Tutorial on agent-based modelling and simulation

Article *in* Journal of Simulation · September 2010

DOI: 10.1057/jos.2010.3 · Source: DBLP

2 authors:

C. M. Macal
Argonne National Laboratory
169 PUBLICATIONS **6,849** CITATIONS
SEE PROFILE

Michael John North
Argonne National Laboratory
130 PUBLICATIONS **7,838** CITATIONS
SEE PROFILE

Journal of Simulation (2010) 4, 151–162

www.palgrave-journals.com/jos/

Tutorial on agent-based modelling and simulation

CM Macal[1,2]* and MJ North[1,2]

[1]Center for Complex Adaptive Agent Systems Simulation, Decision & Information Sciences Division, Argonne National Laboratory, Argonne, Il, USA; and [2]Computation Institute, The University of Chicago, Chicago, Il, USA

Agent-based modelling and simulation (ABMS) is a relatively new approach to modelling systems composed of autonomous, interacting agents. Agent-based modelling is a way to model the dynamics of complex systems and complex adaptive systems. Such systems often self-organize themselves and create emergent order. Agent-based models also include models of behaviour (human or otherwise) and are used to observe the collective effects of agent behaviours and interactions. The development of agent modelling tools, the availability of micro-data, and advances in computation have made possible a growing number of agent-based applications across a variety of domains and disciplines. This article provides a brief introduction to ABMS, illustrates the main concepts and foundations, discusses some recent applications across a variety of disciplines, and identifies methods and toolkits for developing agent models.
Journal of Simulation (2010) **4,** 151–162. doi:10.1057/jos.2010.3

Keywords: agent-based modelling and simulation; modelling behaviour; social simulation

1. Introduction

Agent-based modelling and simulation (ABMS) is a relatively new approach to modelling complex systems composed of interacting, autonomous 'agents'. Agents have behaviours, often described by simple rules, and interactions with other agents, which in turn influence their behaviours. By modelling agents individually, the full effects of the diversity that exists among agents in their attributes and behaviours can be observed as it gives rise to the behaviour of the system as a whole. By modelling systems from the 'ground up'—agent-by-agent and interaction-by-interaction—self-organization can often be observed in such models. Patterns, structures, and behaviours emerge that were not explicitly programmed into the models, but arise through the agent interactions. The emphasis on modelling the heterogeneity of agents across a population and the emergence of self-organization are two of the distinguishing features of agent-based simulation as compared to other simulation techniques such as discrete-event simulation and system dynamics. Agent-based modelling offers a way to model social systems that are composed of agents who interact with and influence each other, learn from their experiences, and adapt their behaviours so they are better suited to their environment.

Applications of agent-based modelling span a broad range of areas and disciplines. Applications range from modelling agent behaviour in the stock market (Arthur *et al*, 1997) and supply chains (Macal, 2004a) to predicting the spread of epidemics (Bagni *et al*, 2002) and the threat of bio-warfare (Carley *et al*, 2006), from modelling the adaptive immune system (Folcik *et al*, 2007) to understanding consumer purchasing behaviour (North *et al*, 2009), from understanding the fall of ancient civilizations (Kohler *et al*, 2005) to modelling the engagement of forces on the battlefield (Moffat *et al*, 2006) or at sea (Hill *et al*, 2006), and many others. Some of these applications are small but elegant models, which include only the essential details of a system, and are aimed at developing insights into a social process or behaviour. Other agent-based models are large scale in nature, in which a system is modelled in great detail, meaning detailed data are used, the models have been validated, and the results are intended to inform policies and decision making. These applications have been made possible by advances in the development of specialized agent-based modelling software, new approaches to agent-based model development, the availability of data at increasing levels of granularity, and advancements in computer performance.

Several indicators of the growing interest in agent-based modelling include the number of conferences and workshops devoted entirely to or having tracks on agent-based modelling, the growing number of peer-reviewed publications in discipline-specific academic journals across a wide range of application areas as well as in modelling and simulation journals, the growing number of openings for people specializing in agent-based modelling, and interest on the part of funding agencies in supporting programmes that require agent-based models. For example, a perusal of the programme for a recent Winter Simulation Conference revealed that 27 papers had the word 'agent' in the title or abstract (see http://www.wintersim.org/pastprog.htm).

This article provides a brief introduction to ABMS. We illustrate the main concepts of agent-based modelling

*Correspondence: CM Macal, Center for Complex Adaptive Agent Systems Simulation, Decision & Information Sciences Division, Argonne National Laboratory, 9700 S. Cass Avenue, Argonne, IL 60439-4867, USA.
E-mail: macal@anl.gov

(Section 2), discuss some recent applications across a variety of disciplines (Section 3), and identify methods and toolkits for developing agent models (Section 4).

2. Agent-based modelling

2.1. Agent-based modelling and complexity

ABMS can be traced to investigations into complex systems (Weisbuch, 1991), complex adaptive systems (Kauffman, 1993; Holland, 1995), and artificial life (Langton, 1989), known as ALife (see Macal (2009) for a review of the influences of investigations into artificial life on the development of agent-based modelling and the article by Heath and Hill in this issue for a review of other early influences). Complex systems consist of interacting, autonomous components; complex adaptive systems have the additional capability for agents to adapt at the individual or population levels. These collective investigations into complex systems sought to identify universal principles of such systems, such as the basis for self-organization, emergent phenomenon, and the origins of adaptation in nature. ABMS began largely as the set of ideas, techniques, and tools for implementing computational models of complex adaptive systems. Many of the early agent-based models were developed using the Swarm modelling software designed by Langton and others to model ALife (Minar *et al*, 1996). Initially, agent behaviours were modelled using exceedingly simple rules that still led to exceedingly complex emergent behaviours. In the past 10 years or so, available agent-based modelling software tools and development environments have expanded considerably in both numbers and capabilities.

Following the conventional definition of simulation, we use the term ABMS in this article to refer to both agent-based simulation, in which a dynamic and time-dependent process is modelled, and more general kinds of agent-based modelling that includes models designed to do optimization (see, eg, Olariu and Zomaya, 2006) or search (see, eg, Hill *et al*, 2006). For example, particle swarm optimization and ant optimization algorithms are both inspired by agent-based modelling approaches and are used to achieve an end (optimal) state rather than to investigate a dynamic process, as in a simulation.

2.2. Structure of an agent-based model

A typical agent-based model has three elements:

1. A set of *agents,* their attributes and behaviours.
2. A set of agent *relationships* and methods of interaction: An underlying topology of connectedness defines how and with whom agents interact.
3. The agents' *environment*: Agents interact with their environment in addition to other agents.

