

Simulation Research Methods

Kevin Dooley

Arizona State University

Contact information:

Kevin Dooley

Arizona State University

Departments of Management and Industrial Engineering

PO Box 875906

Tempe AZ 85287-5906

Kevin.Dooley@asu.edu

Phone: 480-965-4612

Web: <http://www.eas.asu.edu/~kdooley>

Cite: Dooley, K. (2002), "Simulation research methods," *Companion to Organizations*,

Joel Baum (ed.), London: Blackwell, p. 829-848.

Simulation Research Methods

Computer simulation is growing in popularity as a methodological approach for organizational researchers. Other research methods must make various assumptions about the exact cause and effect nature of the system under study; for example, in survey research, one must define the form and content of cause and effect a priori in order to learn from the data observed. Simulation allows for researchers to assume the inherent complexity of organizational systems as a given. If other methods answer the questions “What happened, and how, and why?” simulation helps answer the question “What if?” Simulation enables studies of more complex systems because it creates observations by “moving forward” into the future, whereas other research methods attempt to look backwards across history to determine what happened, and how. Because the nature of living systems is to either increase in entropy (disorder) or complexity (order), looking backwards is inherently more difficult than moving forwards.

There are three main schools of simulation practice:

- Discrete event simulation, which involves modeling the organizational system as a set of entities evolving over time according to the availability of resources and the triggering of events.
- System dynamics, which involves identifying the key “state” variables that define the behavior of the system, and then relating those variables to one another through coupled, differential equations.
- Agent-based simulation, which involves agents that attempt to maximize their fitness (utility) functions by interacting with other agents and resources; agent

behavior is determined by embedded schema which are both interpretive and action-oriented in nature.

Simulation researchers typically remain in one camp and are not facile and do not work in all three domains. This will likely change in the future, however, as complex organizational systems really require elements of all three approaches in order to be their complexity to be appropriately captured.

The Purpose of Simulation

Axelrod (1997) outlines seven different purposes of simulation in the social sciences: prediction, performance, training, entertainment, education, proof, and theory discovery. Numerous examples can be found of each of these uses.

Prediction Simulation takes a model, composed of a structure and rules that govern that structure and produces output (observed behavior). By comparing different output obtained via different structures and governing rules, researchers can infer what might happen in the real situation if such interventions were to occur. The validity of such predictions depends on the validity of the model. The vast majority of models used in the operations research and operations management field, for example, fall into this category. Predictions are made concerning how certain changes in (e.g.) inventory control, quality, productivity, material handling, etc. will impact operations (positively). If predictions turn out as expected, then there is more impetus to try such changes in the real system.

Examples include numerous simulation studies done concerning the scheduling of production in flow lines, assembly shops, and job shops (e.g. Law and Kelton, 1982). A

model of the production or service facility, usually of a discrete event nature, is developed whereby the system is defined by events (e.g. a customer order arrives, a machine starts work, a machine finishes work), entities (the jobs that go through the system, and resources (e.g. machines, people, transports). The prediction in question might concern, for example, whether scheduling of jobs at the bottleneck workstation according to “earliest due date” led to better delivery performance than scheduling jobs according to a “first come first served” rule. This hypothesis would be tested over a number of different configurations to determine how robust the answer was; for example, the scheduling by due date might make a significant difference if the number of jobs in the system is large (utilization is high), but might make no difference if the system is not very busy.

Simulation for prediction is a substitute for experimentation and intervention on the actual system. It is undertaken when such experimentation is too dangerous, costly, untimely, or inconvenient. This is often the case with the “human system”, and with the larger-scale elements of the organization (e.g. its information systems, its supply chain, its strategy), because of scalability problems. Changes can be made at a task level of the organization in such a manner that the change can be scaled down appropriately and attempted on a very small component of the organization, with little (relative) cost. This often cannot be done with larger organizational systems, including human systems.

For example, human resource policy cannot be altered for one small group of individuals, and observations made about the appropriateness of such intervention. Even if it were feasible to practically do so, the results would be biased by the fact that there was a differential treatment; such an effect could not be separated from the experimental

effect. Additionally, there may be network and contagion effects that could impact the results from an organization-wide implementation that would not be observed in a small experiment. Thus simulation for prediction is common in organizations because large-scale change in organizations is difficult, and one wants to be relatively sure of a change's potential before investing greatly in the change effort.

Theory discovery Simulation can uncover phenomena that in turn focus theoretical attention. One concept that has gotten much attention in organization circles is the so-called "edge of chaos" (Brown and Eisenhardt, 1998). Theories concerning the "edge of chaos", and what it might have to do with running real organizations came about first because simulation made discovery of the phenomena possible, and second because people then reframed the structure of their simulations to reflect the conditions under which the "edge of chaos" emerged, using simulation in its predictive mode.

The edge of chaos was discovered in a particular class of agent-based simulations known as cellular automata (CA) (Waldrop, 1992). A CA model has entities (agents) that have certain states (e.g. on, off), connections between entities (typically local), and behavioral rules. A CA simulation evolves the state of system entities over time; for each entity, it uses the corresponding behavioral rules and the states of other entities to which it is connected to determine its future state. One such set of simulations could be described by a single parameter, "lambda" value. Researchers found that when lambda was near zero, the CA's tended to generate patterns that were simple; when lambda was near one, the CA's tended to generate patterns that were extremely complex. With an initial "random" starting condition, and a complex (lengthy) periodic cycle, these patterns

appeared to the human eye as random, or as they were (perhaps inappropriately) coined, "chaotic". As λ was brought into the middle region of the parameter space, geometry emerged that was termed "complex"--they had a pattern that was visible yet not immediately repetitive.

