Proceedings of the 2006 Winter Simulation Conference
L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds.

TUTORIAL ON AGENT-BASED MODELING AND SIMULATION PART 2: HOW TO MODEL WITH AGENTS

Charles M. Macal
Michael J. North

Center for Complex Adaptive Agent Systems Simulation (CAS^2)
Decision & Information Sciences Division
Argonne National Laboratory
Argonne, IL 60439, U.S.A.

ABSTRACT

Agent-based modeling and simulation (ABMS) is a new approach to modeling systems comprised of interacting autonomous agents. ABMS promises to have far-reaching effects on the way that businesses use computers to support decision-making and researchers use electronic laboratories to do research. Some have gone so far as to contend that ABMS is a new way of doing science. Computational advances make possible a growing number of agent-based applications across many fields. Applications range from modeling agent behavior in the stock market and supply chains, to predicting the spread of epidemics and the threat of bio-warfare, from modeling the growth and decline of ancient civilizations to modeling the complexities of the human immune system, and many more. This tutorial describes the foundations of ABMS, identifies ABMS toolkits and development methods illustrated through a supply chain example, and provides thoughts on the appropriate contexts for ABMS versus conventional modeling techniques.

1 INTRODUCTION

Agent-based Modeling and Simulation (ABMS) is a new modeling paradigm and is one of the most exciting practical developments in modeling since the invention of relational databases (North and Macal, in press). ABMS promises to have far-reaching effects on the way that businesses use computers to support decision-making and researchers use electronic laboratories to support their research. The goals of this tutorial are to show how ABMS is:

- Useful: Why ABMS is a good and even better modeling approach in many cases,
- Usable: How we are progressively advancing to usable ABMS systems, with better software development environments and more application experiences, and
- Used: How ABMS is being used to solve practical problems.

This tutorial is organized into two parts. The first part is a tutorial on how to think about ABMS. The background on ABMS and its motivating principles are described to illustrate its main concepts and to indicate the state-of-the-art. The second part is a tutorial on how to do ABMS. Practical applications of ABMS are described, ABMS toolkits are presented, and ABMS development approaches are discussed. Several ABMS examples are demonstrated throughout the tutorial.

2 HOW TO THINK ABOUT ABMS

2.1 What Is An Agent?

Although there is no universal agreement on the precise definition of the term "agent," definitions tend to agree on more points than they disagree. Some modelers consider any type of independent component (software, model, individual, etc.) to be an agent (Bonabeau 2001); an independent component's behavior can range from primitive reactive decision rules to complex adaptive artificial intelligence (AI). Others insist that a component's behavior must be adaptive in order for it to be considered an agent; the agent label is reserved for components that can in some sense learn from their environments and change their behaviors in response. Casti (1997) argues that agents should contain both base-level rules for behavior as well as a higher-level set of "rules to change the rules." The base-level rules provide responses to the environment while the "rules to change the rules" provide adaptation. Jennings (2000) provides a computer science view of agency emphasizing the essential characteristic of *autonomous* behavior. The fundamental feature of an agent is the capability of the component to make independent decisions. This requires agents to be active rather than purely passive.

From a practical modeling standpoint, we consider agents to have certain characteristics (Figure 1):

- An agent is identifiable, a discrete individual with a set of characteristics and rules governing its behaviors and decision-making capability. Agents are self-contained. The discreteness requirement implies that an agent has a boundary and one can easily determine whether something is part of an agent, is not part of an agent, or is a shared characteristic.
- An agent is situated, living in an environment with which it interacts along with other agents. Agents have protocols for interaction with other agents, such as for communication, and the capability to respond to the environment. Agents have the ability to recognize and distinguish the traits of other agents.
- An agent may be goal-directed, having goals to achieve (not necessarily objectives to maximize) with respect to its behaviors. This allows an agent to compare the outcome of its behavior relative to its goals.
- An agent is autonomous and self-directed. An agent can function independently in its environment and in its dealings with other agents, at least over a limited range of situations that are of interest.
- An agent is flexible, having the ability to learn and adapt its behaviors based on experience. This requires some form of memory. An agent may have rules that modify its rules of behavior.

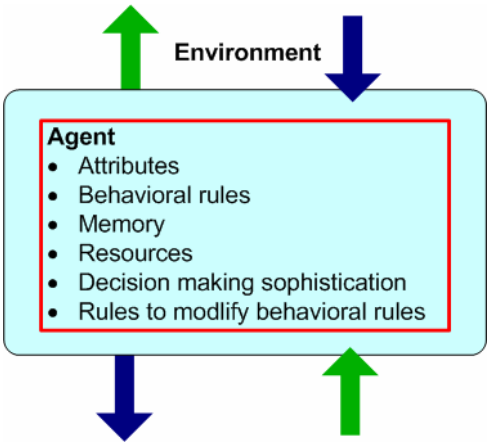


Figure 1: An Agent

Unlike particle systems (idealized gas particles for example) which are the subject of the field of particle simulation, agents are diverse, heterogeneous, and dynamic in their attributes and behavioral rules. Behavioral rules vary in their sophistication, how much information is considered in the agent decisions (cognitive "load"), the agent's internal models of the external world including other agents, and the extent of memory of past events the agent retains and uses in its decisions. Agents also vary by their attributes and accumulated resources.

Agent-based modeling is known by many names. ABM (agent-based modeling), ABS (agent-based systems), and IBM (individual-based modeling) are all widely-used acronyms, but "ABMS" will be used throughout this discussion. The term "agent" has connotations other than ABMS as well. ABMS agents are different from the typical agents found in mobile agent systems. "Mobile agents" are light-weight software proxies that roam over the world-wide web and perform various functions for users and to some extent can behave autonomously.

ABMS has strong roots in the fields of multi-agent systems (MAS) and robotics from the field of AI. But ABMS is not only tied to designing and understanding "artificial" agents. Its main roots are in modeling human social and organizational behavior and individual decision-making (Bonabeau 2001). With this, comes the need to represent social interaction, collaboration, group behavior, and the emergence of higher order social structure.

