DESIGNING ROBUST, OPEN ELECTRONIC MARKETPLACES OF CONTRACT NET AGENTS

Chrysanthos Dellarocas
Mark Klein
Sloan School of Management
Massachusetts Institute of Technology
U.S.A.

Abstract

The creation of open agent-mediated electronic marketplaces requires new architectures, capable of coping with unreliable computational and network infrastructures, limited trust among independently developed agents, and the possibility of systemic failures. In analogy with civil human societies, open marketplaces will benefit from the introduction of appropriate electronic social institutions, whose role will be to help guarantee stability and efficiency in the face of these challenges. This paper presents a research methodology for designing and evaluating such electronic social institutions. It also describes how the methodology is currently being applied in order to design and evaluate robust open architectures for agent-mediated marketplaces based on the contract net protocol.

Keywords: Software agents, electronic commerce, simulation and modeling

1. INTRODUCTION

Software agent technologies promise substantial increases in productivity by automating several of the most time-consuming stages of electronic commerce processes. Agents are software systems, which are capable of interacting with other agents in a flexible and autonomous way, in order to meet the design objectives of their creators (Jennings, Sycara and Wooldridge 1998). In the context of electronic commerce, we can already point to several examples of agents used to compare information about products, buy products, sell products, etc. (Maes, Guttman and Moukas 1999).

Electronic agent marketplaces are formed by collections of software agents, which interact with one another in order to automatically negotiate and form partnerships or trade products and services through the Internet. In the emerging model of 21st century electronic commerce, a variety of open electronic marketplaces will be competing with one another for participants. Independently developed agents will be entering and leaving marketplaces at will, in pretty much the same way that human investors enter and leave different financial markets today. The stakeholders of electronic marketplaces will, therefore, have an interest in making them as attractive to prospective "customers" as possible. One expects that the most successful marketplaces will be the ones that have low barriers to entry (in terms of required agent sophistication) and provide the best "quality of service" guarantees (in terms of security, fairness, efficiency, etc.). The proper design of open electronic marketplaces thus emerges as an important research and practical question.

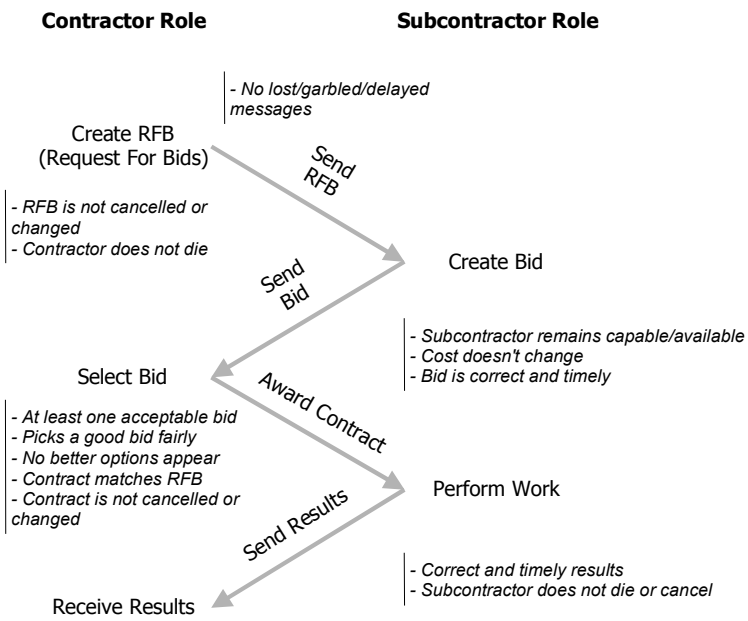

Figure 1. Simplified Description of the Contract Net Protocol
Some of the protocol assumptions are listed in italics.

A lot of the early work on the design of agent marketplaces focused on agent *mechanism design*, that is, on the design of rules of behavior to be followed by individual agents (Wellman and Wurman 1998). The underlying assumption behind this line of work is that if all agents follow the "right" mechanism, the emerging society will exhibit stable and efficient behavior.