The conclusion was that "interesting things" happened at that region between boring repetition and high dimensional behavior--the so-called edge of chaos. From this, it was concluded that all life takes place at the edge of chaos. This interpretation in general has some logical appeal. If a system is too repetitive in its response to external stimuli, then it is possible to get "stuck" in an unhealthy spot without any hope of adaptation and continued survival. If the system is too variable in its response to stimuli, then it cannot predict anything to do with the effects of its future behavior, which leaves it helpless to develop internal models that are predictive in any effective manner. It is that region of system response that is neither too dead nor too haphazard that represents a healthy, adaptive state. Taken beyond the conceptual though, the "edge of chaos" concept is hard to prove in real living systems, and is yet to be universally accepted as an axiom of life. Nevertheless, it has inspired numerous organizational theorists to posit theories concerning organizations at the edge of chaos. For example, Brown and Eisenhardt (1998) talk extensively about how effective organizations balance the predictable and the unpredictable at the edge of chaos.

Performance With an appropriately calibrated and validated model, simulation can be used to perform real tasks for an organization, such as diagnosis or decision-making. In the organizational realm, simulation as a decision-aid is more likely to occur. In

organizational decision making, uncertainty and randomness are often a natural context of the system. While decisions requires taking uncertainty into account, this is not easily done with analytical formulations. Hence, simulation is used to mimic this uncertainty in the form of a Monte Carlo simulation. A Monte Carlo simulation is similar to discrete event simulation in approach, but does not emphasize the variable of “time”.

An example is the use of simulation models in project portfolio management (Cooper, 1993). Given a set of potential projects to engage in, but limited resources, an organization must decide what projects to invest its resources in. The portfolio of projects undertaken should maximize some benefit, such as return on investment; however such numbers can be difficult to predict using formulae, given that they depend on the confluence of many variables that are random in nature. For example, market penetration (for a new product), competitive actions, and development time, each of which could be described by a statistical distribution, may determine investment return. A Monte Carlo simulation would consist of hundreds or thousands of trials, each trial sampling from the distribution of the element specified, and then aggregate the composite answer; thus return would be characterized not by a single number, but by a distribution of possible outcomes.

Training A simulation environment makes it quick, easy, and safe for users to make decisions that mimic the decisions they (will) make in reality. “Flight simulators”—both the real and the so-called--fall into this realm. Two organizations where such simulation is used extensively are military organizations and nuclear plants. In both cases reaction to a given set of conditions must be rapid and yet calculated, almost to the point of

automaticity. Because these “crisis” situations are too difficult, expensive, and dangerous to set up in a physical context, computer simulation can be used, perhaps coupled with virtual reality. Such training can fulfill many different learning purposes:

- What type of environmental data is available for scanning? What are the important variables to pay attention to? What are the cause and effect mechanisms in place?
- How should decisions be appropriately framed?
- What is the timing and pacing required for decision-making?
- How will other people around me react?
- How can we effectively deal with the stress associated with such situations?

Education As opposed to using simulation for training purposes, simulation can also provide users more general education about how complex systems work. Users can gain a deeper conceptual, and perhaps metaphorical and symbolic understanding of mechanisms such as feedback, noise, reciprocity, self-organization, nonlinearity, etc. “Fractals” provide an example. Fractals are naturally occurring shapes with no clearly defined boundaries: clouds, trees, mountains, leaves, etc. While fractals have always existed, their operationalization was not possible until the advent of the computer, which allowed scientists to show how simple rules, iterated back into themselves, could produce such complex patterns.

At first glance the concept of fractals may appear as irrelevant to the world of organizations. Yet, people interested in organizations (especially organizational development researchers and practitioners) who observed these fractal shapes being constructed became inspired by the images they saw. They thought: organizations have

complex boundaries; organizations have self-similar structure across multiple scales. Perhaps if these complex shapes were developed from a set of iterated, simple rules, perhaps organizations are constituted by simple rules iterated repeatedly (Eoyang and Dooley, 1996)? Such simulations inspired organizational scholars and practitioners to learn more about complexity theory and dynamical systems in general, thus a simulation's educational benefits may have secondary value in that they may inspire further inquiry and learning.

Entertainment While the simulation "SimCity" is used for training city planners, it is also simply "played" by millions of users interested in seeing if they can build a metropolis. Educational benefits can accrue however from such entertainment uses (Rushkoff, 1996).

Proof Simulation can be used to prove existence of a possible solution to a problem. This is not in common usage in the social science realm.

Approaches to Simulation

There are three different approaches to simulation in the organizational sciences:

- *Discrete event* simulation models are best used when the organizational system under study can be adequately characterized by variables and corresponding states, and events occur that change the value of these variable states in some rule-oriented but stochastic manner. Discrete event simulation is not appropriate when state variables interact with one another and change on a continuous basis,

and when entities and their internal mechanisms are a more important element of the simulation than events, per se.

