James Bond and Michael Ovitz: The Secret Life of Agents

Katia P. Sycara
The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213, U.S.A.
Voice: (412) 268-8825 Fax: (412) 268-5569
URL: http://www.cs.cmu.edu/~sycara/

Abstract

As agents populate Cyberspace in their many guises and roles, they coordinate and interact in different ways, spanning self-interested, as well as collaborative interactions. Agent coordination should be supported by an agent's internal architecture and agent societal frameworks. We take a micro-economic view of coordination. In this talk we report on our work on adaptive agent architecture and the primitive agent behaviors it supports, agent organizations, contracting protocols among agents and presence of middle agents.

Introduction

Effective use of the Internet by humans or decision support machine systems has been hampered by some dominant characteristics of the Infosphere. First, information is vast, unorganized, multi-modal, and distributed on server sites all over the world. Second, the number and variety of data sources and services is constantly changing. Third, information is ambiguous and possibly erroneous. Therefore, information is becoming increasingly difficult for a person or machine system to collect, filter, evaluate, and use in problem solving.

The notion of Intelligent Software Agents (e.g., (Cohen & Levesque 1987; Rao & Georgeff 1991; Wooldridge & Jennings 1995; Lang 1995; Sycara & Zeng 1994)) has been proposed to address this challenge. A precise definition of an intelligent agent is still forthcoming. For this talk, we will adopt the definition given in economic Agency Theory (Arrow 1985; Eisenhardt 1989; Bergen, Dutta, & Orville C. Walker 1992). An agency relationship is present whenever one party (the principal) depends on another party (the agent) to undertake some task on the principal's behalf. The agency relationship does not cover only economic situations. For example, James Bond is an agent acting on behalf of Her Majesty's Secret Service.

Most current agent-oriented approaches have focussed on what we call *interface agents*—a single agent with simple knowledge and problem solving capabilities whose main task is information filtering to alleviate the user's cognitive overload (e.g., (Maes 1994; Mitchell *et al.* 1994)). Another type of agent is the *Softbot* ((Etzioni & Weld 1994)), a single agent with general knowledge that performs a wide range of user-delegated information-finding tasks. Instead, we advocate use of multi-agent systems (Sycara & Zeng 1996; Oates, Prasad, & Lesser 1994). Such systems can compartmentalize specialized task knowledge, organize themselves to avoid processing bottlenecks, and can be built expressly to deal with dynamic changes in the agent and information-source landscape. In addition, multiple intelligent coordinating agents are ideally suited to the predominant characteristics of the Infosphere, such as the heterogeneity of the information sources, the diversity of information gathering and problem solving tasks that the gathered information supports, and the presence of multiple users with related information needs.

We have been developing RETSINA (Reusable Task Structure-based Intelligent Network Agents), an open society of reusable agents that self organize and cooperate in response to task requirements. In the course of the talk, we will discuss and illustrate how the individual agent architecture we have developed as well as the organization and coordination regimes of our agents support the following crucial characteristics of RETSINA:

- *multi-agent* system where the agents operate asynchronously and coordinate with each other and their users by forming dynamic *teams on demand* which fit in with the user's task and information requirements
- control is both *top down*, through user requests and also *bottom up* through active seeking and monitoring of information in the Infosphere
- the system operates robustly in an *open agent society* where agents, information sources or communication links can suddenly appear and disappear
- the information gathering is *seamlessly integrated* with problem solving and decision support

The structure of an individual agent is based on the BDI-model (Rao & Georgeff 1991; Decker 1995) and is compatible with architectures of behavior based autonomous robotic systems (Simmons 1994). We present a set of architectural building blocks that support the specification of behaviors for agents in a way that allows periodic actions, interleaving of planning and execution, and the concurrent activation of multiple behaviors with asynchronous components. The *planning* module takes as input a set of goals and produces a plan that satisfies the goals. The agent planning process is based on a hierarchical task network (HTN) planning formalism. The *communication and coordination* module accepts and interprets messages from other agents in KQML, or e-mail messages from human users. The *scheduling* module schedules each of the plan steps. Agent reactivity considerations are handled by the *execution monitoring* process. Each agent also has a *domain-independent library of plan fragments* (task structures) that are indexed by goals, as well as *domain-specific library of plan fragments* from which plan fragments can be retrieved and incrementally instantiated according to the current input parameters. The retrieved and instantiated plan fragments are used to form the agent's instantiated task tree that is incrementally executed. The *belief and facts* data structures contain facts and other knowledge related to the agent's functionality.