A model developer must identify, model, and program these elements to create an agent-based model. The structure of a typical agent-based model is shown in Figure 1. Each of the components in Figure 1 is discussed in this section. A computational engine for simulating agent behaviours and agent interactions is then needed to make the model run. An agent-based modelling toolkit, programming language or other implementation provides this capability. To run an agent-based model is to have agents repeatedly execute their behaviours and interactions. This process often does, but is not necessarily modelled to, operate over a timeline, as in time-stepped, activity-based, or discrete-event simulation structures.

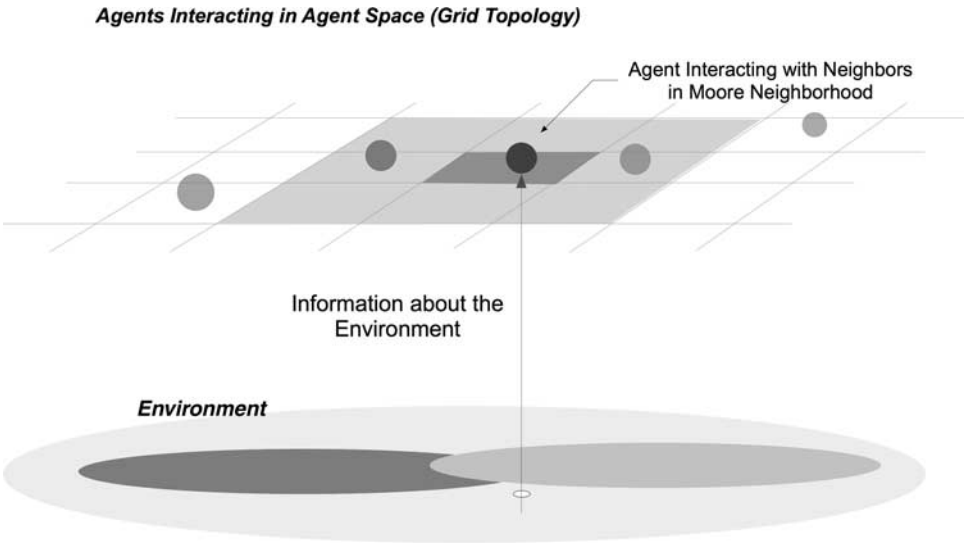

Figure 1 The structure of a typical agent-based model, as in Sugarscape (Epstein and Axtell, 1996).

2.3. Autonomous agents

The single most important defining characteristic of an agent is its capability to act autonomously, that is, to act on its own without external direction in response to situations it encounters. Agents are endowed with behaviours that allow them to make independent decisions. Typically, agents are active, initiating their actions to achieve their internal goals, rather than merely passive, reactively responding to other agents and the environment.

There is no universal agreement in the literature on the precise definition of an agent beyond the essential property of autonomy. Jennings (2000) provides a computer science definition of agent that emphasizes the essential characteristic of autonomous behaviour. Some authors consider any type of independent component (software, model, individual, etc) to be an agent (Bonabeau, 2001). In this view, a component's behaviour can range from simplistic and reactive 'if-then' rules to complex behaviours modelled by adaptive artificial intelligence techniques. Other authors insist that a component's behaviour must be adaptive, able to learn and change its behaviours in response to its experiences, to be called an agent. Casti (1997) argues that agents should contain both base-level rules for behaviour and higher-level rules that are in effect 'rules to change the rules'. The base-level rules provide more passive responses to the environment, whereas the 'rules to change the rules' provide more active, adaptive capabilities.

From a practical modelling standpoint, based on how and why agent-models are actually built and described in applications, we consider agents to have certain *essential* characteristics:

- An agent is a *self-contained,* modular, and uniquely identifiable individual. The modularity requirement implies that an agent has a boundary. One can easily determine whether something is part of an agent, is not part of an agent, or is a shared attribute. Agents have attributes that allow the agents to be distinguished from and recognized by other agents.
- An agent is *autonomous* and self-directed. An agent can function independently in its environment and in its interactions with other agents, at least over a limited range of situations that are of interest in the model. An agent has *behaviours* that relate information sensed by the agent to its decisions and actions. An agent's information comes through interactions with other agents and with the environment. An agent's behaviour can be specified by anything from simple rules to abstract models, such as neural networks or genetic programs that relate agent inputs to outputs through adaptive mechanisms.
- An agent has a *state* that varies over time. Just as a system has a state consisting of the collection of its state variables, an agent also has a state that represents the essential variables associated with its current situation. An agent's state consists of a set or subset of its attributes. The state of an agent-based model is the collective states of all the agents along with the state of the environment. An agent's behaviours are conditioned on its state. As such, the richer the set of an agent's possible states, the richer the set of behaviours that an agent can have. In an agent-based simulation, the state at any time is all the information needed to move the system from that point forward.
- An agent is *social* having dynamic interactions with other agents that influence its behaviour. Agents have protocols for interaction with other agents, such as for communication, movement and contention for space, the capability to respond to the environment, and others. Agents have the ability to recognize and distinguish the traits of other agents.

Agents may also have other useful characteristics:

- An agent may be *adaptive*, for example, by having rules or more abstract mechanisms that modify its behaviours. An agent may have the ability to learn and adapt its behaviours based on its accumulated experiences. Learning requires some form of memory. In addition to adaptation at the individual level, populations of agents may be adaptive through the process of selection, as individuals better suited to the environment proportionately increase in numbers.
- An agent may be *goal-directed*, having goals to achieve (not necessarily objectives to maximize) with respect to its behaviours. This allows an agent to compare the outcome of its behaviours relative to its goals and adjust its responses and behaviours in future interactions.
- Agents may be *heterogeneous*. Unlike particle simulation that considers relatively homogeneous particles, such as idealized gas particles, or molecular dynamics simulations that model individual molecules and their interactions, agent simulations often consider the full range of agent diversity across a population. Agent characteristics and behaviours may vary in their extent and sophistication, how much information is considered in the agent's decisions, the agent's internal models of the external world, the agent's view of the possible reactions of other agents in response to its actions, and the extent of memory of past events the agent retains and uses in making its decisions. Agents may also be endowed with different amounts of resources or accumulate different levels of resources as a result of agent interactions, further differentiating agents.

A typical agent structure is illustrated in Figure 2. In an agent-based model, everything associated with an agent is either an agent attribute or an agent method that operates on the agent. Agent attributes can be static, not changeable during the simulation, or dynamic, changeable as the

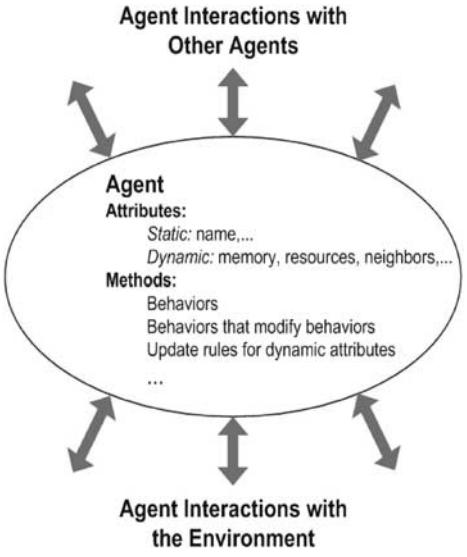

Figure 2 A typical agent.

simulation progresses. For example, a static attribute is an agent's name; a dynamic attribute is an agent's memory of past interactions. Agent methods include behaviours, such as rules or more abstract representations such as neural networks, which link the agent's situation with its action or set of potential actions. An example is the method that an agent uses to identify its neighbours.