- *System dynamics* (or continuous) simulation models are best fit for situations where the variables in question are numerous, and can be related to one another in terms of how their rates of change interact with one another. System dynamic models tend to treat systems rather mechanistically, so it would not be the appropriate paradigm for modeling systems where individuals within the system were highly differentiated, or when behavior is best defined by people (and other entities) rather than the state variables themselves. System dynamics, as contrasted to the other two modeling approaches, is a “top-down” modeling approach, and thus requires fairly extensive knowledge about how the state variables of the system interact with one another.
- *Agent-based* simulation models are best fit for situations when the organizational system is best modeled as a collection of agents who interpret the world around themselves and interact with one another via schema. Agent-based models usually emphasize change in agents’ schema via learning and adaptation, and also highlight the phenomena of emergent, self-organizing patterns in complex organizational systems. Agent-based models are considered “bottom-up” models in that one describes individual agents and their patterns of connectivity and interaction, without necessarily knowing what might emerge as patterns of behavior in the larger, aggregate system.

Table 1 summarizes the basic characteristics of these methods.

--insert Table 1—

Discrete Event Simulation Discrete event simulation is a term used to describe organizational simulations that are discrete, dynamic, and stochastic (Law and Kelton, 1982). If a system can be adequately defined by some collection of variables that at any given moment characterize the “state” of the system, then a discrete system is one in which the state variables only change a finite number of times, at specific instances in time. If one were to plot the quantitative values associated with the state of the system, it would be piecewise continuous, making discrete jumps at particular times and then remaining constant for other periods of time. A model of a system is considered stochastic if its behavior is determined by one or more random variables. Models are dynamic to the extent that time is an over-arching component of the model and its corresponding behavior.

A discrete event simulation is characterized by the following elements (Law and Kelton, 1982):

- Entities: Objects that comprise the system
- System state: the state variables that describe a system at a given moment, often associated with specific entities
- Simulation clock: denoting the passage of simulated time
- Event list: a list specifying the events to occur in the future, and the time at which they will occur
- Statistical counters: for collecting data during the simulation run, to record history, and be analyzed later

- Initialization routine: some means by which to prepare the model for an experimental run
- Timing routine: a subroutine that manages the event list
- Event routine: a subroutine for each different type of event, that specifies the actions (creation of other events, change in state variables) that are associated with triggering of the event
- Report generator: reports the aggregate results as obtained from the statistical counters
- Main program: a program that coordinates activity between all of the various other elements of the simulation system.

Many of these elements are common across all types of simulations. They are often embedded in a user-friendly (often graphically-driven) software package.

March's (1991) influential study of organizational learning as exploitation and exploration (March, 1991) is an example of a discrete event simulation. Entities are considered individual people within the organization, and the state variables that define the system are an "external reality" (an m -dimensional vector made up of +1 and -1 values), an "organizational code" (a similar vector with different values), and n "individual codes". The system does not use a variable time clock, as is common in discrete event models, but rather a fixed clock (e.g. events are timed at regular intervals). As time moves forward, the external reality state vector remains constant, while parts of any individual's code evolve toward the state code according to some probability p_1 (learning from the organizational code), and parts of the organizational code evolve toward individuals with codes that are closer to the external reality than it itself is, with

some probability p_2 (learning by the organizational code). March reports results from an experimental design that fixes the dimension of the reality vector, the number of individuals, and the number of replications (to obtain results), and then varies the probabilities to determine the effect of different learning rates.

From this simulation come the following observations, which can then be appropriated into theoretical propositions:

- Higher rates of learning lead to achieving equilibrium in the system's states earlier.
- Rapid individual learning may not lead to rapid organizational learning.
- Having some small percent of individual "slow learners" may increase organizational learning rates, but having too many slow learners can decrease organizational learning rates.
- Moderate amounts of turnover, in some circumstances, can accelerate organizational learning.

One characteristic strength of March's simulation work is that explanations for the simulation results are always strongly coupled with—in a sense triangulated by—strong theoretical arguments.

System Dynamics System dynamics modeling grew out of the socio-technical systems movement in the 1950's, coupled with cybernetics and the desire to use systems theory (including the mathematics of Weiner's control theory) on problems in the social domain. It is best characterized by Jay Forrester's work in the 1960's; it had a "re-birth" of interest in the organizational world, following the popularity of Senge's (1990) book

“The Fifth Discipline” (which used the models of, but not the computational elements of, system dynamics).

Forrester (1961, p. 13) defines system dynamics (actually, industrial dynamics) as “the study of the information-feedback characteristics of industrial activity to show how organizational structure, amplification (in policies), and time delays (in decisions and actions) interact to influence the success of enterprises. It treats the interaction between the flows of information, money, orders, materials, personnel, and capital equipment in a company, an industry, or a national economy... It is a quantitative and experimental approach for relating organizational structure and corporate policy to industrial growth and stability.”

As in discrete event simulation, one has to define the variables that capture the state of the system. These variables need not be entities or physical items; nor must there be consistency in the *type* of variable that is chosen. For example, in simulating an information systems development team, one might define state variables such as group stress, the amount of workload present, morale of the team members, and external pressures and incentives for performance. In system dynamics, the next step is to characterize the relationship between the state variables, in terms of a functional equation relating one to the other. Using language from its historical roots, these state variables are often referred to as *sinks*, and the relationships between sinks are called *flows*.

There is one twist however to how these functional relationships, or flows are defined. Rather than being stated in the form of the state variables’ natural metrics, they are stated in terms of the first derivative of the state variable. Thus flows define how rates of change in one variable impact rates of change in another. This gives the

simulation a dynamical quality that would not otherwise be obtained. This also means that system dynamic models tend to be the only type to conform to McKelvey's (1997) plea for modeling rates rather than states.

Computationally what this means is that a system dynamics model is essentially a set of coupled, differential equations. More often than not, the differential equation is linear and thus system behavior, while it may be complex, is confined to periodic trajectories. One can use nonlinear flows and stochastic elements or events can be incorporated to enhance the complexity of the model.

