



Trust and Persistence

Paul A. Buhler • College of Charleston • pbuhler@cs.cofc.edu

Michael N. Huhns • University of South Carolina • huhns@sc.edu

We rely on computers to control our power plants and water supplies, our automobiles and transportation systems, and soon our economic and political systems. Increasingly, software agents are enmeshed in these systems, serving as the glue that connects distributed components. They translate messages, manage database queries and updates, represent buyers and sellers in e-commerce transactions, and may even cast votes on our behalf. Clearly, we need mechanisms to determine whether these agents are trustworthy.

What do we need to establish trust? Agents are often characterized by features such as autonomy, sociability, proactiveness, and persistent identity. This latter feature is key in determining trust. When agents operate over an extended period, they can earn a reputation for competence, timeliness, ease of use, and trustworthiness—something ephemeral agents cannot do.

Along with persistence, we need a reliable way to identify an agent and ensure that its true identity is not concealed. Researchers have addressed this issue in the context of mobile code, but only to the extent of protecting against malicious code alteration or behavior.

How can we assess an agent's trustworthiness? As with other aspects of agents and multiagent systems, we can take our cue from the human domain. Our reputations for trustworthiness are determined and maintained by the people we deal with. Analogously, a software agent's reputation will reside within other agents with whom it interacts. Obviously, agent reputations can't exist without persistence and unique identities.

The importance of reputation should not be underestimated: for both humans and agents, a good reputation serves as a social lubricant. As agent research and technology advance, we will ask agents to solve increasingly complex problems. An

agent deemed trustworthy will find it easier to establish relationships with other agents. These relationships provide access to resources that can assist an agent with its task at hand, just as a friendly relationship among neighbors can result in mutual assistance.

For some agent interactions, such as those involving commerce, agents will simply inherit the reputation of their human owner, sharing, for example, their owner's credit rating and financial capability. For other types of interactions, such as those involving information gathering, an agent will determine its own reputation through its efforts at gathering and distilling information. An agent with a reputation for conducting thorough searches will be trusted by other agents wishing to use its Web search results.

Envisioning Agent Societies

Multiagent systems harness cooperation and coordination among a collection of agents to perform meaningful work. These interacting agents form social structures that can resemble a community. Communities gain stability through reciprocal causal relationships among their members.¹ For example, you are more inclined to lend your neighbor your circular saw because he previously let you borrow his electric drill.

Conceptually, it's difficult to envision communities of software agents because we tend to view software applications as monolithic—that is, self-contained and self-sufficient. The word processor that helped create this column is not currently an aggregation of interacting agents. Someday, however, a word processor may represent an alliance uniting a file system agent, a display agent, a keyboard agent, and a spell-check agent sharing a common commitment to assist a user in producing a desirable outcome.

A Great Babbling Bazaar

Using a human enterprise as a reference model should help us understand emerging multiagent organizational structures. *The Cathedral and the Bazaar*, by Eric Raymond, characterizes the open-source community's software development culture.² Raymond compares traditional commercial software development to the building of a cathedral: a lead architect and a relatively small, tightly knit band of highly skilled artisans carry out the construction. But the Internet provides an infrastructure that enables a much larger, geographically dispersed community of developers to participate, facilitating complex software projects like the Linux operating system. These open-source development communities, Raymond says, resemble a "great babbling bazaar of differing agendas and approaches out of which a coherent and stable system" emerges. This model of the bazaar can help us visualize agents interacting within a multiagent system.

The human agents of the bazaar work within a community that shares a common desire to make a software product better or more functional. They interact with one another through software tools that support the community's distributed development needs by providing for configuration management, posting and assignment of tasks, threaded conversations, and so on. The open-source-developed SourceForge site (<http://www.sourceforge.net>) is one such collaborative assistance tool available to developers.

The persistent agents—human software developers—in these open-source communities gain reputations based on their creativity and ability to improve the software product. Reputation is an important measure of a developer's enculturation into the community and indicates how much influence the individual wields as a member of the society.

Reputation as Currency

In traditional economic systems, goods and services are exchanged for some

form of currency. Intellectual property takes this concept a step further by generating markets for intangible assets. Rambus Corp., for example, has created a market for its advanced memory chip technology without actually manufacturing the chips. The firm generates revenue through licensing agreements with manufacturers.³

What if the intangible asset of trade is reputation? In fact, economic systems already exist in which reputation is the currency; they're known as gift economies. In a gift economy, your reputation is based not on your possessions but on what you give away.