We present an initial set of implemented agent behaviors, including responding to repetitive queries, monitoring information sources, advertising capabilities, and self cloning. By "behaviors", we mean the execution by an agent of partially-ordered sequences of basic actions. By "reusable", we mean that the behaviors are specified in terms of domain-independent abstractions and can be reused in building an agent for a new domain or task. Currently we have identified and implemented in each RETSINA agent the following behaviors:

- **Advertising:** Upon startup, every agent creates an internal goal to advertise itself. An agent advertises itself by sending a middle agent the information needed for describing its capabilities and the services that the advertising agent can provide.
- **Message Polling:** Message Polling is the simplest agent behavior. An agent initialization process asserts a goal for the agent to collect and process incoming KQML messages.
- **Answering Simple Queries:** A simple query is one where the agent finds answers to queries and returns the results to the query-initiator. The query might be a *one-shot* question or it might be a request for *periodic* monitoring of a particular information source.
- **Information Monitoring:** An information monitoring query is one that is interpreted as expressing a condition that, when true, will trigger the transmission of the selected information. This condition is periodically checked with given frequency.
- **Cloning:** Cloning is one of an agent's possible responses to overloaded conditions. To recognize whether it is overloaded, the agent uses a simple model of how its ability to meet new deadlines is related to the characteristics of its current queries and other tasks. It compares this model to a hypothetical situation that describes the effect of adding a new agent. If this evaluation suggests that adding a new agent would be beneficial, the agent removes itself (temporarily) from actively pursuing new queries (by "unadvertising" its services) and creates a new agent that is a clone of itself.

The most important reason for our approach is that behavior specification is the proper level for allowing people to construct new classes of software agents in a structured, well-defined way. We are currently working on an *Agent Behavior Editor* which will allow more rapid construction of new classes of agents through the reuse and combination of existing behaviors, as well as specification of new behaviors.

Middle Agents

In open world environments, agents in the system are not statically predefined but can dynamically enter and exit an organization. One of the basic problems facing designers of open, multi-agent systems for the Internet is the connection problem (Davis & Smith 1983)—finding the other agents who might have the information or other capabilities that are needed in support of a task. There are two special types of information used in this process: preferences and capabilities. In multi-agent information systems, a preference is (meta) knowledge about what types of information have utility for a requester. A capability is (meta) knowledge about what types of requests can be serviced by a provider. In open systems, agents as well as their capabilities and preferences can dynamically change.

Agents that deal with preference or capability information and that are neither requesters/principals or providers/agents (from the standpoint of the transaction under consideration), we call *middle-agents*. In human societies, agents such as Michael Ovitz serve as middle agents facilitating service requesters and service providers to get in touch with each other. Middle agents have not been explicitly modeled in Agency Theory. We will examine the connection problem from the standpoint of privacy considerations. From a privacy standpoint, preference information can flow from a requester to a provider, and capability information can flow the other way. Privacy, however, is only one concern when choosing a solution to the connection problem. A designer also needs to consider other characteristics, such as the efficiency with which requests are handled and resources are used, the vulnerability of

the system to the failure of some component, and the ability to quickly adapt to changing preferences and capabilities. Our ongoing research aims to develop empirically validated models of the relationships between the various performance characteristics and system parameters.

Preference information can initially be kept private at the requester, be revealed to some middle agent, or be known by the provider itself. The same three possibilities exist for capability information. This leads to nine general middle-agent roles in information-gathering organizations. Here we mention the three most significant middle agent types. A *blackboard* is a middle-agent that keeps track of requests. Requesters post their problems; providers can then query the blackboard agent for events they are capable of handling. This class includes newsgroups and bulletin boards. A *broker* is a middle-agent that protects the privacy of both the requester and provider. The broker understands both the preferences and capabilities, and routes both requests and replies appropriately. Neither the requester nor provider ever knows directly about the other in a transaction. A *matchmaker/yellow-pages* is a middle agent that stores capability advertisements that can then be queried by requesters. The requesters then choose and contact any provider they wish directly.

