Like the market generated by the games world for clues and tips, a simulation can generate its own knowledge market. Ask users not only for assessment of how they and others performed, but what they liked and disliked about the simulation itself, what was realistic and what wasn’t. Then incorporate their feedback into the next iteration of the simulation design.

## Conclusion

Simulations are the coming wave in e-Learning. By modeling a system and its behavior in an interactive way, many opportunities for learning are created. The instructional design of simulations involves many challenges, including letting go of total control. This article, however, should provide some specific strategies and approaches to achieve learning objectives.