The Integration of Simulation and Knowledge Systems: A Semiotic Perspective

Henk W.M. Gazendam
Faculty of Management and Organization
Groningen University

Faculty of Public Administration and Public Policy
University of Twente

Paper presented at the 7th Mini Euro Conference on Decision Support Systems, Bruges, March 24-27, 1997

Address: prof.dr.H.W.M.Gazendam, Faculty of Management and Organization, P.O.Box 800, NL-9700-AV, Groningen, The Netherlands

Keywords: simulation, knowledge systems, knowledge-based simulation

Introduction

In a time of rapid change of technological possibilities and of internationalization, organizations need to experiment with these new possibilities, and to understand their potential for change and improvement. This potential for change and improvement can only be revealed by organizational thinking, thinking using existing and novel organization concepts. Organizational thinking can be stimulated and supported by formalization of existing theories, in order to analyze their range of expressiveness and logical structure (Gazendam, 1993), as well as by simulation models that show emergent organizational phenomena, especially related to interagent communication and knowledge in organizations.

Discrete event simulation models are traditionally strong in showing emergent organizational phenomena, but lack aspects of reasoning using explicit knowledge, and of learning. Rule-based systems like expert systems or symbol system architectures like Soar typically are strong in rule-based reasoning, that is, utilization of explicitly formulated knowledge, but are weak in showing emergent phenomena, while learning is restricted to remembering rule patterns that have been used. Moreover, these rule-based systems generally use unlimited temporary memory resources in the form of stacks and similar memory structures. It is also very attractive to implement efficient search algorithms, heuristic algorithms, and optimization algorithms. The use of such temporary memory structures and algorithms is not only psychologically implausible, and therefore unattractive in a realistic model of organization, but also inhibits learning. Learning is based on the -relatively slow- building of an efficient world model that can be used by simple algorithms in order to perform tasks efficiently. Strong algorithms and extensive temporary memory structures do not need such a world model and, therefore, do not necessitate or stimulate learning. Matrix-based learning systems, like neural nets and classifier systems, use more efficient memory structures for storing experience, are relatively efficient in learning the most efficient way to perform a task, but are weak in reasoning based on explicit knowledge and weak in showing emergent organizational phenomena. Matrix-based learning models the acquisition of tacit knowledge, of the most efficient association between perceived features and actions that should be chosen as a reaction on these features. There is a gap between this realm of tacit knowledge and the world of explicit, symbolic knowledge used in communication and reasoning. Therefore, a combination of these three modeling techniques, using the strong points of each method, seems to be a promising candidate for an approach to the modeling of organizations in which knowledge as well as emergent phenomena are important. This integrated technique is generally known as knowledge-based simulation (Kreutzer, 1986).

Three projects in the field of knowledge-based simulation

Below, we discuss three projects in the field of knowledge-based simulation: the Information Strategy Model (ISM), the Beehive model, and the Multi-Agent Simulation of Organizations (MASMO).

The Information Strategy Model (ISM)

The Information Strategy Model (ISM) is a multi-agent organization model written in Smalltalk80 (Gazendam, 1990). It simulates the choice and implementation of information management strategies. The basic structure of the Smalltalk80 discrete event simulation classes is used to model agents, objects, resources and blackboards. One of the agents, the information manager, has to make a yearly decision about what strategy to choose. The characteristics of the chosen strategy like minimum and maximum size of projects, the hiring of external consultants or of internal employees (the outsourcing decision) are placed on a blackboard on which all simulated project managers will look if they have to make an operational decision. The information manager uses a personal knowledge for reasoning about the strategy to choose. The knowledge bases uses HUMBLE, a Smalltalk expert system shell with MYCIN-like features (Piersol, 1985). The strategic options modeled in the knowledge base stem from an in-depth investigation of the information strategies in a government agency (Gazendam, 1993). After the information manager’s yearly decision, the simulation model goes on with simulation of the implementation of the strategy. After a year, the information manager decides again, based on the results that have been delivered in the previous year.

The main results of ISM are insight in the stability or instability of strategies and the effects of long implementation trajectories on strategy choice. In some cases, cyclic patterns could be observed due to the feedback of the results of the previous strategy into the choice of the next strategy. In some cases, a new strategy could not be implemented directly because of the effects of previous strategies incorporated in running projects. In this way, a previous strategy can be a burden for a newly chosen strategy.

