Consider millions of individuals making decisions on which products to buy, whether and how to recycle, and whether to drive or take public transportation to work. Each decision affects the use and flow of energy and materials in society. Some of these individuals may work for firms that design and manufacture consumer products, or for government agencies that plan, motivate, or approve public infrastructure elements. Other individuals may conduct research on more sustainable products, processes, and infrastructures. Still others may work to restructure institutions, modify organizational rules and procedures, or change reporting requirements. How are their personal decisions made? How is information generated, received, and processed to support such decisions? How do these decisions interact to determine the aggregate features of social behavior and the resulting economic and environmental profile and performance?

Historically, economists, engineers, and planners have addressed these questions with aggregate mathematical models, assuming informed, rational behavior leading to equilibria that are societally “optimal.” Similarly, to date much of the work in industrial ecology (IE) has focused on aggregate flows of resources, waste, people, and capital. Although this work is essential to understanding the broad dimensions and impacts of pollution generation by industry, it is almost certainly an inadequate basis for proceeding to policy, given its highly retrospective nature. Rather, in assessing the comparative efficacy of distinct policies, one must have a thorough understanding of the values, knowledge, and incentives faced by various actors in both the demand and supply sectors in question.

In recent years a disaggregate approach to analysis, referred to as “agent-based modeling,” has been motivated by new insights on the limitations of traditional economic assumptions and approaches, as well as computational advances (Axtell 2000). In an agent-based model, targeted subjects are represented as individual software objects that interact. The program is “run,” or executed, by simply letting the individual objects interact. The resulting evolution of the system is then studied both from the perspective of the aggregate population of agents and with respect to individual agent behavior. No assumptions concerning the attainment of equilibrium are necessary, as in the social sciences where, for example, the Nash equilibrium is seen as the main “solution concept” in game theory. Rather, equilibrium either obtains or not, and this one assesses from the actual model execution.

Indeed, the general problem of agency—the behavior of individuals in the face of imperfect incentives—lies at the core of IE. The phenom-
ena studied in IE originate in classical economic externalities and the departures from socially optimal behavior by industry and consumers due to distorted prices, incomplete markets, and imperfect regulations. An important role for agent-based modeling in IE should be, therefore, to explicitly treat the incentives that face behaviorally realistic agents in empirically credible environments. Suggestions along these lines have recently been made in this journal by two of us (Fischhoff and Small 2000; Andrews 2001). Here we focus more closely on the methodological aspects of agent models and ways in which industrial ecologists may begin to exploit their capabilities for systems modeling and insight.

The Intellectual Pedigree of Agents

A close relation exists between agents and so-called particle and cellular automata methods in physics and computer science. These methods have been used, for example, to represent fluid flows and have proven especially useful in the simulation of turbulent regimes, precisely where the mathematical representation of flow via the Navier-Stokes equations breaks down (Doolen 1990).

In the social sciences there existed a close connection between game theory and the digital computer from early on in the post–World War II era. Beginning at the RAND Corporation and continuing with developments in early behavioral economics research at Carnegie Mellon in the 1950s, early models were so heavily constrained by limited computing technology that, of necessity, they focused on two or at most a few individuals.

The modern conception of agents is often credited to Schelling (1978). His model of urban segregation situated several dozen purposefully behaving individuals with explicit behavioral rules on a spatial landscape and studied the typical configurations of the model. An important contribution of this work was his demonstration that a population of individuals, none of whom prefers segregated outcomes to integrated ones, can nonetheless end up in segregationist configurations by virtue of system effects—essentially, if agents of the same type wish to have some fraction of their neighbors of their same type, this leads to clustering of like agents at the aggregate level, thus producing segregation despite the innocuous preferences of the individuals. Interestingly, Schelling produced most of his results by hand, moving pennies and nickels around on a chessboard according to rolls of dice.

Today, agents are a rapidly growing research area in the social sciences as well as within computer science (Weiß 2000). In the latter field there has been a progressive movement of research in artificial intelligence toward the multiagent perspective, first in the guise of distributed artificial intelligence and most recently as multiagent systems.

A final dimension of the agent pedigree derives from research in artificial life, often associated with the Santa Fe Institute (Langton 1989). This line of research, pursued by both computer scientists and biologists, seeks to create “life forms” within software through genetic, evolutionary, and other means (Holland 1976).