We will discuss the scope of design possibilities presented by our model. We will also present experimental results that show tradeoffs among agent types using various resource allocation and load balancing schemes. Our experimental results were achieved using an implementation of the RETSINA framework in multi-agent financial portfolio management (the WARREN system)(Sycara *et al.* 1996).

Agency and Contracting

Each agent may handle requests from several other agents and may be in a position to choose which requests it will honor in order to use its local resources most effectively. Correspondingly, an agent chooses to request services from the agent who offers the most attractive deal. Thus, in the most general case there is an *electronic marketplace* consisting of agents that have their own goals and resources, and follow their own strategies (e.g. (Kraus 1994), (Sandholm & Lesser 1995)). The design and analysis of interaction protocols for such agents is part of the growing field of *automated negotiation systems*(Oliver 1996; Rosenschein & Zlotkin 1994). A major component of automated negotiation is *contracting.*

Agency theory uses the metaphor of a contract to describe relationships in which one party delegates work to another. The focus of the theory is on determining the most efficient contract to govern a particular relationship given the characteristics of the parties involved, environmental uncertainty and incomplete information. In general, there are two sets of issues when entering a relationship with an agent. First, *precontractural* issues, arising before the principal decides to offer an agent a contract. The major issues here are whether a particular agent has the capabilities the principal is seeking and what strategy the principal can follow in order to find out. In RETSINA, such issues are handled by middle agents. Second, there are *postcontractual* issues after the principal and the agent have engaged in a relationship. Such issues include how the principal should evaluate and reward the agent's performance, and what information strategy could be used to make such an evaluation. In RETSINA, these issues are handled through (1) the execution monitoring module and (2) option-based computational algorithm for valuing contingent contracts.

In most existing literature on agency theory, game theory or multi-agent systems, the contracts considered are binding. A contract is *binding* for an agent if the agent cannot get out of its contractual obligations. When an agent can get out of a contractual obligation, the contract is called *non-binding* or *contingent.* Contingent contracts allow agents increased flexibility and, in many situations, non-binding contracts are superior to binding ones. Introducing contingent contracts has two main advantages: (1) The space of possible contracts is enhanced, so the expected utility can be higher, and (2) Contingent contracts can reduce the variability of an agent's payoff, since an agent can postpone a decision for the future when more information could be available.

In game theory, the value of a contract is assumed known and used as input to the game-theoretic solution concepts such as Nash equilibrium and its extensions (e.g., sequential equilibrium, perfect Bayesian equilibrium (Fudenberg & Tirole 1991)). This approach does not address issues such as contract valuation, contract flexibility, or the nonstationary nature of the underlying uncertainty. More sophisticated computational mechanisms are needed.

The approach we will present is based on *financial option pricing* theory. We believe that modeling contingent contracts under time-dependent uncertainty and risk as options provides a natural unified framework for taking into account contracting flexibility and complex forms of environmental uncertainty. In addition, option pricing provides a computationally tractable formalism for calculating optimal values of various contracting decision parameters, that to date have not been rigorously modeled. Such parameters include the *value* of a flexible/contingent contract, *when to give* out a contract to an agent, *when to break* a contract, and *which contract to accept* out of a set of offered contracts.

There are many technical difficulties when allowing contingent contracts that make traditional methods obsolete. For instance, in decision analysis with contingent contracts, a decision maker is allowed to have the opportunity to exercise an option at any time point.

Mathematically, this entails the introduction of time-dependent processes into the model. Traditional models don't provide computationally tractable methods to address general time-dependent (non-stationary) random processes. Another example is how to deal with continuous information gathering/updating which can happen before the decision maker makes any commitment. Option pricing theory whose baseline mathematical model is based on a fairly general stochastic optimal control framework, provides satisfactory answers for modeling and evaluating these complicated phenomena.