In the architecture of ISM, the interface gives access to the simulation environment. The reasoning in the knowledge system is accessed by the agents in the simulation environment.

Although the coupling of simulation and knowledge systems in ISM is a successful one, it is not satisfactory in a number of ways. Firstly, there is no learning. The rules used by the expert system are rigid. Rigid rules can lead to unsatisfactory cyclic strategy patterns, while it can be assumed that a more flexible rule base would lead to some kind of relatively stable optimal strategy. Secondly, there is only one agent that is more or less ‘intelligent’ (the information manager). He has unlimited access to all data in the system. This is a kind of omnipotent agent that is not very satisfactory for a simulation of a real organization where knowledge is distributed over agents, and agents only have limited access to available data (the access to data is limited by their observation horizon).

The Beehive simulator

In the Beehive simulator, simulation and knowledge system are more loosely coupled (Rutges and Vens, 1991). The user interface gives access to an animation environment and an advisor window. In the animation environment, the user can see beehives, supers within beehives, and frames within supers. He can manipulate these objects and see the effects on the population of eggs, brood, and bees. This animation is backed up by a discrete event simulation environment in which the ‘laws of nature’ with respect to beehives and bee populations have been implemented. The advisor window gives access to an expert system that holds knowledge about abnormal growth and diseases of bee populations. It is used to diagnose an abnormal situation and to propose a plan for manipulation. The combination of animation and expert advice turns out to be a good medium for education in the field of bee management.

The Beehive simulator has been implemented in ParcPlace Smalltalk, using Humble as the expert system shell.

In the Beehive simulator, there is no direct coupling between simulation and knowledge system. If the user wants to add extra rules based on his experiences with the simulation he can do so. This is not done in an automatic way. When the expert system gives a diagnose and an advice for treatment, this is not implemented automatically, but the user has to do so. There is always an active role for the user of the system. This kind of relationship shows us that there are two possible relationships between simulation and knowledge system: the simulation is feeding the knowledge system with new information that can be added to the rule base, or that modifies the parameters in the rules. The knowledge system can be used to control` the simulation. If both types of relationship would be implemented in an automatic way, this would result in feedback loops that can give rise to development patterns that are stable or unstable (cyclic or even chaotic), as observed in the ISM model.

The Beehive simulator shows us that there can be a two-way coupling between simulation and knowledge system. The Beehive system has been successful, especially because of the animated interface and the extensive knowledge incorporated in the system. It does not, however, couple simulation and knowledge system in an automatic fashion. It also uses only one agent that can reason, the omnipotent agent that is implausible in simulations of real life organizations where knowledge and data is distributed.

Multi-Agent Simulation of Organizations

The multi-agent simulation of organization (MASMO) project (Van den Broek and Gazendam, 1996)tries to overcome the problem of the omnipotent agent by taking a multi-agent perspective as a starting point (Gazendam and Jorna, 1993; Gazendam and Homburg, 1996). Each agent has his own private knowledge, data, and goals. Agents have to obtain data about the environment they wander in by a trial and error process or by communicating with other agents. Knowledge, therefore, is distributed, and information about the environment is imperfect. The MASMO simulation has been implemented in Soar (Newell, 1990). Agents learn in two ways: (1) by remembering what they did (the mechanism of chunking), and (2) by building a world model consisting of states (in terms of features), possible actions, and associations between features and actions.

The environment is modeled as a separate object embodying the laws of nature of the task environment the agents wander in. MASMO uses the Cohen (1992) task as a basis for the task environment. The task environment is modeled in a generic way based on a finite state machine. Each state of the task environment finite state machine corresponds to features shown and actions possible.

In the beginning, agents are clones of a template in which basic capabilities are defined. Agents have to explore their environment and to cooperate to reach their goal. By getting experience, agents will learn, accumulate new knowledge, differentiate, and a form of knowledge distribution appears.

