Platforms and methods for agent-based modeling

Nigel Gilbert*† and Steven Bankes‡§

*Centre for Research on Simulation in the Social Sciences, University of Surrey, Guildford GU2 7XH, United Kingdom; ‡Evolving Logic, 3542 Greenfield Avenue, Los Angeles, CA 90034; and §RAND, 1700 Main Street, Santa Monica, CA 90407

The range of tools designed to help build agent-based models is briefly reviewed. It is suggested that although progress has been made, there is much further design and development work to be done. Modelers have an important part to play, because the creation of tools and models using those tools proceed in a dialectical relationship.

Agent-based modeling has a symbiotic relationship with computing technology. Modeling of the kind represented in the articles in this issue of PNAS became feasible only with the advent of personal workstations. As the technology has grown in power, the scale and sophistication of the software available for modelers has increased. And as more complex algorithms, toolkits, and libraries have been developed, more sophisticated models have become feasible for scholars working on their own or in small teams.

The development of computational tools to support modeling can be seen in the papers presented at a session on "Platforms and Methodologies for Enhancing the Social Sciences through Agent-Based Simulation," held as part of the National Academy of Sciences Sackler Colloquium in October 2001. The history follows in some ways the trajectory previously traced by statistical software. The earliest models were developed on mainframe computers or even, according to a probably apocryphal story, simulated by moving dinner plates around on a black and white tiled kitchen floor (1). From the early 1990s, most models have been developed in conventional programming languages such as C++, JAVA, and SMALLTALK. There has been little consensus about the best general purpose programming language to use; a quick survey of the articles published in the *Journal of Artificial Societies and Social Simulation* (http://www.soc.surrey.ac.uk/JASSS/) in 1998 and 1999 showed that not one language was used by more than one of the 18 papers that described a computational model. Those that were used included TURBOPASCAL, SQPC, C++, SOAR, Z, DYNAMO, and JAVA.

The disadvantages of using a general-purpose language are easy to see: every modeler has to reimplement basic algorithms, graphics libraries are often ill suited to dynamic modeling, and the resulting code is easily accessible only to those familiar with the language and the compiler needed to run it. Exactly the same situation occurred in the development of statistical software, although to a less extreme extent, because FORTRAN was then very widely used for writing analysis programs. Statisticians used to publish their programs verbatim in the appendixes of textbooks (compare the code in ref. 2).

The next step in the history of statistical analysis was the distribution of libraries of routines that could be included in one's own purpose-build program. In place of writing (and validating) one's own regression routine, for example, one could include a standard library procedure to do the job. A similar development is occurring in agent-based modeling, with several standardized libraries emerging. One is represented in this collection by the paper by Cederman (3), describing his work using the REPAST system. REPAST (http://repast.sourceforge.net/) is a set of JAVA libraries that allows programmers to build simulation environments (e.g., regular lattices), create agents in social networks, collect data from simulations automatically, and build user interfaces easily. Its features and design owe a lot to SWARM (http://www.swarm.org/), one of the first agent-based modeling libraries. Another similar library is ASCAPE (http://www.brook.edu/es/dynamics/models/ascape/) (4), described in Inchiosa's paper (5). ASCAPE is derived from the programs developed for Epstein and Axtell's *Growing Artificial Societies* (6), generalized to allow a wider range of models and refined to provide more powerful features.

These libraries have great advantages over "rolling your own," but also have some limitations. They require you to have a good working knowledge of the programming language that they are aimed at: in the case of ASCAPE and REPAST, that is JAVA. Although some models are much easier to program, especially ones that are similar to those that the developers themselves build, the construction of models using other approaches can be hindered by the need to find ways of working around the built-in assumptions. For example, both ASCAPE and REPAST are excellent for simulations involving agents located on a rectilinear grid. They are less useful for simulations that have no spatial aspects or for models that require a geographical information system to simulate an actual terrain.

The breakthrough in statistical computing was the development of "packages," that is, collections of routines assembled with a common standardized user interface, of which SPSS and SAS are the best-known early examples. There is nothing with the scale or sophistication of these statistical packages yet available for building agent-based models. What does exist are packages that will allow the building of very simple models using direct manipulation or "visual programming" meant mainly for students' and even schoolchildren's use, and some systems written in SMALLTALK that provide "total immersion" in an environment in which building blocks can be assembled. The best-known examples of the former are STARLOGO (http://www.media.mit.edu/starlogo/) and AGENTSHEETS (http://agentsheets.com/) (7, 8). However, to ensure that they are sufficiently straightforward for the target audience, some sacrifices in functionality have had to be made. For example, it is hard in AGENTSHEETS for one agent to interact directly with another agent (agents are not individually identified). It is very difficult in either system to develop agents possessing even simple cognitive models, and it would be impossible to create models using any kind of evolutionary approach (e.g., a genetic algorithm). In Europe, several systems based on SMALLTALK have been developed, including SDML (http://www.cpm.mmu.ac.uk/sdml/), CORMAS (http://cormas.cirad.fr/indexeng.