An exemplar of system dynamics modeling is Sterman et al.'s (1997) work on paradoxes of organizational improvement. They used a system dynamics model to simulate the internal and external dynamics of the company *Analog Devices*, over a several year period, as the industry was going through trying times and the company itself was implementing a total quality management program. The model is quite complex—it is not uncommon to see system dynamic models with hundreds of variables—and contains variables such as the rate at which breakthrough new products were being innovated, market share, the effectiveness of process improvement, workforce commitment and morale, cash flow, pricing, labor variances, and R&D spending.

Sterman et al. first present a base case representing what actually happened to Analog Devices during the time frame in question. Analog Device's TQM program led to successful waste reduction in manufacturing, leading to excess capacity. Because new products were not available to take up this slack, the company laid off workers, eventually deteriorating the trust and support for the TQM effort.

Besides the intriguing subject matter, what makes this study stand out as an exemplar is the rigor attended to in validating the simulation's behavior to that of the real system. Unfortunately such care is rarely taken in simulation studies in organizational science. Sophisticated statistical methods for determining goodness of fit (between the simulated and real) were used and results clearly demonstrate high validity for the base case. This makes the subsequent experiments all the more believable.

From the base case, they explore several alternative lines of action, including maintaining a no-layoff policy, maintaining morale while downsizing, and maintaining operating margins. Key organizational performance variables are compared between the base and experimental case to determine the impact of such interventions.

Agent-Based Simulation Agent-based modeling is an outgrowth of artificial intelligence models in computer science. Whereas discrete event and system dynamic models focus on variables and events, agent-based simulation models focus on organizational participants (companies, teams, employees, etc.) and their larger collective behavior. Agents are considered “sentient beings” inside the computer, and are embodied with schemas that are both interpretive and behavioral in their nature. Since their advent, agent-based modeling has coupled with the discipline of complexity science (Anderson, 2000). As complexity science gains more acceptance as *the* normative description of living systems, agent-based simulation has the potential to become the dominant simulation-modeling paradigm for organizational researchers. Holland (1995) discusses the basic components of agent-based models:

- Internal models: schemas that are implemented computationally using behavioral rules. Adaptation is implemented via either genetic algorithms or schemes that change degree of belief parameters associated with behavioral rules.
- Building blocks: schema can combine with other schema into higher order behavioral models; a building block is the simplest form of a schema.
- Tagging: the explicit labeling of an agent in order to assign a class of attributes to the agent (i.e., through object-oriented methodology).
- Aggregation: agents can combine into meta-agents and behave as individuals and/or collectives.
- Nonlinearity: computationally, nonlinearity is implemented via rules that embed nonlinearity in agent behavior. Nonlinearity is a common but not required feature of agent-based models.

- Flows: computationally, flows represent the simultaneous change of attributes of agents.
- Diversity: implemented via the heterogeneity of internal models, tags and associated hierarchical aggregations, flows and agent attributes, and fitness functions.
- Fitness function: the utility function that determines how healthy each individual agent is. Fitness functions inform behavior because as certain rules become associated with certain outcomes that help or hurt the fitness value, these rules can be made to be evoked more or less often in the future (i.e. learning).

Cellular automata (CA) models are perhaps the simplest form of agent-based simulation, and thus serve as a useful demonstration of this approach. To simplify, consider a one-dimensional CA, where a "cell" represents an agent, and each agent possesses a "state". In the simplest arrangement, the state is considered binary, and can be represented visually by two numbers (0,1) or colors (white, black) (Dooley and Walker, 1999). The CA is finite in size, e.g. it has a fixed number of cells (agents). Each cell evolves its state in a discrete manner, such that the state of cell "j" at time t, $state(j,t)$ is determined by the state of cell "j" at time t-1, $state(j,t-1)$, and the state of some neighboring cells at time t-1, for example, $state(j-1,t-1)$ and $state(j+1, t-1)$. The cells are lined up in a specific and fixed spatial order. In order to visualize change in the states over time, the combined state vector is plotted down the screen, thus becoming a two dimensional object. Each column of the graphical object represents the evolution of a cell over time, and each row represents the combined states of all cells at a particular time.

Consider a 5-cell CA; assume that the cells at the left and right end "wrap-around" and consider their neighbor at the other end to be connected to them. We denote the

states by (0,1), and start-off with a random initial configuration: {0, 1, 0, 1, 1}. Now consider the following set of rules:

- If the sum of yourself and left and right neighbors is less than two, change to a 1.
- If the sum of yourself and left and right neighbors is two or greater, change to a 0.

Given the initial condition above, several iterations of the CA are shown in Figure 1.

--insert Figure 1--

By examining the types of patterns that exist over time (Is there a periodic pattern? If so, is it simple or complex? Is there sensitivity to small changes in the initial condition?) one can use CA models, and agent-based models in general, for the set of purposes outlined earlier. Studying the behavior of an agent-based simulation can be difficult though, because they demonstrate self-organization and emergence. (These two behaviors also can exist in discrete event simulation, but are rarely talked about in that domain.)