The scientific and academic communities follow the rules of a gift economy. A researcher's reputation is based on contributions to that researcher's field of study.⁴ Likewise, the open-source development community operates as a gift economy; a participating software developer's reputation is directly related to his or her contributions to the project.⁵

Implications of Gift Economies

Economic theory in regard to value and utility functions has been important to the development of rational agents. Agents can be programmed to behave in ways that maximize their utility while responding to changes in their environment. However, "economic models of agency, although quite general in principle, are typically limited in practice. This is because the value functions that are tractable essentially reduce an agent to one that is selfish."⁶ It is difficult to build a stable social system from a collection of agents motivated by self-serving interests. The gift economies of the bazaar present a new way to measure an agent's utility. The previously selfish agent now maximizes its utility by increasing its reputation through unselfish service to the society. This allows persistent agents to establish reputations.

Gift economies may also protect against the injection of rogue agents into a multiagent system. An agent without a reputation has little influence

in the gift economy. For a rogue agent to influence the society, it would have to establish its reputation by performing acts of service for the same society it intends to harm. This internal contradiction would force the rogue agent to function under cover. Additionally, the level of service required to gain a substantial reputation might require an agent to confer benefits that outweigh the intended harm.

Implementing a Bazaar

Building a persistent agent requires attention to memory usage. Naturally, all software should be developed without memory leaks, but developers of agents must also pay attention to what their agents forget as well as to what they learn. Many technical papers address knowledge acquisition, but

"This is a Space for a small quote"

few deal with knowledge loss. If systems don't occasionally discard information, they will eventually run out of memory and crash.

The open-source bazaar has external organizational history—that is, organizational memory. There's much to learn about an open-source software project and its contributing members by examining the artifacts of the development process itself. Through bug-fix histories, threaded message lists, and even comments within the code, distributed development environments provide information about the creators and maintainers of the software. Since this information lets us assess a developer's service, which is closely aligned with reputation, one developer can judge another without having any direct interaction.

A concrete example of a gift economy implementation is the Experts Exchange (<http://www.experts-exchange.com>).

com), an open marketplace where human experts collaborate and compete with each other in answering questions. The experts receive recognition in their area of expertise by accumulating "expert points" and satisfaction grades supplied by questioners. The experts gain additional points by helping each other. Accumulated points establish a reputation for knowledge within a given IT domain.

Amazon.com provides another example of reputation-assisted commerce. Online shoppers can read reviews submitted by other customers and indicate whether they found the reviews helpful. If customers consistently find someone's reviews valuable, that reviewer's reputation and influence increase. Amazon hopes such trust will lead to additional book purchases.

Online auction houses such as eBay offer similar services so that a bidder can determine whether a seller has a favorable reputation. These reputation-based mechanisms allow people to establish trust with total strangers. The existence of external information stores, from which reputation can be gleaned, relieves an agent from having to store and maintain this information itself.

Organizations like the IEEE Computer Society provide a membership number and an identification card. Producing these artifacts when required establishes legitimate membership in the organization. Perhaps the bazaar's external storage could provide a similar benefit through encryption. When an agent is admitted into the society, a public key could be provided. This key would not only unlock the resource but also establish membership in the society, much like a membership card. In many organizations, knowledge and information are segregated on a need-to-know basis. Hierarchical external memory with associated encryption keys could be used to support different levels of access. If occasional purging of membership roles is needed, new keys could be generated and issued to agents that have recently participated.

A Rich Metaphor

The basic building blocks of societal structures are systems and their relationships. Societies are aggregations of systems and are greater than the sum of their parts. Establishing meaningful relationships to facilitate social and financial interactions requires familiarity and trust. Familiarity requires persistent interaction, and trust depends on reputation. Implementing the robust and resilient societies envisioned by multiagent system researchers will require the adoption of new development paradigms. The open-source community and the bazaars in which it participates provide a rich metaphor for the implementation of multiagent societies. ■

Acknowledgment

This work was supported by the National Science Foundation under grant IIS-0083362.

References

1. D. Weissman, *A Social Ontology*, Yale Univ. Press, New Haven, Conn., 2000.
2. E. Raymond, *The Cathedral and the Bazaar*, available online at <http://www.tuxedo.org/~esr/writings/cathedral-bazaar>.
3. D. Takahashi, "Chipless Firm Stirs Commotion in Chip Market," *The Wall Street Journal*, Mar. 21, 2000, p. B1.
4. L. Hyde, *The Gift: Imagination and the Erotic Life of Property*, Vintage Books, N.Y., 1983.
5. E. Raymond, "Homesteading the Noosphere," available online at <http://www.tuxedo.org/~esr/writings/homesteading>.
6. M.N. Huhns and L.M. Stephens, "Multiagent Systems and Societies of Agents," in *Multiagent Systems*, G. Weiss, ed., MIT Press, Cambridge, Mass., 1999, p. 112.

Paul A. Buhler is an instructor of computer science at the College of Charleston and a PhD candidate at the University of South Carolina, where his primary research interest is agent-oriented software engineering.

Michael N. Huhns is a professor of computer science and engineering at the University of South Carolina, where he also directs the Center for Information Technology.