2.2 The Need for Agent Based Modeling

Why is agent-based modeling becoming so widespread? The answer is because we live in an increasingly complex world. First, the systems that we need to analyze and model are becoming more complex in terms of their interdependencies. Traditional modeling tools are no longer as applicable as they once were. An example application area is the deregulation of the electric power industry, as described in Section 3. Second, some systems have always been too complex for us to adequately model. Modeling economic markets has traditionally relied on the notions of perfect markets, homogeneous agents, and long-run equilibrium because these assumptions made the problems analytically and computationally tractable. We are beginning to be able to take a more realistic view of these economic systems through ABMS, as described below. Third, data are becoming organized into databases at finer levels of granularity. Micro-data can now support micro-simulations. And fourth, but most importantly, computational power is advancing rapidly. We can now compute large-scale micro-simulation models that would not have been plausible just a couple of years ago.

2.3 Background on ABMS

ABMS has connections to many other fields including complexity science, systems science, Systems Dynamics, computer science, management science, the social sciences in general, and traditional modeling and simulation. ABMS draws on these fields for its theoretical foundations, its

conceptual world view and philosophy, and for applicable modeling techniques. ABMS has its direct historical roots in complex adaptive systems (CAS) and the underlying notion that "systems are built from the ground-up," in contrast to the top-down systems view taken by Systems Dynamics. CAS concerns itself with the question of how complex behaviors arise in nature among myopic, autonomous agents. In addition, ABMS tends to be descriptive, with the intent of modeling the actual or plausible behavior of individuals, rather than normative such as traditional operations research (OR), which seeks to optimize and identify optimal behaviors.

The field of CAS was originally motivated by investigations into *adaptation* and *emergence* of biological systems. CAS have the ability to self-organize and dynamically reorganize their components in ways better suited to survive and excel in their environments, and this adaptive ability occurs, remarkably, over an enormous range of scales. John Holland, a pioneer in the field, identifies properties and mechanisms common to all CAS (Holland 1995) such as (1) Aggregation: allows groups to form, (2) Nonlinearity: invalidates simple extrapolation, (3) Flows: allow the transfer and transformation of resources and information, and (4) Diversity: allows agents to behave differently from one another and often leads to the system property of robustness. CAS mechanisms are: (1) Tagging: allows agents to be named and recognized, (2) Internal models: allows agents to reason about their worlds, and (3) Building blocks: allows components and whole systems to be composed of many levels of simpler components. These CAS properties and mechanisms provide a useful reference for designing agent-based models.

2.3.1 Simple Rules Result in Emergent Organization and Complex Behaviors

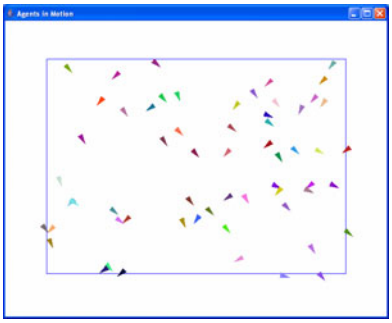
The Boids simulation is a good example of how interacting agents, characterized by simple behavioral rules, lead to emergent and seemingly organized behavior at the system level (Reynolds 2006). Agent behavior is reminiscent of schooling or flocking behavior in fish or birds. In the Boids model, each agent has three rules governing its movement:

1. Cohesion: each agent steers toward the average position of its nearby "flockmates,"
2. Separation: each agent steers to avoid crowding local flockmates, and
3. Alignment: each agent steers towards the average heading of local flockmates.

Here, nearby or local refers to agents in the immediate neighborhood of an agent as defined by some distance measure. Even with only these three simple rules applied at the individual agent level and only to the agents in its "neighborhood", the agents' behavior begins to appear coordinated, and a leaderless flock emerges (Figure 2).

Two observations are important about the Boids rules: (1) the rules are simple, and (2) the rules use only local information. We can make some observations from the Boids model that have implications for practical ABMS: (1) sustainable patterns can emerge in systems that are completely described by simple deterministic rules based on only local information, and (2) patterns that develop can be extremely sensitive to the initial conditions. Based on simple rules of behavior and agent interaction, natural systems seemingly exhibit collective intelligence, or *swarm intelligence*, even without the existence of or the direction provided by a central authority.

Natural systems are able to not only survive, but also to adapt and become better suited to their environment, effectively optimizing their behavior over time. How is it that an ant colony can organize itself to carry out the complex tasks of food gathering and nest building and at the same time exhibit an enormous degree of resilience if the colony is seriously disrupted? Swarm intelligence has inspired practical optimization techniques, such as ant colony optimization that have been used to solve practical scheduling and routing problems (Bonabeau et al. 1999).



(a) Initial Random Configuration

(b) After 500 Updates

Figure 2: Boids Simulation

2.3.2 Agent-Based Modeling in the Sciences

In applications of ABMS to social processes, agents represent people or groups of people, and agent relationships

represent processes of social interaction (Gilbert and Troitzsch 1999). The fundamental assumption is that people and their social interactions can be credibly modeled at some reasonable level of abstraction for at least specific and well-defined purposes, if not in general. This limited scope for representing agent behaviors in ABMS contrasts with the more general goals of AI. From an ABMS perspective, some important questions become immediately apparent: (1) How much do we know about credibly modeling people's behavior?, and (2) How much do we know about modeling human social interaction? These two questions have spawned and to some extent reinvigorated basic research programs in the social sciences that have the promise of informing ABMS on theory and methods for agent representation and behavior.

Thomas Schelling is credited with developing the first social agent-based simulation in which agents represent people and agent interactions represent a socially relevant process (Schelling 1978). Schelling applied cellular automata to study housing segregation patterns and posed the question, "is it possible to get highly segregated settlement patterns even if most individuals are, in fact, color-blind?" The Schelling model demonstrated that ghettos can develop spontaneously. Interpreted more generally, Schelling showed that *patterns can emerge that are not necessarily implied or even consistent with the objectives of the individual agents*.