This paper results from the Arthur M. Sackler Colloquium of the National Academy of Sciences, "Adaptive Agents, Intelligence, and Emergent Human Organization: Capturing Complexity through Agent-Based Modeling," held October 4–6, 2001, at the Arnold and Mabel Beckman Center of the National Academies of Science and Engineering in Irvine, CA.

†To whom reprint requests should be addressed. E-mail: n.gilbert@soc.surrey.ac.uk.

htm) and DESIRE (http://www.cs.vu.nl/vakgroepen/ai/projects/desire/). These are much more complex and powerful than STARLOGO and AGENTSHEETS and take correspondingly longer to learn. Unlike the JAVA libraries such as ASCAPE and REPAST, they do not demand that users are fluent in the underlying programming language, in their case SMALLTALK, but they do require users to learn a complex interface that can be as difficult to master as a full programming language.

All of the existing tools are designed primarily to assist the builders of models. The facilities for other phases of a model's life cycle, model evaluation, model maintenance, and many types of model use are rather limited at this time. The primary supports for model use are visualizations of model state (especially the ubiquitous displays of two-dimensional grids of agent positions) and modest facilities for collecting statistics in a single run. Tool developers have not yet confronted issues of comparing multiple model runs, loading or calibrating models from data, automatically generating large numbers of cases from experimental designs, or collecting and statistically analyzing the results of large numbers of experiments. As the field of agent-based modeling moves from initial development and demonstration to regular use in social science, improved facilities for these other aspects of modeling will be needed (see the paper by S.B., ref. 9).

The ideal is a system that requires a minimum of learning, is completely flexible in the models that it will support, and runs efficiently on any hardware. As Axtell (10) points out in his contribution, efficiency may not at first seem to be important, given the strides made in computer hardware, but the ambitions of modelers are constantly rising, and there are problems for which the behavior of one million agents is significantly different from that of 100. The ideal is, of course, unobtainable, but more important, it is not even possible at the moment to imagine what it would look like. First, for a useful simulation platform to emerge, there needs to be some consensus about what kinds of model are best for social and economic simulation. That requires a great deal of hard work to advance the state of simulation within the modeling community itself. Designers of platforms can help sediment approaches and ideas, but the development of simulation platforms will always be too scientifically important to leave to the developers themselves.

1. Schelling, T. C. (1971) *J. Math. Sociol.* **1,** 143–186.
2. Gilbert, N. & Troitzsch, K. G. (1999) *Simulation for the Social Scientist* (Open Univ. Press, Buckingham, U.K.).
3. Cederman, L.-E. (2002) *Proc. Natl. Acad. Sci. USA* **99,** Suppl. 3, 7296–7303.
4. Parker, M. (2001) *J. Artificial Societies Social Simulation*, Vol. 4, http://www.soc.surrey.ac.uk/JASSS/4/1/5.html.
5. Inchiosa, M. E. & Parker, M. T. (2002) *Proc. Natl. Acad. Sci. USA* **99,** Suppl. 3, 7304–7308.
6. Epstein, J. & Axtell, R. (1996) *Growing Artificial Societies* (MIT Press, Cambridge, MA).
7. Repenning, A., Ionnidou, A. & Zola, J. (2000) *J. Artificial Societies Social Simulation*, Vol. 3, http://www.soc.surrey.ac.uk/JASSS/3/3/forum/1.html.
8. Carvalho, J. (2000) *J. Artificial Societies Social Simulation*, Vol. 3, http://www.soc.surrey.ac.uk/JASSS/3/3/forum/2.html.
9. Bankes, S. C. (2002) *Proc. Natl. Acad. Sci. USA* **99,** Suppl. 3, 7199–7200.
10. Axtell, R. L., Epstein, J. M., Dean, J. S., Gumerman, G. J., Swedlund, A. C., Harburger, J., Chakravarty, S., Hammond, R., Parker, J. & Parker, M. (2002) *Proc. Natl. Acad. Sci. USA* **99,** Suppl. 3, 7275–7279.