Despite possible flexibility advantages, uncontrolled breaking of contingent contracts could lead to system thrashing. Therefore, there is a need to understand the tradeoffs involved. We will present experimental results that deal with agent-level and system-level tradeoffs between binding and contingent contracts. At the close of the talk, we will tie together the various intellectual threads presented during the talk.

Acknowledgements

This research has been sponsored in part by ONR Grant #N-00014-96-1-1222, by ARPA Grant #F33615-93-1-1330, and by NSF grant #IRI-9612131. Current and past members of the RETSINA group include Keith Decker, Constantine Domashnev, Somesh Jha, Anandeep Pannu, Onn Shehory, Rande Shern, Vandana Verma, Mike Williamson and Dajun Zeng,

References

Arrow, K. J. 1985. The economics of agency. In *Principles of Agents: The Structure of Business.* Harvard Business School Press. chapter 2, 37–51.

Bergen, M.; Dutta, S.; and Orville C. Walker, J. 1992. Agency relationships in marketing: A review of the implications and applications of agency and related theories. *Journal of Marketing* 56:1–24.

Cohen, P. R., and Levesque, H. J. 1987. Intention=choice + commitment. In *Proceedings of AAAI-87*, 410–415. Seattle, WA.: AAAI.

Davis, R., and Smith, R. G. 1983. Negotiation as a metaphor for distributed problem solving. *Artificial Intelligence* 20(1):63–109.

Decker, K. 1995. *Environment Centered Analysis and Design of Coordination Mechanisms.* Ph.D. Dissertation, University of Massachusetts.

Eisenhardt, K. M. 1989. Agency theory: An assessment and review. *Academy of Management Review* 14(1):57–74.

Etzioni, O., and Weld, D. 1994. A softbot-based interface to the internet. *Communications of the ACM* 37(7).

Fudenberg, D., and Tirole, J. 1991. *Game Theory.* The MIT Press.

Kraus, S. 1994. Contracting tasks in multi-agent environments. In *Technical Report CS-TR 3254 UMIACS-TR-94-44.*

Lang, K. 1995. Newsweeder: Learning to filter netnews. In *Proceedings of Machine Learning Conference.*

Maes, P. 1994. Agents that reduce work and information overload. *Communications of the ACM* 37(7).

Mitchell, T.; Caruana, R.; Freitag, D.; McDermott, J.; and Zabowski, D. 1994. Experience with a learning personal assistant. *Communications of the ACM* 37(7).

Oates, T.; Prasad, M. V. N.; and Lesser, V. R. 1994. Cooperative information gathering: A distributed problem solving approach. Technical Report UMass Computer Science Techincal Report 94-66, Depart of Computer Science, University of Massachusetts.

Oliver, J. 1996. *On automation negotiation and electronic commerce.* Ph.D. Dissertation, The Wharton School, Univ. of Pennsylvania.

Rao, A. S., and Georgeff, M. P. 1991. Modeling rational agents within a BDI-architecture. In *Proceedings of Knowledge Representation and Reasoning*, 473–484.

Rosenschein, J., and Zlotkin, G. 1994. *Rules of encounter: Designing conventions for automated negotiation systems.* MIT Press.

Sandholm, T., and Lesser, V. 1995. Issues in automated negotation and electronic commerce: extending the contract net protocol. In *Proc. First Int. Conf. on Multiagent Systems (ICMAS-95).*

Simmons, R. 1994. Structured control for autonomous robots. *IEEE Journal of Robotics and Automation.*

Sycara, K., and Zeng, D. 1994. Towards an intelligent electronic secretary. In *Proceedings of the CIKM-94 (International Conference on Information and Knowledge Management) Workshop on Intelligent Information Agents.*

Sycara, K., and Zeng, D. 1996. Coordination of multiple intelligent software agents. *International Journal of Cooperative Information Systems* 5(2 & 3):181–211.

Sycara, K.; Decker, K.; Pannu, A.; Williamson, M.; and Zeng, D. 1996. Distributed intelligent agents. *IEEE Expert* 11(6).

Wooldridge, M., and Jennings, N. R. 1995. Intelligent agents: Theory and practice. *The Knowledge Engineering Review* 10(2):115–152.