The MASMO model is a model that is psychologically plausible at the agent level, and that has the potential to be organizationally plausible as well because of the principles of distributed knowledge and imperfect information. Once you distribute knowledge, however, problems arise: how to model communication, and how to achieve cooperation (if necessary). Experimentation with MASMO shows that various deadlocks in communication and cooperation can appear and have to be overcome by developing a common language of signals and symbols. The model, therefore, investigates the area where the learning of tacit knowledge (in the form of world models, a kind of matrix learning) and the development of explicit, symbolic knowledge are touching each other. Because of this focus on rather detailed level problems of learning, communication, and cooperation, the MASMO model is rather poor in its possibilities to model real life organizations with their business processes of task fulfillment and negotiation between actors. For this, we need a further development of discrete event simulation in combination with knowledge systems.

Integration problems

The experiences in these projects show that the integration of discrete event simulation, rule-based reasoning, and learning encounters difficulties in the field of incompatible model architectures, knowledge characteristics and data structures.

Architecture Issues

Knowledge systems and simulation are inherently different in the way how they approach initiative. In knowledge systems, initiative is mainly located at the interface where the interaction with the user produces most events. A knowledge system is based on the tool metaphor. The knowledge system is driven by the user inputs, and a simulation shell would be only an extension of the reasoning system, perhaps to test the effects of proposed actions. In simulation, initiative is mainly located at the simulated world where objects generate events. Simulation is based on the virtual world metaphor. The simulated world is depicted in an animated interface, where the user can participate by the manipulation of objects. The simulation is A knowledge system component can have no predominant place here; perhaps it can advise the user like is done in the Beehive simulator. There is no natural link between a knowledge system that operates centralized and a simulation system that operates decentralized. Two links are possible: at a central level and at a decentral level. Linked at a central level, the knowledge system is either (1) an isolated advisor to the user, or (2) the dominant part of the system that uses simulation for finding out the effects of a course of action. Linked at a decentral level, each simulated intelligent agents uses its own knowledge system and world model. This means a an implementation in the direction of distributed knowledge and distributed data, thus leaving an important characteristic of most knowledge systems.

Knowledge Issues

Simulation, rule-based knowledge systems, and matrix learning-based systems differ in the way they approach knowledge in the form of information about objects, rules of behavior, goals, time structure and space structure. In simulation, information about objects is distributed as well as rules and goals. Time structure and time constraints are handled centrally. In most discrete event simulations, there is no explicit representation of space. In rule-based systems, information about objects, rules and goals are centralized. In most cases, there is no explicit representation of time and space. In matrix-based learning systems, only rules are represented, and this is done centrally. The other aspects are nor represented explicitly (Goals are implicit in the training of these systems). For knowledge-based simulation, we would prefer to have information about objects to be decentral and bound to the observation horizons, and observation efforts, of agents. Rules should be distributed, each agent having its own rules obtained through inheritance of 'genetic information' or obtained by experience. Goals should be distributed as well, maybe except some basic rues concerning basic needs. For the maintenance of the rules of nature in the simulated world, there should be centralized control of time and space constraints and space and time structure (see table below).

Model type	*Information about objects*	*Rules*	*Goals*	*Time constraints*	*Space constraints*
Simulation	Distributed	Distributed	Distributed	Centralized	Not represented
Rule-based systems	Centralized	Centralized	Centralized	Not represented	Not represented
Matrix-based learning	Not represented	Centralized	Implicit in training	Not represented	Not represented
Knowledge-based simulation	Distributed	Distributed	Distributed	Centralized	Centralized

Data Structure Issues

Because of their centralized nature, knowledge systems tend to use a state space representation of information, based on dimensions or attributes. Rules connect ranges of values on these attributes. The entities, attributes and hypotheses used in reasoning in a rule-based system tend to be temporary, their existence restricted to one instance of a reasoning process. Simulation, on the contrary, uses an object-oriented model of information preventing a direct access of all attributes in the system. The objects in simulation are semi-permanent, that is, they exist until they have a reason to die or to exit the simulated world. In normal discrete event simulation, objects are clones. Their methods and rules reside in a class, of which each object is a clone. Only data are private in a normal discrete event simulation shell, thus preventing easy attachment of knowledge in the form of rules to an individual object. These differences in the field of data structure prevent an easy technical integration of discrete event simulation with knowledge systems.