Behavior in a simulation is induced not by a single entity but rather by the simultaneous and parallel actions of many entities within the system itself. Thus, we refer to the simulation as self-organizing as it undergoes “a process . . . whereby new emergent structures, patterns, and properties arise without being externally imposed on the system. Not controlled by a central, hierarchical command-and-control center, self-organization is usually distributed throughout the system” (Goldstein, 1998). Self-organization leads to emergent behavior. Emergence is “the arising of new, unexpected structures, patterns, properties, or processes in a self-organizing system. These emergent phenomena can be understood as existing on a higher-level than the lower level components from which (emergence took place). Emergent phenomena seem to have a

life of their own with their own rules, laws and possibilities unlike the lower level components.” (Goldstein, 1998)

Self-organization and emergence are important organizational phenomena that are just beginning to be recognized and understood (Anderson et al., 2000). For example, the evolution of Napster.com, a free music-sharing Internet site whose presence eventually challenged the fundamental assumptions of the music industry, can best be understood as an emergent event shaped by the actions of semi-autonomous actors coupled together in a complex web of interaction.

Consider, for example, a simulation of bird flocking. At first glance, one may be tempted to believe that the complex order observed in the flocking pattern is the result of either a predetermined plan, or the result of unilateral control employed by the lead bird. In fact, flocking patterns emerge as part of the system’s self-organizing behavior (this has been proven to be true in reality). Individual birds behave according to simple rules that are enacted based on local information. Any individual bird determines their speed and direction by flying towards the center of the flock, mimicking the velocity of birds around them and staying a safe distance away from neighboring birds. We also observe this same type of self-organizing behavior in complex organizational systems, where simple behavior based on local information can, in the aggregate, lead to complex global behavior.

Many agent-based simulation models go beyond the simplifying assumptions of CA models, by incorporating fitness functions (measures of how well, or “healthy” an agent is; it’s utility function), learning and adaptation (typically incorporated by genetic algorithm-type formulations), systemic and institutional constraints (represented by non-

local connections), global influences, and agent idiosyncrasies (implemented by differentiating fitness function, connectivity, learning rates, and/or behavioral rules). One such model is Axtell's (1997) very detailed computational model of the emergence of firms.

In Axtell's model, an agent is an individual who self-organizes into a firm as it so benefits their own fitness function. First, each agent is assigned an "income-leisure" parameter that determines the amount of time it would prefer to spend working (making income) versus at leisure (any amount of time not allocated to work effort). These assignments are made randomly so agent heterogeneity is maintained. The agent's fitness is a multiplicative function of the amount of effort they are expending, the return on that effort at the firm level, and the amount of leisure time available. At random intervals of time an agent will consider their current position and either (a) maintain their current effort level, (b) increase or decrease their current effort level, (c) start a new firm (by themselves), or (d) migrate to one of two friend's firms. The amount of return that a firm gets from the aggregated efforts of its members depends on the number of members, their individual efforts, and any synergistic effects that may be included (e.g. increasing returns). Returns to the individual are equally divided among all members of the firm.

Despite the number of different parameters and starting conditions that can be varied, and the fact that the simulation itself is based on some rather simplistic assumptions about human nature, the corresponding results support the convergent and predictive validity of the model. For example:

- As firms become large, an individual agent's contribution becomes miniscule compared to the whole, which produces "free riders".

- As free-riding becomes more commonplace, firm productivity decreases and those who most want “work” leave for other firms, bringing the large firm into decline.
- While “equilibrium” conditions theoretically exist that would lead to a static situation, there is sufficient dynamic instability such that these equilibrium conditions never arise.
- The history of any firm is emergent and path dependent.
- At the aggregate level, firm size, growth and death rates mimic existing empirical evidence.

Axtell’s model demonstrates the potential power of computational models to mimic actual behavior in complex organizational and market systems.

There are at least two important methodological issues that must be considered when developing agent-based simulation models (Dooley and Cormann, 2000). First, it is difficult to evaluate the impact of structural and behavioral changes to such models, because emergent macro-patterns that depend on shifting micro-patterns tend to invalidate the assumption base of most modern statistical methods. One of the challenges in simulation research is to invent means by which emergence and self-organization in simulations can be visualized, operationalized and measured, and statistically modeled. The fractal nature of self-organization and emergence, both over time and space, make it difficult to use existing statistical methods that make restrictive assumptions about equilibrium, constancy of structure, and independence. This is an arena where advances in the modeling of these phenomena in the physical domain will come first, and be exported into the social science domain.

The second problem is related to the trade-off between model parsimony and model validity. Viewed as a simple trade-off, the analyst simply has to decide which perspective to favor and live with the consequences. There is a side effect--Bonini's paradox (1963)--induced by model complexity, however that tends to give favor to more parsimony and less model complexity for validity. The paradox states that as a model of a complex system becomes more complete, it becomes less understandable. Since understandability is major consideration in any simulation study, having too much complexity can be deadly for a simulation study.

The Implementation of Simulation Modeling

It is rare for a simulation study to be flawed because simulation was an inappropriate selection for a research method; or flawed because someone chose to do a simulation using one approach (e.g. discrete event) when another approach (e.g. system dynamics) would have been more appropriate. Simulation studies often falter, however, when it comes to implementation. This is especially true in disciplines where simulation has not been a common component of research training, as is the case in the areas of organizational structure, strategy, and behavior.

Implementing a simulation model involves the following steps and challenges (see Table 2 for summary):

--insert Table 2--

Conceptual design The user must determine what is to be modeled, what questions are to be asked via such a model, who will be using the model, and what their requirements

are. An elegant conceptual design will also simultaneously consider the medium in which the computational model is to be developed. In determining the conceptual design, preference is often given to models that are more parsimonious, that is, all things being equal, one should choose a simpler model over a complex model, unless the increase in complexity is well-justified.