A theory of agent behaviour for the situations or contexts the agent encounters in the model is needed to model agent behaviour. One may begin with a normative model in which agents attempt to optimize profits, utility, etc, as a starting point for developing a simpler, more descriptive, but realistic, heuristic model of behaviour. One may also begin with a behavioural model if there is available behavioural theory and empirical data to support the application. For example, numerous theories and empirically based heuristics exist for modelling consumer shopping behaviour. These could be implemented and compared in an agent-based model. Cognitive science and related disciplines focus on agents and their social behaviours (Sun, 2006). Behavioural modelling frameworks such as BDI (Belief-Desire-Intent) combine modal and temporal logics as the basis for reactive planning and agent action selection (Wooldridge, 2000). In agent-based modelling applications in which learning is important, theories of learning by individual agents or collectives of agents become important. The field of machine learning is another source of learning algorithms for recognizing patterns in data (such as data mining) through techniques such as supervised learning, unsupervised learning, and reinforcement learning (Alpaydỳn, 2004; Bishop, 2007). Genetic algorithms (Goldberg, 1989) and related techniques such as learning classifier systems (Holland *et al*, 2000) are also commonly used in agent-based models.

2.4. Interacting agents

Agent-based modelling concerns itself with modelling agent relationships and interactions as much as it does modelling agent behaviours. The two primary issues of modelling agent interactions are specifying who is, or could be, connected to who, and the mechanisms of the dynamics of the interactions. Both aspects must be addressed in developing agent-based models.

One of the tenets of complex systems and agent-based modelling is that only *local information* is available to an agent. Agent-based systems are decentralized systems. There is no central authority that either pushes out globally available information to all agents or controls their behaviour in an effort to optimize system performance. Agents interact with other agents, but not all agents interact directly with all the other agents all the time, just as in real-world systems. Agents typically interact with a subset of other agents, termed the agent's *neighbours*. Local information is obtained from interactions with an agent's neighbours (not *any* agent or *all* agents) and from its localized environment (not from *any* part of the entire environment). Generally, an agent's set of neighbours changes rapidly as a simulation proceeds and agents move through space.

How agents are connected to each other is generally termed an agent-based model's *topology* or connectedness. Typical topologies include a spatial grid or network of nodes (agents) and links (relationships). A topology describes who transfers information to whom. In some applications, agents interact according to multiple topologies. For example, a recent agent-based pandemic model has agents interacting over a spatial grid to model physical contact as agents go through daily activities and possibly pass on infections. Agents also are members of social networks that model the likelihood of contact with relatives and friends.

An agent's *neighbourhood* is a general concept applicable to whatever agent spaces are defined in the model. For example, an agent could interact only with its neighbours located close-by in physical (or geographical) space as well as neighbour agents located close-by in its social space as specified by the agent's social network.

Originally, spatial agent-based models were implemented in the form of cellular automata (CA). Conway's Game of Life (Gardner, 1970) is a good example. CA represent agent interaction patterns and available local information by using a grid or lattice environment. The cells immediately surrounding an agent are its neighbourhood. Each cell can be interpreted as an agent that interacts with a fixed set of neighbouring cells. The cell (agent) state is either 'on' or 'off' at any time. Most early spatial agent-based models had the form of a CA. Epstein and Axtell's *Sugarscape* model is an example (Epstein and Axtell, 1996). In *Sugarscape*, the topology was more complex than in a simple CA. Agents were mobile and able to move from cell to cell. The grid essentially became the agents' environment. Agents were

able to acquire resources from the environment that were distributed spatially across the grid.

Other agent interaction topologies are now commonly used for modelling agent interactions (Figure 3). In the CA model, agents move from cell to cell on a grid and no more than a single agent occupies a cell at one time. The von Neumann '5-neighbour' neighbourhood is shown in Figure 3a; the '9-neighbour' Moore neighbourhood is also common. In the Euclidean space model, agents roam in two, three or higher dimensional spaces (Figure 3b). Networks allow an agent's neighbourhood to be defined more generally. For the network topology, networks may be static or dynamic (Figure 3c). In static networks, links are pre-specified and do not change. For dynamic networks, links, and possibly nodes, are determined endogenously according to the mechanisms programmed in the model. In the geographic information system (GIS) topology, agents move from patch to patch over a realistic geo-spatial landscape (Figure 3d). In the 'soup', or aspatial model, agents have no location because it is not important (Figure 3e); pairs of agents are randomly selected for interaction and then returned to the soup as candidates for future selection. Many agent-based models include agents interacting in multiple topologies.

2.5. Agent environment

Agents interact with their environment and with other agents. The environment may simply be used to provide information on the spatial location of an agent relative to other agents or it may provide a rich set of geographic information, as in a GIS. An agent's location, included as a dynamic attribute, is sometimes needed to track agents as they move across a landscape, contend for space, acquire resources, and encounter other situations. Complex environmental models can be used to model the agents' environment. For example, hydrology or atmospheric dispersion models can provide point location-specific data on groundwater levels or atmospheric pollutants, respectively, which are accessible by agents. The environment may thus constrain agent actions. For example, the environment in an agent-based transportation model would include the infrastructure and capacities of the nodes and links of the road network. These capacities would create congestion effects (reduced travel speeds) and limit the number of agents moving through the transportation network at any given time.