Modern Agent Technology

Recent generations of agent models have been made possible by technical developments in both hardware and software. On the software side, the advent of object-oriented programming (OOP) has proved crucial for this development. In older programming languages such as FORTRAN, BASIC, or C, a program consists of data structures and procedures for modifying the data in those structures. OOP merges the notion of data and algorithm into a single structure, known as an object. An object’s data are its instance variables, whereas its procedures for modifying these variables are its so-called methods. This ability to encapsulate data and procedures into a single object facilitates object replication and reuse, two primary advantages of OOP.

One natural by-product of object technology was the ability to give each object “self-interest.” This might be accomplished by explicitly defining a utility function for each object and permitting the object to seek, through its behavior, improvements in its utility. Alternatively, in simpler environments in which the tradeoffs implicit in the utility calculus are not present, the objects would display their self-interest through merely
purposive behavior. Software objects endowed with such self-interest are known as “agents”.

On the hardware side, the exponential growth in computer memory, processing speed, and display resolution permitted the creation on personal computers of models in which a large number of objects could be instantiated and their behavior visualized in real time. Whereas the first generation of agent-based models of a decade ago employed a few simple agents, today one can deploy some tens of thousands of sophisticated agents, or even millions of simple agents, on a single machine. Typically, each agent in such models is characterized by the same behavioral rules, but heterogeneous data. Thus, the actual behavior of the individual agents can be quite variegated. Because agents are similar, the actual code that describes the typical agent can be quite short, whereas a computer’s entire memory can be populated with agents by cloning. Thus, agent models tend to have a small source (compile-time) code, but a large execution (run-time) profile that can be reduced by parallel processing.

An agent model is built simply by specifying the agent population, including all the ways in which agents interact. Then one creates computationally the environment in which the agents “live.” This might be a physical landscape or a social network, and also can be object oriented. The timing of agent activation is specified: Agents can operate in parallel or serially, and if the former then either partially or fully asynchronously. Finally, once the agent population is instantiated and permitted to interact, data-gathering objects—sometimes agents—can query the population. This is done so that the user might better understand the behavior of the overall system of agents, but could also be used to simulate firm or societal data collection and information dissemination. This information can be used by selected agents to assess system performance, and thus influence their subsequent decisions.

To implement working models there exist several agent-based software packages. SWARM (www.swarm.org), Ascape (www.brookings.edu/dynamics/models/ascape), and RePast (re-past.sourceforge.net) are among the best known of those suitable for constructing research-quality models. Each makes use of Java and is in the public domain, that is, is freely downloadable for noncommercial use.

Application to IE

Agent models have been applied in a variety of contexts of relevance to IE, including models of resource extraction and trade (Epstein and Axtell 1996) and organizational dynamics (Prietula et al. 1998). A recent dissertation uses agents to model the ways in which firms adapt to changed regulatory environments (Teitelbaum 1998). Future models will investigate conflicting incentives within firms as barriers to adopting efficient technologies. Agents can also be used to model interfirm interactions within an industry. The role of the media and other opinion-affecting agents such as citizen groups could be added to such a model in order to consider the full range of organizational stakeholders that influence private and public decisions. Individual consumers that learn to recycle, buy green products, and vote on environmental referenda also could be modeled. Such models will give industrial ecologists a needed test bed for safe, low-cost management and public policy experiments. For that purpose, models (so long as their predictions can be validated) will surely beat real life.

References


---

**About the Authors**

Robert L. Axtell is a fellow in the Economic Studies Program of the Brookings Institution in Washington, D.C., USA. Clinton J. Andrews is an assistant professor of urban planning and policy development at the Edward J. Bloustein School of Planning and Public Policy at Rutgers University in New Brunswick, New Jersey, USA. Mitchell J. Small is the H. John Heinz III Professor of Environmental Engineering at Carnegie Mellon University in Pittsburgh, Pennsylvania, USA.

Address correspondence to:
Mitchell J. Small
Carnegie Mellon University
Department of Engineering and Public Policy
Carnegie Mellon University
Porter Hall 119, Frew Street
Pittsburgh, PA 15213-3890 USA
ms35@andrew.cmu.edu