Extending the notion of modeling people to growing entire artificial societies through agent simulation was taken up by Epstein and Axtell in their groundbreaking Sugarscape model (Epstein and Axtell 1996). In numerous computational experiments, Sugarscape agents emerged with a variety of characteristics and behaviors, highly suggestive of a realistic, although rudimentary and abstract, society. Emergent processes were observed including death, disease, trade, wealth, sex and reproduction, culture, conflict and war, and externalities such as pollution.

Economics is experiencing a paradigm shift in response to agent-based modeling. Some of the classical assumptions of standard micro-economic theory are: (1) Economic agents are rational, which implies that agents have well-defined objectives and are able to optimize their behavior (the basis for the "rational agent" model used in economics and many other social science disciplines), (2) Agents are homogeneous, having identical characteristics and rules of behavior, (3) There are decreasing returns to scale from economic processes, decreasing marginal utility, decreasing marginal productivity, etc., and (4) The long-run equilibrium state of the system is the primary information of interest. Each of these assumptions is relaxed in ABMS applications to economic systems. First, do organizations and individuals really optimize? Herbert Simon, a Nobel Laureate who pioneered the field of AI, developed the notion of "satisficing" to describe what he observed people and organizations doing in the real world (Simon 2001). Behavioral economics is a relatively new field that incorporates experimental findings on psychology and cognitive aspects of agent decision making to determine people's actual economic and decision making behavior. Second, that agent diversity universally occurs in the real-world is a key observation of complexity science. Many natural organizations from ecologies to industries are characterized by populations whose diversity gives rise to its stability and robustness. Third, "positive feedback loops" and "increasing returns" have been identified as underlying dynamic processes of rapid exponential growth in economic systems (Arthur et al. 1997). Positive feedback can create self-sustaining processes that quickly take a system away from its starting point to a faraway state. Fourth, long-run equilibrium states are not the only results of interest. The transient states that are encountered along the way to a long-run state are often of interest. Furthermore, not all systems come to an equilibrium (Axtell 2000). The field of Agent-based Computational Economics (ACE) has grown up around the application of ABMS to economic systems (Tesfatsion 2002, 2005).

Anthropologists are also developing large-scale agent-based simulations of ancient civilizations to help explain their growth and decline, based on archaeological data. ABMS has been applied to help understand the social and cultural factors responsible for the disappearance of the Anasazi in the southwestern U.S. (Koehler et al. 2005).

Sociologists are doing agent-based modeling as well. Macy and Willer (2002) conclude that agent-modeling is a promising basis for modeling social life as interactions among adaptive agents who influence one another in response to the influences they receive. Cognitive science has had its own notion of agency. Cognitive scientists are developing agent-based models of emotion, cognition, and social behavior based on the notion that a person's emotional state impacts their behavior as well as their social interactions. The goal is to create synthetic agents who embody the nuanced interplay between emotion, cognition and social behavior. Computational social science is becoming a subfield in the social sciences (Sallach and Macal 2001).

2.3.3 Topologies as a Basis for Social Interaction

As much as modeling agent behaviors, agent modeling concerns itself with modeling agent *iteractions*. The primary issues of modeling agent interaction are who is connected to who and the mechanisms governing the nature of the interactions. Cellular automata represent agent interaction patterns and available local information by using a grid or lattice and the cells immediately surrounding an agent as the neighborhood. Other agent interaction topologies, such as networks, allow an agent's neighborhood to be defined more generally and may more accurately describe social agents' interaction patterns.

Social Network Analysis (SNA) is a field with a long history that studies the characterization and analysis of social structure and interaction through network representations. Traditionally, SNA has focused on static networks, i.e., networks that do not change their structure over time or as a result of agent behavior. Recently, much progress has been made in understanding the processes of growth and change of real-world networks (Barabási 2002). Dynamic network analysis (DNA) is a new field that incorporates the mechanisms of network growth and change based on agent interaction processes. Understanding the agent rules that govern how networks are structured and grow, how quickly information is communicated through networks, and the kinds of relationships that networks embody are important aspects of "network ABMS."

2.3.4 Modeling Agent Processes

Identifying the social interaction mechanisms for how cooperative behavior emerges among individuals and groups is an interesting question with practical implications. Evolutionary Game Theory is related to traditional game theory and takes into account the repeated interactions of the players and their effect on strategies. Axelrod has shown that a simple Tit-For-Tat strategy of reciprocal behavior toward individuals is enough to establish sustainable cooperative behavior (Axelrod 1997). The broader need is for a generative type of social science in which the processes from which social structure emerges can be understood as the necessary result of social interactions (Epstein 2005, Sallach 2003).

3 ABMS APPLICATIONS

Practical agent-based modeling and simulation is actively being applied in many areas (Table 1). ABS applications range from modeling agent behavior in the stock market (LeBaron 2002) and supply chains (Fang et al. 2002), to predicting the spread of epidemics (Huang et al. 2004) and the threat of bio-warfare (Carley 2006), from modeling the growth and decline of ancient civilizations (Kohler et a. 2005) to modeling the complexities of the human immune system (Folcik and Orosz 2006), and many other areas.

ABMS applications range across a continuum, from small, elegant, minimalist models to large-scale decision support systems. Minimalist models are based on a set of idealized assumptions, designed to capture only the most salient features of a system. These are exploratory electronic laboratories in which a wide range of assumptions can be varied over a large number of simulations. Decision support models tend to be large-scale applications, designed to answer a broad range of real-world policy questions. These models are distinguished by including real data and having passed some degree of validation testing to establish credibility in their results.