3. Agent-based modelling applications

3.1. The nature of agent-based model applications

Agent-based modelling has been used in an enormous variety of applications spanning the physical, biological, social, and management sciences. Applications range from modelling ancient civilizations that have been gone for hundreds of years to modelling how to design new markets that do not currently exist. Several agent-based modelling applications are summarized in this section, but the list is only a small sampling. Several of the papers covered

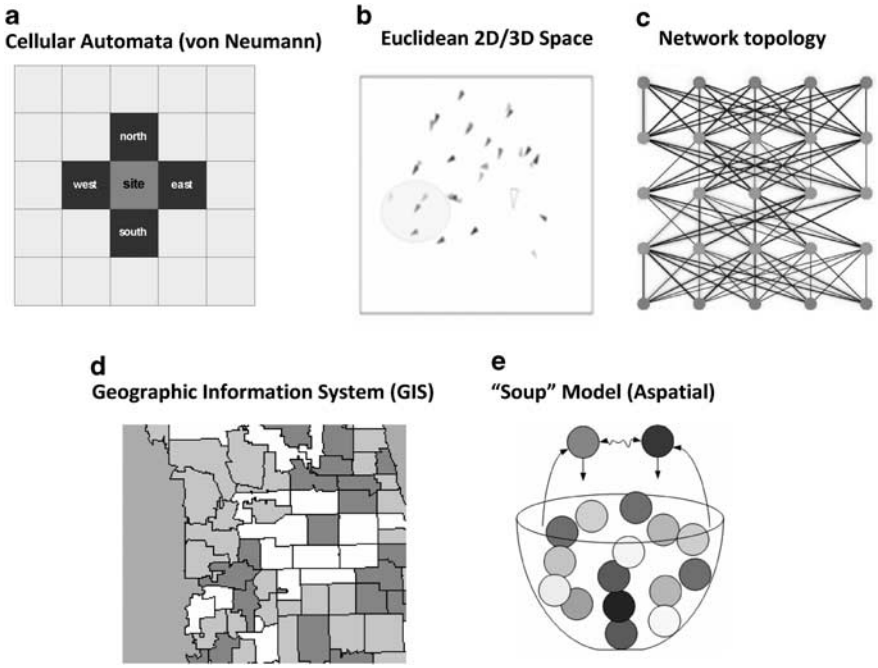

Figure 3 Topologies for agent relationships and social interaction.

here make the case that agent-based modelling, *versus* other modelling techniques is necessary because agent-based models can explicitly model the complexity arising from individual actions and interactions that exist in the real world.

Agent-based model structure spans a continuum, from elegant, minimalist academic 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. Decision support models tend to serve large-scale applications, are designed to answer real-world policy questions, include real data, and have passed appropriate validation tests to establish credibility.

3.2. Applications overview

Troisi *et al* (2005) applied agent-based simulation to model molecular self-assembly. Agents consist of individual molecules, and agent behaviours consist of the physical laws of molecular interaction. Such agent-based modelling approaches have found use in investigating pattern formation in the self-assembly of nano-materials, in explaining self-organized patterns formed in granular materials, and other areas.

In the biological sciences, agent-based modelling is used to model cell behaviour and interaction, the workings of the immune system, tissue growth, and disease processes. Generally, authors contend that agent-based modelling offers benefits beyond traditional modelling approaches for the problems studied and use the models as electronic laboratories as an adjunct to traditional laboratories. Cellular automata are a natural application for modelling cellular systems (Alber *et al*, 2003). One approach uses the cellular automata grid to model structures of stationary cells comprising a tissue matrix. Mobile cells consisting of pathogens and antibodies are agents that diffuse through and interact with tissue and other co-located mobile cells. The *Basic Immune Simulator* is built on a general agent-based framework to model the interactions between the cells of the innate and adaptive immune system (Folcik *et al*, 2007). Approaches for modelling the immune system have inspired several agent-based models of intrusion detection for computer networks (Azzedine *et al*, 2007) and modelling the development and spread of cancer (Preziosi, 2003). Emonet *et al* (2005) developed an agent-based simulator *AgentCell* for modelling the chemotaxis processes for motile behaviour of the *E. Coli* bacteria. In this multi-scale simulation, agents are modelled as individual molecules as well as whole cells. The model is used to study how the range of natural cell diversity at the molecular level is responsible for the observed range of cell movement behaviours.

In ecology, agent-based modelling is used to model diverse populations of individuals and their interactions. Mock and Testa (2007) develop an agent-based model of predator-prey relationships between transient killer whales and threatened marine mammal species (sea lions and sea otters) in Alaska. The authors state that until now only simplistic, static models of killer whale consumption had been constructed because of the fact that the interactions between transient killer whales and their marine mammal prey are poorly suited to classical predator-prey modelling approaches.

Agent-based epidemic and pandemic models incorporate spatial and network topologies to model people's realistic activity and contact patterns (Carley *et al*, 2006; Epstein *et al*, 2007). The focus is on understanding tipping point conditions that might lead to an epidemic and identifying possible mitigation measures. These models explicitly consider the role of people's behaviour and interactions through social networks as they affect the spread of infectious diseases.

Computational social science is an emerging field that combines modelling and simulation with the social science disciplines (Sallach and Macal, 2001). Agent-based models have been developed in the fields of economics, sociology, anthropology, and cognitive science. Various social phenomena have been investigated using agent-based models that are not easily modelled using other approaches (Macy and Willer, 2002; Gilbert and Troitzsch, 2005). Theoretical applications include social emergence (Sawyer, 2005), the emergence of cooperation (Axelrod, 1997), the generation of social instability (Epstein, 2002), and the collective behaviour of people in crowds (Pan *et al*, 2007). Sakoda (1971) formulated one of the first social agent-based models, the Checkerboard Model, which relied on a cellular automaton. Using a similar approach, Schelling developed a model of housing segregation in which agents represent homeowners and neighbours, and agent interactions represent agents' perceptions of their neighbours (Schelling, 1978). Schelling showed that housing segregation patterns can emerge that are not necessarily implied or consistent with the objectives of the individual agents. Epstein and Axtell (1996) extended the notion of modelling people to growing entire artificial societies through agent-based simulation in the grid-based *Sugarscape* model. *Sugarscape* agents emerged with a variety of characteristics and behaviours, highly suggestive of a realistic, although rudimentary and abstract, society. These early grid-based models with limited numbers of social agents are now being extended to large-scale simulations over realistic social spaces such as social networks and geographies through real-time linkages with GIS.

In many economic models based on standard micro-economic theory, simplifying assumptions are made for analytical tractability. These assumptions include (1) economic agents are rational, which implies that agents have well-defined objectives and are able to optimize their behaviour, (2) economic agents are homogeneous, that is, agents have identical characteristics and rules of behaviour, (3) the system experiences primarily 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.

Agent-based modelling allows relaxing the standard assumptions of classical economics (Arthur *et al*, 1997) so the transient states that are encountered along the way to equilibrium can be investigated (Axtell, 2000). This interest has spawned the field of Agent-based Computational Economics (Tesfatsion, 2002; Tesfatsion and Judd, 2006). Much applicable work is being done on understanding how people make decisions in actual situations in such fields as behavioural economics and neuro-economics. This work offers promise in building better empirically based models of agent behaviours that consider rational factors and emotion.

Agent-based models are being used to analyse markets, both existing and hypothetical. Charania *et al* (2006) use agent-based simulation to model possible futures for a market in sub-orbital space tourism. Each agent is a representation of an entity within the space industry. Tourism companies seek to maximize profits while they compete with other companies for sales. Customers evaluate the products offered by the companies according to their individual tastes and preferences. López-Sánchez *et al* (2005) developed a multi-agent based simulation of news digital markets adapting traditional business models to investigate market dynamics. Yin (2007) developed an agent-based model of Rocky Mountain tourism applied to the town of Breckenridge, Colorado; the model was used to explore how homeowners' investment and reinvestment decisions are influenced by the level of investment and amenities available in their neighbourhoods. Tonmukayakul (2007) developed an agent-based computational economics model to study market mechanisms for the secondary use of the radio spectrum. Using transaction cost economics as the theoretical framework, the model was used to identify the conditions for when and why the secondary use market could emerge and what form it might take.