A number of different design errors could occur during this stage:

- Behavior may be insufficiently specified, especially at the “boundaries” of the system.
- The researcher may include too many rules, thus making the model more difficult to test and validate, with little added value; or may exclude some important elements of the system (this is rare)
- Researchers may make erroneous assumptions about the sequencing of activities. For example, since many of the behaviors in an agent-based simulation occur simultaneously, but the computer is sequential, appropriate decisions must be made about sequencing and interval times.

In many papers there is a lack of cohesion between the theoretical propositions put forth and the model that is eventually developed. A simulation model is indeed the codification of a set of theoretical propositions, and in that sense is no different than the operationalization given to a set of survey items representing a construct. Researchers must pay careful attention to the alignment of their propositions, and the corresponding assumptions and boundaries of their theories, and the operationalization of those propositions, assumptions, and boundaries in the form of a simulation model.

In many ways the domain of the simulation model is much more limiting and constrained than the domain of theoretical propositions and assumptions. There are some things that simulation simply cannot do effectively; some concepts are not well addressed by computer modeling. In such instances it may be more effective to work backwards, that is, use the developed simulation model to suggest how specific theoretical propositions should be framed and stated.

Another problem often associated with the conceptual design of a simulation model is its relationship to reality—again a problem of aligning the research questions to the model. This is not to say that the model should attempt to mimic reality--that would be self-defeating. Rather, the modeler should make choices about the simulation that make it more realistic, thus enhancing its potential validity and generalizability. For example, a simulation model of behavior in a cross-functional team can use a model that differentiates the agents in an essentially "blind" and uninformed manner; alternatively, one can specify what functions would realistically be working together, and then differentiate them in a specific manner that relates to what we know about such differences from other studies and theories.

Code development The computational model is implemented in software code and tested. Today, simulation languages exist that enable the user to write statements that then refer back to lower-level computer languages such as Fortran or Basic. These second generation languages reduce the amount of code that needs to be developed by an order of magnitude, or more. Even more sophisticated software packages enable a user to construct a simulation model completely using click, drag, and drop objects on a

computer screen. In general, the more user-friendly the modeling environment, the more complexity is hid from the user and therefore difficult to access, if indeed such complexity is needed. Most research projects require the flexibility of the more general languages.

One problem present in many simulation studies is a lack of discipline concerning the development of what often turns out to be a complex and very large software project. There is a lack of appreciation for the software development process; indeed researchers typically believe that all is required to “get going” is to learn the intricacies of the particular simulation language. For small models this might suffice, but for large models an ad hoc development process is more likely than not to lead to significant errors (which may or may not ever be discovered), thus leading to loss of productivity and/or validity.

Researchers involved in the development of large simulation models should follow basic software development practices (McConnell, 1997), such as:

- Maintain a documented set of requirements.
- Implement a change control process, whereby changes to requirements and/or code are documented, dated, and explained; and a corresponding document (or version) control process.
- Develop the architecture independently of the features of the model, to the extent possible.
- Reuse code (that has been validated for its quality) whenever possible. Write code for future potential reuse.
- Develop code in a modular fashion, to the extent possible, so that it can be easily reused, tested, and integrated.

- Use a variety of testing methods, including code walk-throughs, scenario testing, and user testing.
- Develop a project plan for coding and testing.
- Use trained personnel only.

Validation A simulation might be technically without error—but it may have no truth, or validity associated with it. Model validity concerns how close the computed behavior is to the “real” answer. In some cases there is no real answer and validity can only be determined “at face”, by an expert—do the results make sense? Sometimes though there may be actual quantitative behavior that can be compared to. There are three ways in which the computational model can match quantitative system history—exactly, distributionally, or pattern-wise. Since most simulations involve random variables, one rarely seeks an exact fit. A distributional fit means that a variable of interest demonstrates the same mean and covariance structure as seen in reality. A pattern-wise fit means variables are generally related to one another in a valid manner, but their actual numerical values have no relationship to real behavior. Validity checks are too often found missing from computational studies of social systems.

Experimental Design The user must design a set of experiments, indicating particular initial and run-time parameter values that will be used to determine answers to questions posed. This is another area where there is often a mismatch of between the goals of the research embodied in the research questions and theoretical propositions, and the operationalization of that in an experimental design. There is considerable room

for improvement of rigor and use of state-of-the-art methods in organizational studies. At the present time I am hard-pressed to find even a single example of high quality, let alone exemplary experimental design in the organizational studies area. The dominant mode of experimentation in such studies is “one-variable-at-a-time”, which is known to be a woefully inadequate methodology (Box et al., 1978), incapable of determining interactions between variables, and dismally inefficient.

Researchers should be drawing from the family of experimental designs known as full and fractional factorials, and other similar experimental design models (Box et al., 1978). These experimental designs enable the researcher to examine the effects of specified variables in a cost effective manner. They also generate data that is “optimized” for ease of analysis and interpretation (e.g. estimated statistical effects of variables are independent). Additionally, researchers should differentiate experimental variables into control factors and nuisance factors, and use Taguchi’s methods (or variations thereof; see Dooley and Mahmoodi, 1992 for example) to determine robustness of their results. Good experimental design not only makes the discovery process more efficient, but it makes analysis easier (“cleaner”), and gives one the potential to begin to see quasi-nonlinearities (in terms of variable interactions).