Why a simple integration is not simple

The most simple solution to the problem of integration of simulation with knowledge systems would be to use the scheme in Paragraph 3.2. for knowledge-based simulation and implement a knowledge base and a matrix learning system for each simulated agent. For two reasons, this simple solution is not as simple as it seems at first glance. Firstly, the information in the database part of knowledge base has a recursive nature, and secondly, components like simulation, agent knowledgebase, and agent matrix learner behave differently in time, demanding a separate implementation.

The information in the database part of the knowledgebase has a recursive nature, as has been well-known in the logical field of reasoning about knowledge for a long time. An agent has a model of all other agents and of himself. In the model an agent A has of another agent B, it may be included that A knows that B has a certain model of several other agents, and so on.

The simulated world, the agent knowledgebases, and the agent matrix learners behave differently in time. The representations used in reasoning by an agent may be of a temporary nature and should not be taken as more or less identical to the real (simulated) objects. For instance, an agent may be reasoning about different strategies with regard to a thing, making necessary several representations in his mind of this thing, while, at the same time, the facts about the thing in the observed (simulated or real) world may stay the same. The behavior towards an object represented in the matrix learner may have an evolution that is relatively slow compared to the evolution of the simulated world (for instance, when basic behavior patterns are concerned), or may be changing fast in an intensive process of learning by doing.

The recursiveness of representations that an agent has of other agents, and the differences in time scales of simulated world, rule-based reasoning, and matrix learning makes it necessary to design a careful pattern of relations between the components in a knowledge-based simulation. A careful design is also necessary to avoid overcomplex systems that tend to result from the integration of different modeling techniques.

An architecture for the integration of simulation and knowledge systems: KANT

For solving the problems of integration of simulation with knowledge systems mentioned above, knowledge-based simulation systems should embody the following developments:

- from tool-based system to virtual reality system;
- from centralized information, rules, and goals to decentralized information, rules, and goals;
- from attribute-based representations to object-based representations.
- from agents as clones (only data are different) to agents as individuals (data, methods and knowledge are different).

Moreover, knowledge-based simulation needs control of time and space constraints in a centralized manner. While all other aspects of knowledge may be distributed, these Kantian dimensions need to be controlled at a metalevel.

The KANT (Knowledge Agents Never Tired) architecture for knowledge-based simulation embodies these developments and requirements. The way it links the simulation component, and the rule-based system component is based on ideas of the semiotician Charles Sanders Peirce (1958). Peirce distinguishes three aspects of signs: the *sign* itself (the word or symbol structure), the *signified* (the thing or other entity that the sign is referring to), and the process of *semiosis* (the dynamic concept enabling to link sign and signified). We can interpret the simulated world as being full of entities that have to be referred to. Agents use concepts in their rule-based reasoning. To link their perception of the simulated world with the concepts used in reasoning they need a dictionary of signs. Copies of these signs can also be used in communication with other agents. In this way, the sign dictionaries of the agents become the heart of the KANT architecture.

A first version of the KANT knowledge-based simulation environment has been implemented in the object-oriented language Smalltalk/V. The simulation part of the KANT system is a discrete event simulation shell based on the Goldberg/Robson (1983; 1989) and Birtwistle (1979) architectures. KANT adds the concept of simulated space to this in order to implement observation horizons of agents and centralized enforcement of time as well as space constraints.

Each agent in KANT can have his own knowledgebase and matrix learner. The knowledgebase system part of KANT consists of an expert system shell (a modified version of HUMBLE) offering probabilistic reasoning. The Mycin-like shell HUMBLE has been revised, amongst others to accommodate for multiple object hierarchies.

The matrix based learning abilities of agents have been implemented in the Agent subclasses of the SimulationObject class. These abilities are therefore fully integrated in the simulation environment. The matrix-based learning is aimed at choosing a preferent action given a certain situation. Reasoning outcomes are part of this situation. The matrix-based learning abilities have been implemented as a sort of classifier system. This matrix-based learner has to be tuned. Tuning encompasses for instance the type of heuristic algorithms available, the type of reinforcement mechanism, the strength of freezing a certain behavior pattern once positive results have been obtained versus the strength of random choice of behavior, and the size of memory to be used in matrix learning. A replication mechanism for the automatic tuning of agents using a genetic algorithm type of selection mechanism is under development.