Archaeology and anthropology are making use of large-scale agent-based modelling by providing an experimental virtual laboratory for long-vanished civilizations. Kohler *et al* (2005) employed large-scale agent-based simulations based on archaeological evidence to understand the social and cultural factors responsible for the disappearance of the ancient Pueblo in some parts of the south-western USA. Wilkinson *et al* (2007) used agent-based modelling to understand the growth and decline of ancient Mesopotamians.

Agent-based models of many real-world systems tend to consist of a mix of physical components (modelled as agents) and social agents, termed 'socio-technic' systems. Examples of such systems for which large-scale agent-based models have been developed include traffic, air traffic control, military command and control and net-centric operations, physical infrastructures and markets, such as electric power and integrated energy markets. For example, Cirillo *et al* (2006) used an agent-based approach to model the Illinois electric power markets under conditions of deregulation in an effort to anticipate likely effects on electricity prices and reliability.

This special issue adds to the growing list of agent-based model applications. Qu *et al* use their model of egg plant growth to promote understanding of the interactions between plant architecture and physiological processes. Chen and Hardoon use their model to examine cell division and migration in the colonic crypt to better understand the mechanisms of tumorigenesis.

4. Methods for agent-based modelling

4.1. Agent model design

When developing an agent-based model, it is useful to ask a series of questions, the answers to which will lead to an initial model design:

1. What specific problem should be solved by the model? What specific questions should the model answer? What value-added would agent-based modelling bring to the problem that other modelling approaches cannot bring?
2. What should the agents be in the model? Who are the decision makers in the system? What are the entities that have behaviours? What data on agents are simply descriptive (static attributes)? What agent attributes would be calculated endogenously by the model and updated in the agents (dynamic attributes)?
3. What is the agents' environment? How do the agents interact with the environment? Is an agent's mobility through space an important consideration?
4. What agent behaviours are of interest? What decisions do the agents make? What behaviours are being acted upon? What actions are being taken by the agents?
5. How do the agents interact with each other? With the environment? How expansive or focused are agent interactions?
6. Where might the data come from, especially on agent behaviours, for such a model?
7. How might you validate the model, especially the agent behaviours?

Answering these questions is an essential part of the agent-based model design process. There are a variety of approaches to designing and implementing agent-based models. North and Macal (2007) discuss both design methodologies and selected implementation environments in depth. Marsh and Hill (2008) offer an initial methodology for defining agent behaviours in an application for unmanned autonomous vehicles. Overall, bottom-up, highly iterative design methodologies seem to be the most effective for practical model development. Modern software (and model) development practices dictate that model design be independent of model implementation. That is, a good

software (model) design should be able to be implemented in whatever computer language or coding scheme is selected.

The communication of a model, its design assumptions, and detailed elements is essential if models are to be understood and reused by others than their original developers. Grimm *et al* (2006) present a proposed standard protocol for describing agent-based and related models as a first step for establishing a more detailed common format.

4.2. Agent model implementation

Agent-based modelling can be done using general, all-purpose software or programming languages, or it can be done using specially designed software and toolkits that address the special requirements of agent modelling. Agent modelling can be done in the small, on the desktop, or in the large, using large-scale computing cluster, or it can be done at any scale in-between these extremes. Projects often begin small, using one of the desktop ABMS tools, and then grow in stages into the larger-scale ABMS toolkits. Often one begins developing their first agent model using the approach that one is most familiar with, or the approach that one finds easiest to learn given their background and experience.

We can distinguish implementation alternatives to building agent-based models on the basis of the software used. Spreadsheets, such as Microsoft Excel, in many ways offer the simplest approach to modelling. It is easier to develop models with spreadsheets than with many of the other tools, but the resulting models generally allow limited agent diversity, restrict agent behaviours, and have poor scalability compared to the other approaches. Some macro-level programming is also needed using the VBA language.

General computational mathematics systems such as MATLAB and *Mathematica*, which many people may be already familiar with, can also be used quite successfully; however, these systems provide no specific capabilities for modelling agents. General programming languages such as Python, Java, and C++, and C also can be used, but development from scratch can be prohibitively expensive given that this would require the development of many of the available services already provided by specialized agent modelling tools. Most large-scale agent-based models use specialized tools, toolkits, or development environments based on reasons having to do with usability, ease of learning, cross-platform compatibility, and the need for sophisticated capabilities to connect to databases, graphical user interfaces and GIS.

4.3. Agent modelling services

Regardless of the specific design methodology that is selected, a range of services is commonly required for implementing large-scale models that include real data and geo-spatial environments, which are becoming more prevalent. Some of the more common capabilities include project specification services; agent specification services; input data specification and storage services; model execution services; results storage and analysis services; and model packaging and distribution services.

Project specification services provide a way for modellers to identify which sets of resources (eg files) constitute each model. There are three common approaches, depending on how much support the implementation environment provides for the modeller: (1) the library-oriented approach, (2) the integrated development environment (IDE) approach, and (3) the hybrid approach.

In the library-oriented approach to project specification, the agent modelling tool consists of a library of routines organized into an application programming interface (API). Modellers create models by making a series of calls to the various functions within the modelling toolkit. It is the responsibility of modellers to ensure that the correct call sequences are used and that all of the required files are present. In exchange, modellers have great flexibility in the way that they define their models. Examples include the Java archives (JAR) used by Repast for Java (North *et al*, 2006; ROAD, 2009) or MASON (GMU, 2009); the binary libraries used by Swarm (SDG, 2009); and the Microsoft .NET assemblies used by Repast for the Microsoft.NET framework (North *et al*, 2006; ROAD, 2009).

The IDE approach to project specification uses a code or model editing program to organize model construction. IDE's also provide a built-in mechanism to compile or interpret and then execute models. There are several options including combined 'one file' IDEs, factored multiple-file IDEs, and hybrid approaches. Combined 'one file' IDEs use a single file to describe each model. An example is NetLogo (Wilensky, 1999; NetLogo, 2009). These systems are often quite easy to initially learn and use, but do not always scale well to larger and more complex models as compared to the other project specification approaches. The scalability issues include difficulties supporting team development, difficulties with editing increasingly large model files, and difficulties in organizing and reorganizing model code as it grows. Factored multiple-file IDEs use a set of files to describe each model. They usually include some type of built-in file manager along with the editor. Factored multiple-file IDEs can use either custom development environments which are specially built for a given agent platform; standards-based environments such as Eclipse (Eclipse Foundation, 2009), or a mixture of custom and standards-based environments. Support for features like team development (ie two or more modellers simultaneously creating a model), version control (ie automated tracking of code changes), and refactoring (ie automated tools for reorganizing code) helps to make these environments more powerful than typical combined 'one file' IDEs. In many cases, these environments require more knowledge to use than 'one file' IDEs but they also tend to scale more effectively. However, they may be less flexible

than hybrid systems in the more extreme cases of model size and complexity.