Implementation The user must execute the experimental design, which will likely include replicates; a replicate is a multiple-run of a single experimental condition, but using a different stream of random numbers. There are various ways of managing random number streams associated with the stochastic elements of the model that make it more or less easy to make comparisons (Law and Kelton, 1982). Another issue that

must be addressed is the transient behavior of the simulation. When a simulation model begins to run, it is initialized to some starting condition (often randomly). There is a period of time when observed the observed behavior is overwhelmingly due to the particular initial configuration—this data should be ignored, unless one is specifically interested in transient behavior. It is assumed that after some time, the simulation behavior will reach a “steady state”, whereby further observation is unlikely to yield new information. First, researchers should be aware that there are sophisticated methods that can be used to determine when transient conditions have curtailed. Second, the notion of “steady-state” was developed in a world of equilibrium-based systems, and therefore may have not as much relevance (or more specifically, a different type of relevance) for complex, nonlinear models such as found in agent-based models.

Analysis Replicates are averaged over an experimental condition for subsequent analysis, and the standard error of replicates can be used to determine confidence intervals for other subsequent work. The user analyzes the output data according to the model being hypothesized.

Again, practice in the organizational research area lags behind other disciplines with respect to analysis of simulation results. Often, no statistical analysis is done at all—a graph of the time-oriented behavior for the different experimental cases is shown. This is adequate for system dynamic models that are completely deterministic (and thus there is not uncertainty associated with the outcomes), but it is not adequate for simulations with stochastic components. Under such circumstances, several “problems” might exist that should be addressed by appropriate and powerful statistical techniques:

- There may be correlation between the averages of different replicates, thus necessitating the need to take into account the covariance structure when doing hypothesis testing.
- There is almost certainly correlation between the different variables of interest within a single replicate. This correlation structure should be analyzed and understood. Coupling this issue to the previous statement, researchers may need to use techniques such as MANCOVA in order to appropriately analyze their results.
- Transients (temporal behavior that is solely due to the initial configuration of the simulation) should be identified and removed.
- There is almost certainly autocorrelation within many of the variables temporal behavior. This does not cause a problem with calculation of the mean, as the mean is unbiased by such dynamical behavior. Variance estimates, however, are likely to be inflated and thus make comparisons more difficult. The autocorrelation structure of the corresponding time series should be taken into account when making a variance estimate.
- To go further, this autocorrelation structure should be analyzed as an outcome of the simulation itself. Knowledge of internal dynamics may well inform the researcher of generative mechanisms that would otherwise be blind to them (Dooley and Van de Ven, 1999).

Interpretation Observations from analysis are noted, and results are discussed in order to sense-make. A common mistake is over-interpretation of the results, as the researcher attempts to link the simulation results back to the theory(ies) of interest.

Conclusions

Simulation is a powerful research method that enables researchers to look at an artificial world move forward into the future, giving the user the unprecedented opportunity to intervene and attempt to make improvements to performance. As such it is a laboratory, safe from the risks of the real environment, for testing out hypotheses and making predictions.

For organizational researchers, simulation can be used as a theory testing mechanism, although it is perhaps more frequently and successfully used in this domain as a theory building mechanism. Whereas other research methods discussed in this book help researchers answer questions such as “What happened? Why? How?”, simulation is best used to answer the question “What if?”.

Three different approaches to simulation modeling have emerged over time. Discrete event simulation is historically the most common, and assumes the organizational system can well be described as a “machine” where inputs (entities) arrive and are transformed (by events) into outputs (entities). Uncertainty associated with “real life” is implemented in the form of random variables. Discrete event simulation is good for studying the more mechanical, predictable, and orderly elements of the organization.

System dynamics models also treat the organizational system as somewhat mechanical, and perhaps even deterministic. To the extent that one can describe an organization by its constituent states and their interrelationships in terms of rates, system dynamics offers a rich modeling paradigm. Corporations oft use system dynamic models for strategic planning purposes, but rarely do such efforts reach the light of day of the public. Researchers would be best to use system dynamic models for specific purposes and instantiations; its usefulness on “generic” problems is marginal. Therefore, if one was going to study “organizational learning” using system dynamics, one would be much more likely to obtain useful results if the developed model came from observation of a real organization. System dynamic models that are created abstractly rarely inform.

Agent-based models treat the organizational member as all-important, and expect that organizational behavior will somehow emerge from the coupled interactions of organizational members. These models borne from artificial intelligence have merged with complexity science to offer a more complete and theoretically sound landscape within which to develop interesting and informative models. Agent-based models are probably best suited to answer the typical questions that organizational researchers have; they also probably have a steeper learning curve, as the field is still relatively young.

I have made editorial comments in this chapter noting the disparity between the state-of-the-art in simulation research, as exhibited in other disciplines such as operations management, industrial engineering, and computer science, and current practice in the organizational research area. In particular, organizational researchers must pay much closer attention to the following issues:

- Alignment of theory and model

- Testing of code, and disciplined project management of the simulation (software) development process
- Validation of model and results
- Rigorous experimental design
- Appropriate and rigorous statistical analyses.

I hope this overview increases the frequency and rigor of simulation research in the organizational sciences.

Acknowledgements

This work in part has been sponsored by the *Laboratory for Organization, Communication, and Knowledge Studies* (LOCKS), at Arizona State University.

Bibliography

- Anderson, P., “Complexity theory and organization science’, *Organization Science*, 10 (1999), 216-232.
- Axelrod, R.: “Advancing the art of simulation in the social sciences,” Santa Fe Institute, paper no. 97-05-048 (1997).
- Axtell, R.: “The emergence of firms in a population of agents,” Santa Fe Institute, paper no. 99-03-019 (1999).
- Bonini, C. P.: *Simulation of Information and Decision Systems in the Firm* (Englewood Cliffs, NJ: Prentice-Hall, 1963).