Conclusion and Perspectives

Simulation models of organizations need knowledge systems to represent and handle agent knowledge. Knowledge systems need simulation to overcome problems in the field of distributed knowledge. Integration of simulation and knowledge systems is possible but leads to problems in the field of architecture, knowledge, and data structures.

For solving these problems, knowledge-based simulation systems should embody the following developments:

- from tool-based system to virtual reality system;
- from centralized information, rules, and goals to decentralized information, rules, and goals;
- from attribute-based representations to object-based representations.

- from agents as clones (only data are different) to agents as individuals (data, methods and knowledge are different).

Moreover, knowledge-based simulation needs control of time and space constraints in a centralized manner. While all other aspects of knowledge may be distributed, these Kantian dimensions need to be controlled at a metalevel. The KANT architecture for knowledge-based simulation embodies these developments and requirements. A first version has been implemented in software.

A new interface for the KANT environment is under development. The KANT system will be accessible via the Internet, address http://www.bdk.rug.nl/mais/ after May 1st, 1997.

The KANT knowledge-based simulation environment is being applied to a project aiming at the sharing of legal knowledge in a government agency. In the near future, KANT will be modified for use in business process innnovation projects and rapid application development projects.

Literature

Birtwistle, G.M.

1979 DEMOS: A system for discrete event modelling on Simula. Houndmills: MacMillan.

Broek, Hans van den, and Henk W.M.Gazendam.

1996 “Explorative Agents: Learning as the Development of World Models.”, *Unpublished Research Report, Faculty of Management and Organization, Groningen University, December 1996.*

Cohen, M.D.

1992 “When can two heads learn better than one? Results from a computer model of organizational learning.” In: M. Masuch & M. Warglien (Eds*.). Artificial Intelligence in Organization and Management Theory: Models of Distributed Activity.* Amsterdam, North-Holland.

Gazendam, Henk W.M.

1990 "Expert Systems Supporting Organization and Information Management." In: Masuch, Michael. (ed.). *Organization, Management, and Expert Systems: Models of Automated Reasoning.* Berlin: De Gruyter: 123-153.

1992 "Organization Theories as Grammar." In: M.Masuch and M.Warglien (eds.). *Artificial Intelligence in Organization and Management Theory: Models of Distributed Activity.* Amsterdam: North-Holland: 41-63.

1993 *Variety Controls Variety: On the Use of Organization Theories in Information Management*. Groningen: Wolters-Noordhoff, 1993.

Gazendam, Henk W.M., and Vincent Homburg

1996 "Emergence of Multi-Actor Systems: Aspects of Coordination, Legitimacy and Information Management." *Proceedings of the COST A3 Conference 'Management and New Technologies', Madrid, June 12-14, 1996*: 323-327.

Gazendam, Henk W.M., and René J.Jorna

1993 "Theories about Architecture and Performance of Multi-Agent Systems." *Paper presented at the III European Congress of Psychology, July 4-9, Tampere, Finland*.

Gazendam, Henk W.M., René J.Jorna, and Kenneth R. Blochowiak

1991 "The Mind Metaphor for Decision Support Systems." In: A.A.Verrijn-Stuart, H.G.Sol, and P.Hammersley (eds.). *Support Functionality in the Office Environment.* Amsterdam: Elsevier (North -Holland) / IFIP: 203-223.

Goldberg, Adele, and David Robson

1983 *Smalltalk-80: The language and its implementation.* Reading, MA: Addison-Wesley.

1989 *Smalltalk-80: The language.* Reading, MA: Addison-Wesley.

Kreutzer, W.

1986 *System simulation: Programming styles and languages.* Sydney: Addison-Wesley.

Newell, A.

1990 *Unified Theories of Cognition: The William James Lectures, 1987.* Cambridge, MA: Harvard University Press.

Peirce, Charles Sanders

1958 *Selected Writings: Values in a Universe of Chance*. Philip P. Wiener (ed.). New York: Dover, 1958: 113-136.

Piersol, K.W.,

1985 *HUMBLE V.2.0. Reference Manual.* Pasadena: Xerox Special Information Systems.

Rutges, Ap C., and Rob Vens.

1991 "A Beehive Simulator for Teaching and Research." *Paper presented at the WACRA Conference, Berlin, June0-July 3, 1991.*