The hybrid approach to project specification allows modellers to use the environment as either a stand-alone library or a factored multiple-file IDE. Examples include Repast Simphony (North *et al*, 2007; ROAD, 2009) and AnyLogic (XJ Technologies, 2009). In exchange for this added flexibility, these environments may require more knowledge to use than other types of IDEs but they also tend to scale the most effectively.

Agent specification services provide a means for modellers to define the attributes and behaviours of agents. These services can use general purpose languages such as C++ or Java; textual domain-specific languages (DSLs) such as *Mathematica* or MATLAB (Macal, 2004b); or visual DSLs such as the Repast Simphony flowchart shown in Figure 4. Along with or included in the language features, some implementation environments provide special support for features such as adaptation and learning (eg neural networks); optimization (eg genetic algorithms); social networks; geographical information systems (GIS); and systems dynamics.

Input data specification and storage services allow users to setup and save data that defines model runs. Input data setup can be done visually by pointing and clicking to create agents, by using custom programs to create agents in specified patterns, or by using external input data files in customized or standardized file formats. The standard storage formats can include extensible markup language (XML) files, spreadsheets, databases, or GIS files. Some

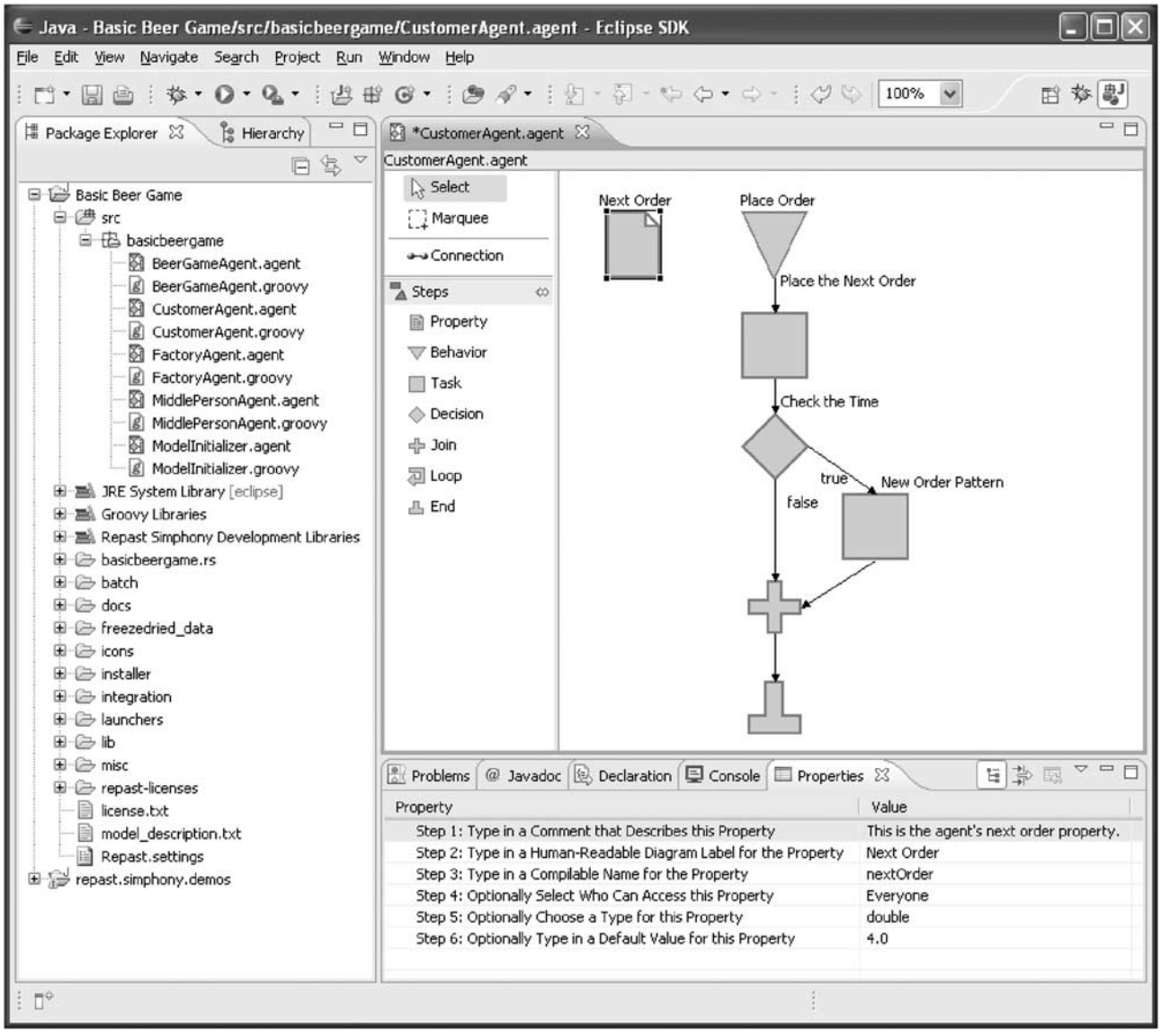

Figure 4 A Repast Simphony agent behaviour flowchart.

systems also allow 'checkpointing', which is saving and restoring the current state of a model at any time during execution.

Model execution services provide a means for model users to run and interact with simulations. Interactive execution can include viewing and modifying the attributes of agents (ie agent 'probing'); displaying agents in two and three dimensions; and running models without visual displays to quickly generate data (ie 'batch execution'). Batch execution can include the execution of multiple model runs on one local computer or on clusters of computers.

Results storage and analysis services allow model users to conveniently examine the results of individual model runs or sets of runs. Major analysis mechanisms include visualization, data mining, statistics, and report generation. Most implementation environments allow modellers to produce output text or binary files during execution, primarily using programming. These output files can then be manually read into separate external analysis tools. Some implementation environments such as Repast Simphony (North *et al*, 2007; ROAD, 2009) and AnyLogic (XJ Technologies, 2009) include either built-in analysis tools or point-and-click mechanisms to create output files and directly invoke external analysis tools.