- Box, G., Hunter, J., & Hunter, W.: Statistics for Experimenters (NY: Wiley and Sons, 1978).
- Brown, S., and K. Eisenhardt: Competing on the Edge, (Cambridge, MA: Harvard Business School Press, 1998).
- Cooper, R.G.: Winning at New Products (Reading, Massachusetts: Addison Wesley, 1993).
- Dooley, K., and S. Corman, "Agent-based genetic and emergent computational models of complex systems," Encyclopedia of Life Support Systems (forthcoming).
- Dooley, K., and F. Mahmoodi: "Identification of robust scheduling heuristics: Application of Taguchi methods in simulation studies," Computers and Industrial Engineering, 22 (1992), 359-368.
- Dooley, K., and A. Van de Ven: "Explaining complex organization dynamics," Organization Science, 10 (1999), 358-372.
- Eoyang, G., and K. Dooley, "Boardrooms of the future: The fractal nature of organizations," Fractals Horizons: The Future Use of Fractals, Cliff Pickover (ed.) (St. Martin's Press, 1996, 195-203).
- Forrester, J.: Industrial Dynamics (Cambridge, MA: Productivity Press, 1961).
- Goldstein, J: "Glossary", in Zimmerman, B., Lindberg, C., and P. Plsek, Edgeware, (VHA: Irving, TX, 1998), 270.
- Holland, J.H.: Hidden Order (Reading, MA: Addison-Wesley, 1995).
- Law, A., and D., Kelton: Simulation Modeling and Analysis (NY: McGraw-Hill, 1982)

- March, J.: "Exploration and exploitation in organizational learning," *Organization Science*, 1 (1991).
- McConnell, S.: *Software Project Survival Guide* (Redmond, WA: Microsoft Press, 1997).
- McKelvey, B.: "Quasi-natural organization science", *Organization Science*, 8 (1997), 351-380.
- Rushkoff, D.: *Playing the Future*, (Harper-Collins, 1996).
- Senge, P: *The Fifth Discipline* (NY: Doubleday, 1990).
- Sterman, J., Repenning, N., and F. Kofman: "Unanticipated side effects of successful quality programs: Exploring a paradox of organizational improvement," *Management Science* 43 (1997), 503-521.
- Waldrop, M.: *Complexity* (Touchstone Books, 1992).
- Walker C. and K. Dooley: "The stability of self-organized rule following work teams," *Computational and Mathematical Organization Theory*, 5 (1999), 5-30.

Table 1 Characteristics of Three Different Simulation Approaches

Simulation approach	Conditions for use	Main characteristics
Discrete event	System described by variables and events that trigger change in those variables	Events that trigger other events sequentially and probabilistically
System dynamics	System described by variables that cause change in each other over time	Key system variables and their interactions with one another are explicitly (mathematically) defined as differential equations
Agent-based	System described by agents that react to one another and the environment	Agents with schema that interact with one another and learn

Table 2 Issues and Challenges in Organizational Simulation

Implementation step	Key decisions	Key decision criteria	Common problems	Follow-up resources
<i>Conceptual design</i>	Boundaries of system, key variables, agent/entity attributes	Driven by theoretical propositions or research questions	Lack of linkage between model and theory(ies) of interest	Exemplars: March, 1991; Sterman et al., 1997
<i>Code development</i>	How to effectively manage the software development project	Effective project management practice	No explicit requirements; poor change control; little reuse of code; untested code; lengthy development times	McConnell, 1997
<i>Validation</i>	Match between simulation results and real system	Extent to which validity is important	Validity completely ignored	Axelrod, 1997; exemplar: Sterman et al., 1997
<i>Experimental design</i>	How to manipulate variables across experimental conditions	Effective experimental design practice	No (formal) experimental design used	Box et al., 1978
<i>Implementation</i>	Initial configuration, transient versus steady-state behavior	Statistical theory	Simulation runs not long enough; or not enough runs	Law and Kelton, 1982
<i>Analysis</i>	Determination of significant effects	Statistical theory	Transient and steady-state behavior intermixed; independence assumptions violated; dynamics not examined	Law and Kelton, 1982; Dooley and Van de Ven, 1999
<i>Interpretation</i>	Link simulation results back to theory	Logical arguments	Over-interpretation of results	Exemplar: Sterman et al., 1997

Figure 1 Cellular Automata Example

Random initial configuration:	0	1	0	1	1
After first iteration:	0	1	0	0	0
After second iteration:	1	1	1	1	1
etc.	0	0	0	0	0
	1	1	1	1	1
	0	0	0	0	0

Prof. Kevin Dooley has a joint appointment with the Department of Industrial Engineering and the Department of Management at Arizona State University. He has a Ph.D. in Mechanical Engineering from the University of Illinois, in 1987. Prof. Dooley's research interests lie in the areas of complex systems theory, quality management, innovation & new product development, organizational change, knowledge management, text analysis, information technology, and health care management. He is currently President of the *Society for Chaos Theory in Psychology and the Life Sciences*, and international society devoted to applying complexity science to the study of living systems. He is on the editorial boards of *Journal of Operations Management*, *Quality Management Journal*, *Journal of Quality Management*, *Nonlinear Dynamics*, *Psychology*, & *the Life Sciences*, *Production and Operations Management*, and *Emergence*. He has consulted with over 100 companies in the areas of quality, organizational change, and innovation.