Table 1: Agent-based Modeling Applications

Business and Organizations • Manufacturing Operations • Supply chains • Consumer markets • Insurance industry Economics • Artificial financial markets • Trade networks Infrastructure • Electric power markets • Transportation • Hydrogen infrastructure Crowds • Pedestrian movement • Evacuation modeling	Society and Culture • Ancient civilizations • Civil disobedience • Social determinants of terrorism • Organizational networks Military • Command & control • Force-on-force Biology • Population dynamics • Ecological networks • Animal group behavior • Cell behavior and sub cellular processes

3.1 A Real-World ABMS Example

An agent simulation of emerging deregulated electric power markets illustrates the use of agent modeling as a decision support tool. The EMCAS (Electricity Market Complex Adaptive System) model is a large-scale agent-based simulation model of the electric power market designed to investigate market restructuring and deregulation. EMCAS has been used to understand the implications of the coming competitive market in Illinois on electricity prices, availability, and reliability (Cirillo et al. 2006), and these results have been entered into the public record of the Illinois Commerce Commission. EMCAS is an example of an agent-based model that has been successfully applied to a real-world policy issue and provided information that would otherwise have not been available using any other modeling approach.

EMCAS is described elsewhere from various perspectives including the benefits of agent-based modeling for deregulated electric power markets (Koritarov 2004). The agents in EMCAS represent the participants in the restructured electricity market (Figure 3). Different types of agents capture the heterogeneity of restructured markets, including generation companies, demand companies, transmission companies, distribution companies, independent system operators, and consumers. The agents perform diverse tasks using specialized decision rules. For example, generation company agents learn about the market response to their price-quantity bids into a simulated day-ahead market for generating electric power, infer the strategies of their competitors, and adapt their actions accordingly. They engage in price discovery and learn how they can influence the market through their actions to increase their utility, defined as a combination of profits and market share. EMCAS agents reside in the "business layer" and are constrained in their behaviors by the physics of the

electric power grid which is implemented in the "infrastructure layer," which represents a regional transmission network at the individual node (bus) level.

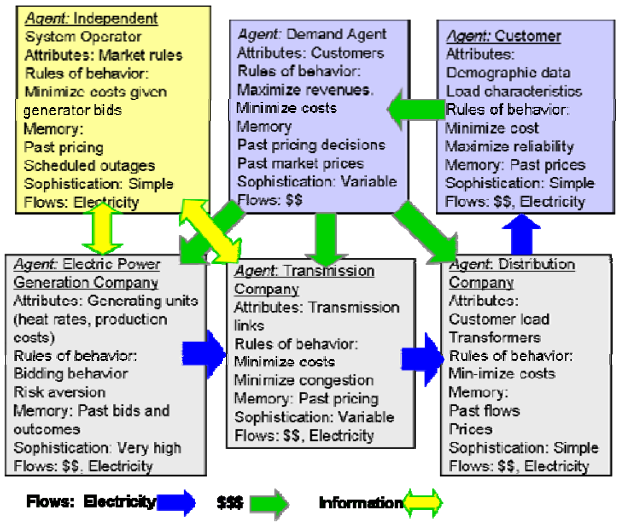


Figure 3: Agents in EMCAS Electric Power Market Model

4 HOW TO DO ABMS

At a general level, one goes about building an agent-based model in much the same way as any other type of model. First, identify the purpose of the model, the questions the model is intended to answer and engage the potential users in the process. Next, systematically analyze the system under study, identifying components and component interactions, relevant data sources, and so on. Then, apply the model and conduct a series of "what-if" experiments by systematically varying parameters and assumptions. Finally, understand the robustness of the model and its results by using sensitivity analysis and other techniques. These general steps of model building apply to agent-based modeling as well. See (Law and Kelton 2000) for an excellent description of good simulation model building practice.

4.1 How to Build an Agent-based Simulation Model

Agent-based modeling brings with it a few unique aspects owing to the fact that ABMS takes the agent perspective, first and foremost, in contrast to the process-based perspective that is the traditional hallmark of simulation modeling. In addition to the standard model building tasks, practical ABMS requires one to: (1) identify the agents and get a theory of agent behavior, (2) identify the agent relationships and get a theory of agent interaction, (3) get the requisite agent-related data, (4) validate the agent behavior models in addition to the model as a whole, and (5) run the model and analyze the output from the standpoint of linking the micro-scale behaviors of the agents to the macro-scale behaviors of the system.

ABMS is more often than not a stochastic modeling approach. A model generally includes stochastic elements to model the range of outcomes for agent behaviors and interactions which are not known with certainty.

Agent-based modeling does not as of yet have a mature set of standard formalisms or procedures for model development and agent representation such as those that are part of Systems Dynamics modeling. Other than the implemented software code, there is no scheme for unambiguously representing an agent-based model. However, agent modeling documentation schemes along these lines have recently been proposed with the intent of promoting agent model transferability and reproducibility (Grimm et al. 2006). Agent-based modeling can benefit from the use of the Unified Modeling Language (UML) for representing models. UML is a visual modeling language for representing object-oriented (O-O) systems (Booch, Rumbaugh et al. 1998) that is commonly adopted to support agent-based models in both the design and communication phases. UML consists of a number of high-structured types of diagrams and graphical elements that are assembled in various ways to represent a model. The UML representation is at a high level of abstraction, independent of the model's implementation in the particular O-O programming language used.

Most large-scale agent-based modeling toolkits that provide basic agent functionality are based on the object-oriented paradigm. Agent-based simulation is not the same as object-oriented simulation, but the O-O modeling paradigm is a useful basis for agent modeling, since an agent can be considered a self-directed object with the capability to autonomously choose actions based on the agent's situation. The O-O paradigm is natural for agent modeling, with its use of object classes as agent templates and object methods to represent agent behaviors. O-O modeling takes a data-driven rather than a process-driven perspective. One way to begin the modeling process is to define abstract data types and objects. This is the methodology used in the supply chain example below.

The general steps in building an agent model are as follows:

1. *Agents:* Identify the agent types and other objects (classes) along with their attributes.
2. *Environment:* Define the environment the agents will live in and interact with.
3. *Agent Methods:* Specify the methods by which agent attributes are updated in response to either agent-to-agent interactions or agent interactions with the environment.
4. *Agent Interactions:* Add the methods that control which agents interact, when they interact, and how they interact during the simulation.