Model packaging and distribution services allow modellers to disseminate completed models to end users. There are a range of methods for packaging models including embedded-platform packaging, IDE-based packaging, and stand-alone packaging. Once models are packaged there are several ways to distribute the results including file-based distribution, installer-based distribution, and web-based execution. In principle, any of the distribution options can be used with any of the packaging approaches. Embedded-platform packaging places models within larger surrounding software systems. This kind of packaging is often used for models that are built using the library project specification approach. This approach usually requires substantial software development knowledge. IDE-based packaging occurs when a model is developed using the IDE project specification approach and is then disseminated by distributing copies of the IDE with the model inside. This approach usually allows users to examine and change the model when they receive it. It also sometimes requires greater skill on the user's part compared to the other packaging approaches since IDEs can be somewhat complex. Stand-alone packaging binds a model into a program separate from the development environment that was used to create it. This new program, commonly called the 'runtime version' of the model, can be distributed to end users. This approach is usually the simplest for users who want to execute the model but not examine or change the code.

File-based distribution places the files that constitute the model in a user accessible location such as a CD, DVD, file server, or website. These files can be individually accessed or distributed in a compressed or uncompressed archive. Installer-based distribution uses a custom program which copies the model onto the user's computer and then configures it for execution. Installers usually have graphical wizard-based interfaces that make installation more reliable than for the other distribution approaches because of the ability of the installation software to automatically fix common configuration issues. Web-based execution embeds a packaged model into a web page for execution from within a browser. Web-based execution is differentiated from simply making raw files or an installer available from a website in that it requires models to execute from within a browser or browser plug-in rather than simply being downloaded and installed from an online source. Web-based execution is often the easiest and fastest distribution method for users. However, reliability can suffer because of the varying functionality of the wide range of browsers and browser plug-ins that are in common use today.

This section shows that there is a wide range of ways to implement agent-based models. When evaluating agent modelling tools, it should be noted that no one approach is universally better for all situations. Rather, different kinds of implementation approaches and environments have various strengths and weakness depending on the modelling questions of interest. Furthermore, it is common to use different tools during different stages of model development. For example, a modeller might start with a combined 'one file' IDE for initial model prototyping and then later transition to a factored multiple-file IDE as the model scales up in size and complexity. Therefore, the existing range of tools can best be thought of as a portfolio of options from which good selections can be made for each modelling question and stage.

5. Summary and conclusions

ABMS is a new approach to modelling systems comprised of autonomous, interacting agents. There are a growing number of agent-based applications in a variety of fields and disciplines. ABMS is particularly applicable when agent adaptation and emergence are important considerations. Many agent-based software and toolkits have been developed and are widely used. A combination of several synergistic factors is moving ABMS forward rapidly. These factors include the continuing development of specialized agent-based modelling methods and toolkits, the widespread application of agent-based modelling, the mounting collective experience of the agent-based modelling community, the recognition that behaviour is an important missing element in existing models, the increasing availability of micro-data to support agent-based models, and advances in computer performance. Taken together, these factors suggest that ABMS promises to have far-reaching effects into the future on how businesses use computers to support decision-

making, government uses models to make and support policy, and researchers use electronic laboratories to further their research.

Acknowledgements—This work was supported by the US Department of Energy under contract number DE-AC02-06CH11357.

References

Alber MS, Kiskowski MA, Glazier JA and Jiang Y (2003). On cellular automaton approaches to modeling biological cells. In: Rosenthal J and Gilliam DS (eds). *Mathematical Systems Theory in Biology, Communication, and Finance*, IMA Vol. 134, Springer: New York, pp 1–39.

Alpaydin E (2004). *Introduction to Machine Learning*. MIT Press: Cambridge, MA.

Arthur WB, Durlauf SN and Lane DA (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 University Press: Princeton, NJ.

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, DC.

Azzedine B *et al* (2007). An agent-based and biological inspired real-time intrusion detection and security model for computer network operations. *Comp Commun* **30**(13): 2649–2660.

Bagni R, Berchi R and Cariello P (2002). A comparison of simulation models applied to epidemics. *J Artif Soc Social Simul* **5**(3), http://jasss.soc.surrey.ac.uk/5/3/5.html accessed 30 June 2002.

Bishop CM (2007). *Pattern Recognition and Machine Learning*. Springer: New York.

Bonabeau E (2001). Agent-based modeling: Methods and techniques for simulating human systems. *Proc Natl Acad Sci* **99**(3): 7280–7287.

Carley KM *et al* (2006). Biowar: Scalable agent-based model of bioattacks. *IEEE Trans Syst Man Cybernet* **36**(2): 252–265.

Casti J (1997). *Would-Be Worlds: How Simulation is Changing the World of Science*. Wiley: New York.

Charania AC, Olds JR and DePasquale D (2006). *Sub-orbital Space Tourism Market: Predictions of the Future Marketplace Using Agent-based Modeling*. SpaceWorks Engineering, Inc.: Atlanta, GA, http://www.sei.aero/uploads/archive/IAC-06-E3.4.pdf.

Cirillo R *et al* (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.

Eclipse Foundation (2009). Eclipse home page. http://www.eclipse.org/.

Emonet T *et al* (2005). AgentCell: A digital single-cell assay for bacterial chemotaxis. *Bioinformatics* **21**(11): 2714–2721.

Epstein JM (2002). Modeling civil violence: An agent-based computational approach. *Proc Natl Acad Sci* **99**(90003): 7243–7250.

Epstein JM and Axtell R (1996). *Growing Artificial Societies: Social Science from the Bottom Up*. MIT Press: Cambridge, MA.

Epstein JM *et al* (2007). Controlling pandemic flu: The value of international air travel restrictions. *PLoS ONE* **2**(5): e401. doi:10.1371/journal.pone.0000401.

Folcik VA, An GC and Orosz CG (2007). The basic immune simulator: An agent-based model to study the interactions between innate and adaptive immunity. *Theoret Biol Med Model* **4**(39), http://www.tbiomed.com/content/4/1/39.

Gardner M (1970). The fantastic combinations of john conway's new solitaire game "Life". *Scient Amer* **223**: 120–123.

Gilbert N and Troitzsch KG (2005). *Simulation for the Social Scientist*, 2nd edn, Open University Press: Maidenhead, UK.

GMU (George Mason University) (2009). MASON home page. http://cs.gmu.edu/~eclab/projects/mason/.

Goldberg DE (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley: Reading, MA.

Grimm V *et al* (2006). A standard protocol for describing individual-based and agent-based models. *Ecol Model* **198**(1–2): 115–126.

Hill RR, Carl RG and Champagne LE (2006). Using agent simulation models to examine and investigate search theory against a historical case study. *J Simul* **1**(1): 29–38.

Holland J (1995). *Hidden Order: How Adaptation Builds Complexity*. Addison-Wesley: Reading, MA.

Holland JH *et al* (2000). What is a learning classifier system? In: Lanzi PL, Stolzmann W and Wilson SW (eds). *Learning Classifier Systems, from Foundations to Applications*. Springer-Verlag: London, UK, pp 3–32.

