The Metaglue Software Agent System

Michael Coen, Brenton Phillips, Nimrod Warshawsky, Luke Weisman, Stephen Peters, Krzysztof Gajos, Nicholas Hanssens & Deb Dasgupta

Artificial Intelligence Laboratory
Massachusetts Institue Of Technology
Cambridge, Massachusetts 02139

http://www.ai.mit.edu

The Problem: Traditional programming languages provide no support for managing systems of interactive, distributed computations, i.e., those in which different components run asynchronously on a heterogeneous collection of networked computers.

Metaglue [5] is an extension to the Java programming language that provides very high-level support for writing large groups of software agents that interact with one another.

Motivation: Metaglue was developed as part of the AI Lab's Intelligent Room Project [4, 3]. The Intelligent Room has literally dozens of hardware and software components that run on a variety of networked workstations. We needed a system that could provide the *computational glue* for linking all of these components and coordinating the flows of data among them.

The computational needs of the Intelligent Room – while not unique – were not satisfied by any pre-existing software systems or programming environments. We wanted the Intelligent Room's software infrastructure to be persistent, robust, and dynamically reconfigurable. We needed the ability to modify (or even introduce) individual components without bringing the whole system down, and we wanted to have tools for understanding and debugging the behavior of large groups of interacting software agents.

Previous Work: There are currently several other pragmatic research systems for creating software agents [6, 7, 8]. They provide low-level functionality, e.g., support for *mobile agents* and directory services, which is necessary but not sufficient. These systems have no ability to manage groups of agents, to modify their activities in principled ways, or to abstractly and concisely describe their behaviors.

Metaglue is based on the Sodabot [1] programming language, which was used to create the first distributed control system in the Intelligent Room [2].

Approach: Metaglue adds several new primitives and capabilities to the Java programming language. At the lowest level, it allows distributed agents to locate and refer to one another by their functions and capabilities, without respect to where they are physically running. Metaglue agents are also mobile, in that they can move among physical computers while they are running.

Metaglue replaces Java's remote method invocation (RMI) mechanism with one that allows dynamic reconnection. This allows individual Metaglue software agents to invisibly resume previously established connections to other Metaglue agents that have been lost for some reason. Useful not only for ensuring reliability, this capability is essential for debugging a persistent, distributed software agent system. It makes it possible to stop, debug, and then seamlessly restart components in a running system. (No other current software agent programming language supports this.)

Using the current version of Metaglue, we have written a robust, distributed controller for the Intelligent Room that consists of 60 software agents running on eight networked workstations.

Difficulty: The largest challenge in the design of Metaglue has been making it powerful while keeping its semantics simple and elegant. It seems quite clear that for Metaglue to not only be useful but also influential, it must be easy to learn, understand, and remember.

We have also tried quite hard to resist *creeping featurism* – the tendency towards adding every new capability that came to mind during development.

Impact: Because Metaglue is an extension to the Java programming language, it shares Java's portability and flexibility. Its low computational overhead, coupled with the ease of transforming pre-existing software into "Metaglue agents," has made it a very attractive platform for creating systems of distributed, interacting, and persistent software agents. For this reason, Metaglue has been of enormous benefit in the Intelligent Room.

However, there are growing communities of researchers and implementers with similar computational needs, in areas ranging from information retrieval to online commerce to industrial manufacturing. We believe Metaglue's capabilities will be equally useful for them as well.

Future Work: We are currently adding a resource management system, to make it simpler for agents to request hardware devices or other agents with more localized knowledge of their requirements. In addition, we are adding a rule-based expert system environment (the Metaglue Expert System Shell) and extending the system to better represent agents acting on behalf of users.

Research Support: This material is based upon work supported by the Advanced Research Projects Agency of the Department of Defense under contract number F30602-94-C-0204, monitored through Rome Laboratory.

References:

[1] Coen, M. *SodaBot: A Software Agent Environment and Construction System*. MIT AI Lab Technical Report 1493, June, 1994.

[2] Coen, M. Building Brains for Rooms: Designing Distributed Software Agents. Proceedings of the Ninth Conference on Innovative Applications of Artificial Intelligence. IAAI-97. Providence, R.I.

[3] Coen, M. Design Principles for Intelligent Environments. Proceedings of the Fifteenth National Conference on Artificial Intelligence. AAAI-98. Madison, Wisconsin.

[4] Coen, M. The Future Of Human-Computer Interaction or How I learned to stop worrying and love My Intelligent Room. *IEEE Intelligent Systems*. March/April 1999.

[5] Coen, M., Phillips, B., Warshawsky, N., Weisman, L., Peters, S., and Finin, P. Meeting the Computational Needs of Intelligent Environments: The Metaglue System. To Appear MANSE99.

[6] General Magic. *Odyssey (Beta 2)* Agent System Documentation. http://www.genmagic.com/agents.

[7] IBM. *Aglets* Software Development Kit. http://www.trl.ibm.co.jp/aglets.

[8] ObjectSpace, Inc. *ObjectSpace Voyager Core Package Technical Overview (Version 1.0)*. December 1997. http://www.objectspace.com/voyager/whitepapers.