5. *Implementation:* Implement the agent model in computational software.

4.1.1 Discovering Agents

Identifying agents, accurately specifying their behaviors, and appropriately representing agent interactions are the keys to developing useful agent models. Agents are generally the decision-makers in a system. These include traditional decision-makers, such as managers, as well as non-traditional decision-makers, such as complex computer systems that have their own behaviors.

How can agent behaviors be discovered? First, one needs a theory of agent behavior. One may begin with a normative model in which agents attempt to optimize and use this model as a starting point for developing a simpler and more descriptive heuristic model of behavior. One may also begin with a behavioral model if applicable behavioral theory is available. For example, numerous theories abound for modeling consumer shopping behavior based on empirical studies. Alternatively, a number of formal logic frameworks have been developed in order to reason about agents, such as the BDI (Belief-Desire-Intent) model, and these can serve as a basis for agent models.

Knowledge engineering and participatory simulation are also useful techniques to employ. Knowledge engineering consists of a collection of techniques for eliciting and organizing the knowledge of experts while accounting for reporting errors and situational biases. Participatory ABMS combines the agent paradigm with ideas from organization theory to specify goal-driven simulations that consist entirely of human participants playing roles, akin to gaming, but with much more structure.

4.1.2 The ABMS Modeling Lifecycle

Developing an agent-based simulation is part of the more general model software development process. The development timeline typically has several highly interleaved stages. The concept development and articulation stage defines the project goals. The requirements definition stage makes the goals specific. The design stage defines the model structure and function. The implementation stage builds the model using the design. The operationalization stage puts the model into use. In practice, successful ABMS projects typically iterate over these stages several times with more detailed models resulting from each iteration.

Agent modeling can be done in the small, on the desktop, or in the large, using large-scale cluster computers, or at any scale in-between. Successful projects also begin small using one or more of the desktop ABMS tools and then grow into the larger-scale ABMS toolkits in stages. Desktop agent-based models can be simple, designed and developed in a period of a few days by a single computer-literate modeler using tools learned in a few days or weeks. Desktop ABMS can be used to learn how to do agent modeling, test agent modeling design concepts and perform many types of serious modeling and analysis. Desktop tools include general spreadsheets and computational mathematics systems such as MATLAB or Mathematica.

Large-scale ABMS extends agent modeling beyond simple desktop environments and allows thousands to millions of agents to engage in sophisticated behaviors and interchanges. Large-scale agent modeling is usually done with computer-based agent simulation environments that support features specific to agent modeling. Several standards for agent software have influenced agent-based toolkit development including the Foundation for Intelligent Physical Agents' (FIPA 2005) specifications. Features include the availability of a time scheduler, agent communication mechanisms, flexible interaction topologies, a range of architectural choices, facilities for storing and displaying agent states, large-scale development support, and in some cases special topic support. Large-scale agent models generally require more advanced skills and development resources than desktop environments.

4.1.3 ABMS Modeling Toolkits

Thanks to substantial public research and development investments, many ABMS software environments are now freely available. These include Repast (North et al. 2006), Swarm (SDG 2006; Minar et al. 1996), NetLogo (NetLogo 2006) and MASON (GMU 2006) among many others. Proprietary toolkits are also available such as AnyLogic (2006). A recent review and comparison of Java-based agent modeling toolkits is provided by Tobias and Hoffman (2004).

Swarm was the first ABMS software development environment launched in 1994 at the Santa Fe Institute. Following the original Swarm innovation, the Repast (REcursive Porous Agent Simulation Toolkit) toolkit illustrates the state-of-the-art in agent-based modeling toolkits (North et al. 2006). Repast has been used extensively in social simulation applications (North and Macal 2005). Figure 4 illustrates a Repast social network application.

Repast is the leading free and open source large-scale agent-based modeling and simulation library. Users build simulations by incorporating Repast library components into their own programs or by using the visual scripting environments. There are three production versions of Repast, namely Repast for Python (Repast Py), Repast for Java (Repast J), and Repast for the Microsoft .NET framework (Repast .NET). Repast Py is a cross-platform visual model construction system that allows users to build models using a graphical user interface and write agent behaviors using Python scripting. Repast J is a pure Java modeling environment to support the development of large-scale agent models. It includes a variety of features such as a

fully concurrent discrete event scheduler, a model visualization environment, integration with geographical information systems for modeling agents on real maps, and adaptive behavioral tools such as neural networks and genetic algorithms. Repast .NET is a pure C# modeling environment that brings all of the features of Repast J to the Microsoft .NET framework.

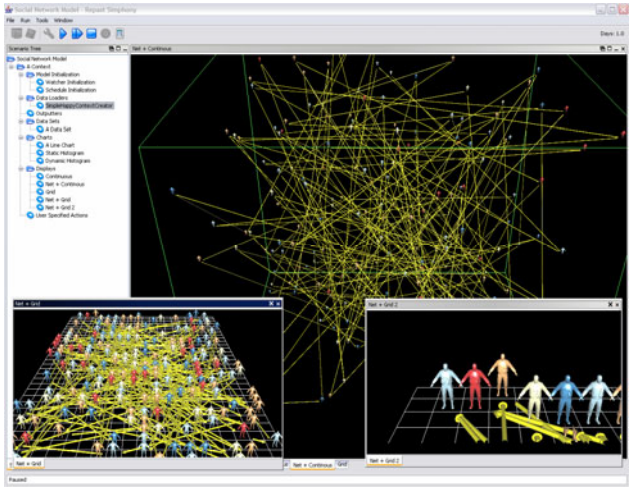


Figure 4: Agents Interact Over a Self-Organizing Social Network in a Repast Simulation of Social Influence

4.2 A Supply Chain Example

An agent-based model of a supply chain illustrates the general steps in building a simple agent model. A supply chain consists of five stages: factories, distributors, wholesalers, and retailers who respond to customers' demand. Multiple agents of each type exist at each stage forming a network of supply chain agents (Figure 5).