Jennings NR (2000). On agent-based software engineering. *Artif Intell* **117**: 277–296.

Kauffman SA (1993). *The Origins of Order: Self-Organization and Selection in Evolution*. Oxford University Press: Oxford, UK.

Kohler TA, Gumerman GJ and Reynolds RG (2005). Simulating ancient societies. *Scient Amer* **293**(1): 77–84.

Langton CG (1989). Artificial life. In: Langton CG (ed). *Artificial Life: The Proceedings of an Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems* (held September 1987, Los Alamos, New Mexico, Vol. VI in Santa Fe Institute Studies in the Sciences of Complexity), Addison-Wesley: Reading, MA, pp 1–47.

López-Sánchez M, Noria X, Rodríguez JA and Gilbert N (2005). Multi-agent based simulation of news digital markets. *Int J Comp Sci Appl* **II**(I). http://www.tmrfindia.org/ijcsa/v21.html.

Macal CM (2004a). Emergent structures from trust relationships in supply chains. In: Macal C, Sallach D and North M (eds). *Proceedings of Agent 2004: Conference on Social Dynamics: Interaction, Reflexivity and Emergence*. Argonne National Laboratory: Chicago, IL, 7–9 October, pp 743–760.

Macal CM (2004b). Agent-based modeling and social simulation with *Mathematica* and MATLAB. In: Macal C, Sallach D and North M (eds). *Proceedings of Agent 2004: Conference on Social Dynamics: Interaction, Reflexivity and Emergence*. Chicago, IL, 7–9 October. http://www.agent2004.anl.gov/pp 185–204.

Macal CM (2009). Agent based modeling and artificial life. In: Meyers R (ed). *Encyclopedia of Complexity and Systems Science*. Springer: New York, pp 112–131 (ISBN 978-0-387-75888-6).

Macy MW and Willer R (2002). From factors to actors: Computational sociology and agent-based modeling. *Ann Rev Sociol* **28**: 143–166.

Marsh WE and Hill RR (2008). An initial agent behavior modeling and definition methodology as applied to unmanned aerial vehicle simulation. *Int J Simul Process Model* **4**(2): 119–129.

Minar N, Burkhart R, Langton C and Askenazi M (1996). *The swarm simulation system, a toolkit for building multi-agent simulations*. Working Paper 96-06-042, Santa Fe Institute, Santa Fe, NM. http://www.santafe.edu/projects/swarm/overview/overview.html.

Mock KJ and Testa JW (2007). *An Agent-based Model of Predator-Prey Relationships between Transient Killer Whales and Other*

Marine Mammals. University of Alaska Anchorage, Anchorage, AK, 31 May 2007. http://www.math.uaa.alaska.edu/~orca/.
Moffat J, Smith J and Witty S (2006). Emergent behaviour: theory and experimentation using the MANA model. *J Appl Math Decis Sci* **10**: 1–13.
NetLogo (2009). NetLogo home page. http://ccl.northwestern.edu/netlogo/.
North M, Collier N and Vos J (2006). Experiences in creating three implementations of the repast agent modeling toolkit. *ACM Trans Model Comp Simul* **16**(1): 1–25.
North M *et al* (2009). Multi-scale agent-based consumer market modeling. *Complexity* **15**: 37–47.
North MJ and Macal CM (2007). *Managing Business Complexity: Discovering Strategic Solutions with Agent-Based Modeling and Simulation*. Oxford University Press: Oxford, UK.
North MJ, Tatara E, Collier NT and Ozik J (2007). Visual agent-based model development with repast simphony. In: Macal C, Sallach D and North M (eds). *Proceedings of Agent 2007: Conference on Complex Interaction and Social Emergence*. Chicago, IL, 7–9 October. http://www.agent2007.anl.gov/ pp 173–192.
Olariu S and Zomaya AY (eds). (2006). *Handbook of Bioinspired Algorithms*. Chapman & Hall/CRC: Boca Raton, FL USA, p 679.
Pan X, Han CS, Dauber K and Law KH (2007). A multi-agent based framework for the simulation of human and social behaviors during emergency evacuations. *AI Soc* **22**(2): 113–132.
Preziosi L (ed). (2003). *Cancer Modelling and Simulation*. Chapman and Hall/CRC: Boca Raton, FL.
ROAD (Repast Organization for Architecture and Design) (2009). Repast home page. http://repast.sourceforge.net/.
Sallach D and Macal C (2001). The simulation of social agents: An introduction. *Special Issue Soc Sci Comp Rev* **19**(3): 245–248.
Sakoda JM (1971). The checkerboard model of social interaction. *J Math Sociol* **1**: 119–132.
Sawyer RK (2005). *Social Emergence: Societies ad Complex Systems*. Cambridge University Press: Cambridge, UK.
Schelling TC (1978). *Micromotives and Macrobehavior*. Norton: New York.
SDG (Swarm Development Group) (2009). Swarm development group home page. http://www.swarm.org/.
Sun R. (ed). (2006). *Cognition and Multi-Agent Interaction: From Cognitive Modeling to Social Simulation*. Cambridge University Press: Cambridge.
Tesfatsion L (2002). Agent-based computational economics: Growing economies from the bottom up. *Artif Life* **8**(1): 55–82.
Tesfatsion L and Judd KL (eds). (2006). *Handbook of Computational Economics, Volume II: Agent-Based Computational Economics*. Elsevier/North-Holland: Amsterdam, p 904.
Tonmukayakul A (2007). *An agent-based model for secondary use of radio spectrum*. PhD Thesis, School of Information Sciences, University of Pittsburgh.
Troisi A, Wong V and Ratner M (2005). An agent-based approach for modeling molecular self-organization. *Proc Natl Acad Sci* **102**(2): 255–260.
Weisbuch G (1991). *Complex Systems Dynamics: An Introduction to Automata Networks* (translated from French by S. Ryckebusch), Addison-Wesley: Redwood City, CA.
Wilensky U (1999). *Netlogo, Center for Connected Learning and Computer-Based Modeling*. Northwestern University: Evanston, IL. http://ccl.northwestern.edu/netlogo/.
Wilkinson TJ *et al* (2007). Modeling settlement systems in a dynamic environment. In: Kohler TA and Leeuw SEvd (eds). *The Model-Based Archaeology of Socionatural Systems*. School for Advanced Research Press: Santa Fe, NM, pp 175–208.
Wooldridge M (2000). *Reasoning About Rational Agents*. MIT Press: Cambridge, MA.
XJ Technologies (2009). AnyLogic home page. http://www.xjtek.com/.
Yin L (2007). Assessing indirect spatial effects of mountain tourism development: An application of agent-based spatial modeling. *J Region Anal Pol* **37**(3): 257–265. http://www.jrap-journal.org/pastvolumes/2000/v37/F37-3-8.pdf.

Received 17 November 2009;
accepted 30 November 2009