We make various simplifying assumptions for this example such as we ignore suppliers, there is only one commodity, no transformation of goods is made, and no assembly of materials into products is required. The flows of goods and information in the form of orders between stages (agents) as well as physical shipments are included in the model. The flows of payments and the complexities of pricing, negotiation, and financial accounting that this would entail are not included in this simple model but could easily be added.

Each period, supply chain agents execute behaviors (Figure 6):

1. The customer places an order with the retailer.
2. The retailer fills the order immediately from its respective inventory if it has enough inventory in stock (if the retailer runs out of stock, the customer's order is placed on backorder and filled when stock is replenished).
3. The retailer receives a shipment from the upstream wholesaler in response to previous orders. The retailer then decides how much to order from the wholesaler based on an "ordering rule." The ordering decision is based in part on how much the retailer expects customer demand will be in the future. The retailer estimates future customer demand using a "demand forecasting" rule. The retailer then orders items from the wholesaler to cover expected demand and any shortages relative to explicit inventory or pipeline goals.
4. Similarly, each wholesaler receives a shipment from the upstream distributor, forecasts future demand by the downstream retailer, and places an order with the distributor. This process continues up the chain to the factory who decides on how much to put into new production.

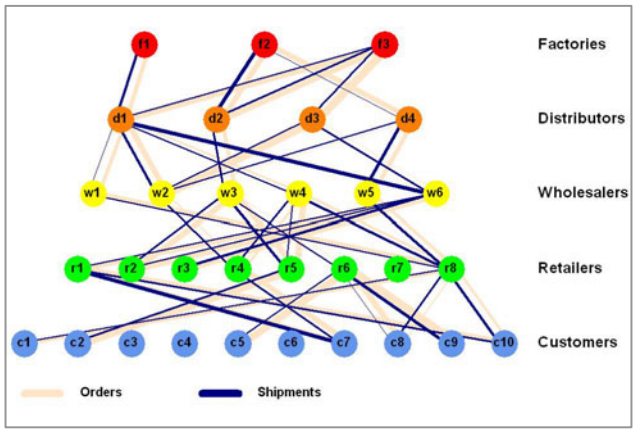


Figure 5: Typical Supply Chain Network and Agents

The goal of the supply chain agents, other than customers, is to manage their inventory in such a way as to minimize their costs through judicious decisions based on how much to order each period. When inventories are too low and there is a danger of running out of stock, agents order more; when inventories are too large and agents incur high inventory holding costs, agents order less. Besides the inventory holding charge, agents incur a backorder charge when they receive an order and cannot immediately fill it because they are out of stock. Each agent strikes a delicate balance between having too much inventory, which runs up inventory holding costs, and too little inventory, which puts the agent at a greater risk of running out of stock and incurring excessive backorder charges.

In this supply chain example, agents only have access to local information. No agent has a global view of the supply chain or is in a position to optimize the system as a whole. Agents adopt decision rules that only consider this local information in making their decisions.

This simple agent-based model is a useful foundation for more realistic models of supply chains in which the various simplifying assumptions have been relaxed and

more complex agent decision rules are considered. Several agent models of supply chains have been developed with various enhancements such as agent access to non-local information and to modeling the endogenous development of agent relationships based on trust (Macal and North 2003; Macal 2004).

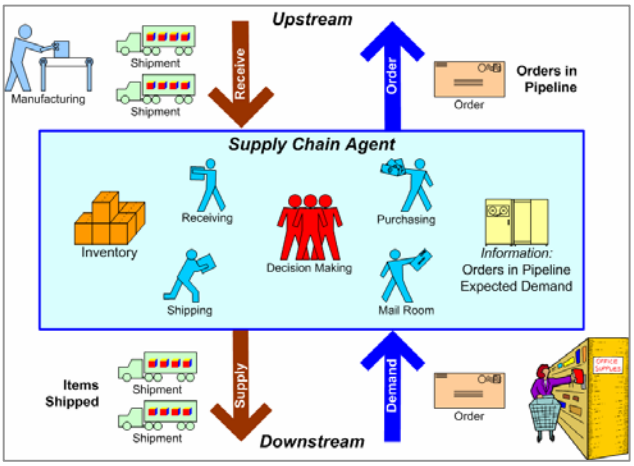


Figure 6: World of the Supply Chain Wholesaler Agent

4.3 Building the Supply Chain Agent-based Model

One begins developing an ABS model by identifying the agent types and other objects (classes) along with their attributes. (Step 1). In the supply chain model, the supply chain agents consist of factory, distributor, wholesaler, retailer and customer agents. Everything in the simulation is either an agent or an object; other objects include the clock and the set of output reports. The distributor, wholesaler, and retailer agents are grouped together in a class called "middleAgents" because they all have the same structure in terms of their attributes and the methods.

Each agent class is represented by a set of attributes and methods that operate on the agent class. A UML class diagram is a convenient way of representing the agents of the supply chain model (Figure 7). For example, the factory agent is represented by the following attributes: the agent's name; inventory level; desired inventory level; amount in pipeline; desired amount in pipeline; the amounts received, shipped, ordered, and demanded; various decision parameters; and the costs incurred of holding inventory or backorders. The values of these variables at any point in time constitute the agent state.

One then specifies the environment in which the agents live and interact (Step 2). For the supply chain model, the environment consists of external (non-agent) factors that influence agent behavior. For example, an environment variable could be the labor rate and its dependence on geographic locale, which could also be included as an agent attribute.

Next, one specifies the methods by which agent attributes are updated during the simulation in response to either agent-to-agent interactions or agent interactions with the environment (Step 3). For example, in the supply chain model, the inventory level is an attribute of each agent. Inventory is updated when orders and shipments are received and sent. For example, agent methods that embody processing of orders and shipments include: *arriveOrder()*, *sendOrder()*, *arriveShipment()*, and *sendShipment()*. These methods would be applied to the agents upon receipt of an order or shipment and affect the values of agent attributes. The factory class also has methods that more directly embody the agent's behavioral decision rules. These include a rule for determining how much to order and from whom at any point in time, embodied in the procedure *orderRule()*, and a rule for forecasting demand, embodied in the procedure *forecastRule(),* the details of which are not shown here.

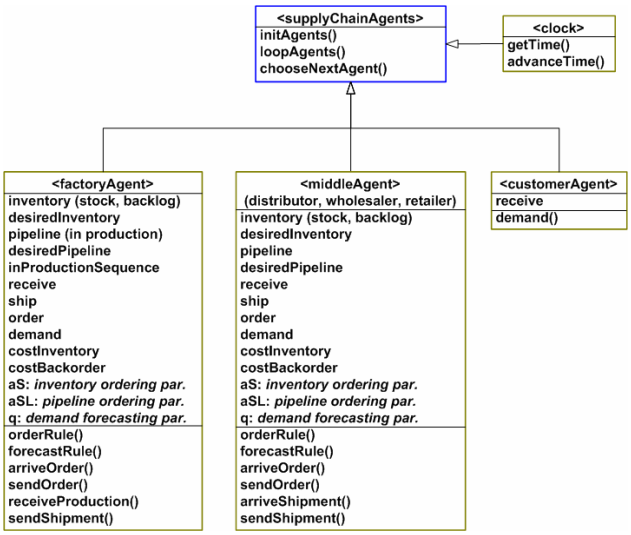


Figure 7: Supply Chain Agent UML Class Diagram

One then adds the methods that control which agents interact, when they interact, and how they interact (Step 4). For example, one may develop a procedure for selecting which agents to interact with based on a bidding process in which, for example, the least-costly factory agent is selected from among all factory agents by a distribution agent placing an order. The *agent selection procedure* could be invoked at every time period or when inventory levels reach specified thresholds. The *agent interaction procedure* would consist of placing an order with the selected agent at the determined time.

In addition to agents, the supply chain model consists of agent relations. If agents are the nodes in the supply chain, agent relations are the links or edges in the network. As such, each agent relation involves two agents. For example, the factory-distributor relation includes the attributes of the number of items in-transit from factory to dis-

tributor and the order in-transit from distributor to factory. Agent relations also have methods that operate on them just as agents have. For example, *getShipments(), getOrders(), getUpstreamAgent()*, and *getDownstreamAgent()* are useful methods for agent relations.

The complete set of object class definitions and methods, parameter values, and initial values for all the agent and other object states constitutes a complete specification of an agent model.

One implements an agent model by either writing an object-oriented program using, for example, the Java or C++ programming languages, or using a higher-level agent-based toolkit, as discussed previously (Step 5). The toolkit provides an extensive set of classes that encapsulate the basic functionality required by the agent models. For example, the functionality for the sequence of agent operations and interactions in the supply chain model and the control mechanisms that cause each of the agent methods to be invoked at the proper time or in the proper situation would be part of the functionality provided by the scheduler class of an agent-based toolkit.

5 WHY AND WHEN ABMS

Situations for which agent-based modeling can offer distinct advantages to conventional simulation approaches, reveal new insights and answer long-standing questions are becoming better understood every day. When is it beneficial to think in terms of agents?

- When there is a natural representation as agents
- When there are decisions and behaviors that can be defined discretely (with boundaries)
- When it is important that agents adapt and change their behaviors
- When it is important that agents learn and engage in dynamic strategic behaviors
- When it is important that agents have a dynamic relationships with other agents, and agent relationships form and dissolve
- When it is important that agents form organizations, and adaptation and learning are important at the organization level
- When it is important that agents have a spatial component to their behaviors and interactions
- When the past is no predictor of the future
- When scaling-up to arbitrary levels is important
- When process structural change needs to be a result of the model, rather than a model input

ACKNOWLEDGMENTS

The authors thank Averill Law for recent discussions on the relationship between agent modeling and conventional simulation. This work is sponsored by the U.S. Department of Energy under contract W-31-109-ENG-38.

REFERENCES

AnyLogic. 2006. <http://www.xjtek.com/>.

Arthur, W. B., S. N. Durlauf, and D. A. Lane (eds.) 1997. *The economy as an evolving complex system II*, SFI Studies in the Sciences of Complexity, Addison Wesley: Reading, MA.

Axelrod, R. 1997. *The complexity of cooperation: agent-based models of competition and collaboration*, Princeton, NJ: Princeton University Press.

Axtell, R. 2000. *Why agents? On the varied motivations for agent computing in the social sciences,* Working Paper 17, Center on Social and Economic Dynamics, Brookings Institution, Washington, D.C.

Barabási, A.-L. 2002. *Linked: the new science of networks,* Cambridge, MA: Perseus Pub.

Bonabeau, E., M. Dorigo and G. Theraulaz. 1999. *Swarm intelligence: from natural to artificial systems*, Oxford: Oxford University Press.

Bonabeau, E. 2001. Agent-based modeling: methods and techniques for simulating human systems. In *Proc. National Academy of Sciences* 99(3): 7280-7287.

Booch, G., J. Rumbaugh and I. Jacobson. 1998. The Unified Modeling Language User Guide, Addison-Wesley:New York.

Carley, K. 2006. *BioDefense through City Level Multi-Agent Modeling of Bio and Chemical Threats*. Arizona Spring Biosurveillance Workshop, Tucson, Arizona.

Casti, J. 1997. *Would-be worlds: how simulation is changing the world of science*, New York: Wiley.

Cirillo, R., P. Thimmapuram, T. Veselka, V. Koritarov, G. Conzelmann, C. Macal, G. Boyd, M. North, T. Overbye and X. Cheng. 2006. *Evaluating the Potential Impact of Transmission Constraints on the Operation of a Competitive Electricity Market in Illinois*, Argonne National Laboratory, Argonne, IL, ANL-06/16 (report prepared for the Illinois Commerce Commission), April.

Epstein, J. M. and R. Axtell. 1996. *Growing artificial societies: social science from the bottom up*. Cambridge, MA: MIT Press.

Fang, C., S. O. Kimbrough, A. Valluri, and Z. Zheng. 2002. On Adaptive Emergence of Trust Behavior in the Game of Stag Hunt. *Group Decision and Negotiation,* 11(6): 449–467.

FIPA (Foundation for Intelligent Physical Agents). 2005. FIPA Home Page, <http://www.fipa.org/>.

Folcik, V. and C. G. Orosz. 2006. An Agent-based Model Demonstrates That the Immune System Behaves Like a Complex System and a Scale-Free Network. *SwarmFest 2006*, University of Notre Dame, South Bend, IN, June.

Gilbert, N. and K. G. Troitzsch. 1999. *Simulation for the Social Scientist*, Buckingham UK: Open University Press.

GMU (George Mason University). 2006. <http://cs.gmu.edu/~eclab/projects/mason/>.

Grimm, V., U. Berger, F. Bastiansen, S. Eliassen, V. Ginot, J. Giske, J. Goss-Custard, T. Grand, S. K. Heinz and G. Huse. 2006. A Standard Protocol for Describing Individual-Based and Agent-Based Models. *Ecological Modelling*. [In Press.]

Holland, J. H., 1995, *Hidden order: how adaptation builds complexity*. Reading, MA: Addison-Wesley.

Huang, C.-Y., C.-T. Sun, J.-L. Hsieh, and H. Lin. 2004. Simulating SARS: Small-world epidemiological modeling and public health policy assessments. *Journal of Artificial Societies and Social Simulation* 7(4): 100-131.

Jennings, N. R. 2000. On agent-based software engineering. *Artificial Intelligence,* 117:277-296.

Koritorov, V., 2004, Real-World Market Representation with Agents, *IEEE Power and Energy Magazine*:39-46.

Kohler, T. A., G. J. Gumerman and R. G. Reynolds. 2005. Simulating ancient societies. *Scientific American.* July.

Law, A. M. and D. W. Kelton. 2000. *Simulation modeling and analysis*. 3rd ed. New York: McGraw-Hill.

LeBaron, B. (2002). Short-memory traders and their impact on group learning in financial markets. *Proc. National Academy of Sciences* 99(90003): 7201-7206.

Minar, N., R. Burkhart, C. Langton, and M. Askenazi. 1996. The Swarm simulation system, a toolkit for building multi-agent simulations, <http://www.santafe.edu/projects/swarm/overview/overview.html>.

Macal, C., and M. North. 2003. Effects of global information availability in networks of supply chain agents. *Proc. Agent 2003: Conf. on Challenges in Social Simulation*, Eds., C. Macal, D. Sallach and M. North, Chicago, IL, Oct. 2-4, 235-252, Argonne National Laboratory.

Macal, C. 2004. Emergent structures from trust relationships in supply chains. *Proc. Agent 2004: Conf. on Social Dynamics,* Eds., C. Macal, D. Sallach and M. North, Chicago, IL, Oct. 7-9, 743-760, Argonne National Laboratory.

NetLogo. 2006. NetLogo home page. <http://ccl.northwestern.edu/netlogo>.

North, M. J., and C. M. Macal. [In press.] *Managing business complexity: discovering strategic solutions with agent-based modeling and simulation*, Oxford: Oxford University Press.

North, M. J., N. T. Collier, and J. R. Vos. 2006. Experiences in Creating Three Implementations of the Repast Agent Modeling Toolkit, *ACM Transactions on Modeling and Computer Simulation*, 16(1):1-25, January.

North, M. J. and C. M. Macal. 2005. Escaping the accidents of history: an overview of artificial life modeling with Repast, in *Artificial Life Models in Software*, Eds., A. Adamatzky and M. Komosinski, Springer-Verlag: Dordrecht, Netherlands.

NRC (National Research Council). 2003. *Dynamic social network modeling and analysis: workshop summary and papers,* R. Brieger, K. Carley, and P. Pattison, Committee on Human Factors, Washington, DC: National Academies Press.

Repast. 2006. Repast home page. <http://repast.sourceforge.net/>.

Reynolds, Craig. 2006. Boids. <http://www.red3d.com/cwr/boidss/>.

Sallach, D. and C. Macal. 2001. The simulation of social agents: an introduction. *Social Science Computer Review* 19(3):245–248.

Schelling, T. C. 1978. *Micromotives and macrobehavior.* New York: Norton.

SDG (Swarm Development Group). 2006. Swarm Development Group home page. <http://www.swarm.org>.

Simon, H. 2001. *The sciences of the artificial*, Cambridge, MA: MIT Press.

Tesfatsion, L. 2005. Agent-based Computational Economics (ACE) home page. <http://www.econ.iastate.edu/tesfatsi/ace.htm>.

Tobias, R. and C. Hofmann. 2004. Evaluation of free Java-libraries for social-scientific agent based simulation. *Journal of Artificial Societies and Social Simulation*. 7(1), Jan. 31.

AUTHOR BIOGRAPHIES

CHARLES M. MACAL, Ph.D., P.E., is the Director, Center for Complex Adaptive Agent Systems Simulation (CAS2), Argonne National Laboratory. He is a member of the INFORMS-Simulation Society, Society for Computer Simulation Int'l., the Systems Dynamics Society and a founding member of NAACSOS. Charles has a Ph.D. in Industrial Engineering & Management Sciences from Northwestern University. His e-mail is <macal@anl.gov>.

MICHAEL J. NORTH, M.B.A., Ph.D., is the Deputy Director of CAS2 at Argonne. Michael has over 15 years of experience developing advanced modeling and simulation applications for the federal government, international agencies, private industry, and academia. Michael has a Ph.D. in Computer Science from the Illinois Institute of Technology as well as degrees in business and mathematics. His e-mail is <north@anl.gov>.