Such research typically assumes that agents will be homogeneous and rational, that their infrastructure will be reliable, and therefore that their relatively simple and "optimistic" rules of behavior will be "intelligent" enough to avoid or cope with whatever deviant behavior or systemic dysfunction they encounter. The contract net protocol, for example, one of the best-known mechanisms for structuring contractor and subcontractor marketplaces (Smith 1980), owes its simplicity to many assumptions about agent behavior, some of which are listed in Figure 1. Although such assumptions may be possible to guarantee in closed environments, where all agents are developed by the same team, they are becoming less realistic in the open world of the Internet.

Designing efficient and robust open electronic marketplaces, whose participants will be independently developed software agents, is a difficult problem. Some of the most important challenges include:

- *Unreliable infrastructure.* Unpredictable node and link failures may cause agents to unexpectedly die in the middle of transactions, messages to be delayed or lost, etc. Open marketplaces must be able to gracefully cope with the notoriously unreliable nature of the Internet.

- *Limited trust*. Independently developed agents can not always be trusted to follow the rules properly due to bugs, bounded rationality, malice, and so on. For example, subcontractor agents may crash or fail to deliver a promised service on time, contractor agents may refuse to pay, etc. Open marketplaces should be prepared to deal with potential fraud or other deviant behavior.

- *Possibility of systemic failures*. Most market mechanisms, especially those simple enough to be reasonable for implementation and efficient in execution in a large set of agents, will have "holes" in terms of the potential for unintended emergent dysfunctional behaviors. This is especially true since agent societies operate in a realm where relative coordination, communication, and computational costs and capabilities can be radically different from those in human society, leading to behaviors with which we have little previous experience. It has been argued, for example, that the 1987 stock crash was due in part to the action of computer-based "program traders" that were able to execute trade decisions at a speed and volume that was unprecedented in human experience and thus led to unprecedented stock market volatility (Waldrop 1987).

Human societies have successfully coped with similar challenges by developing *social institutions* that set and enforce laws (e.g., courts, police), monitor for and respond to emergencies (e.g., ambulance system), prevent and recover from disasters (e.g. coast guard, firefighters), etc. In that way, civil societies allow citizens to utilize relatively simple, optimistic and efficient rules of behavior, offloading the prevention and recovery of many problem types to social institutions that can handle them efficiently and effectively by virtue of their economies of scale and widely-accepted legitimacy. Successful civil societies have thus achieved a division of labor between individuals and institutions that decreases the "barriers to survival" for each citizen, while helping increase the welfare of the society as a whole. In an analogous manner, we believe that the design of the right *electronic social institutions* will be a crucial success factor in the new universe of open electronic marketplaces.

Isolated examples of useful electronic social institutions have been proposed and analyzed by software agent researchers (for example, reputation mechanisms in Zacharia, Moukas and Maes 1999). However, up to this date, there has been almost no experience in systematically identifying the issues that arise when popular agent market mechanisms are used in an open environment, as well as the electronic institutions that are needed in order to provide robust, stable, and efficient systemic behavior.

Our work aims to fill this gap. Our research complements much of the current research in designing agent-mediated electronic marketplaces by focusing on the design of appropriate *social* (shared infrastructure) mechanisms that complement the mechanisms of *individual* agents in order to improve the flexibility, robustness, and efficiency of the resulting systems.

2. TOWARD ROBUST, OPEN MARKETPLACES OF CONTRACT NET AGENTS

Our methodology consists of the following three steps: (1) systematic analysis of mechanism exceptions in an open environment, in order to identify issues that may require the introduction of social institutions, (2) design of candidate institutions, and (3) simulation-based experimental evaluation, in order to determine how the introduction of these institutions affects various system properties. This section describes our ongoing work in applying this methodology to design and evaluate robust open architectures for agent-mediated marketplaces based on the contract net protocol.

2.1 Exception Analysis

Given an agent society whose members interact using a given market mechanism, the first step in our methodology consists of a systematic identification of possible exceptions that may arise in an open environment.

We define exceptions as any deviation from an ideal sequence of agent behaviors. We further distinguish exceptions into infrastructure failures, protocol violations, and systemic exceptions. *Infrastructure failures* are exceptions due to the limited reliability of computers, networks, and software. In the context of the contract net protocol, examples include contractor and subcontractor agents crashing in the middle of transactions, messages getting lost or delayed, etc. *Protocol violations* are situations where the behavior of an agent deviates from the agreed upon protocol. This may be due to malice, incompatible agent implementation, or simply a programming bug. An example of a violation of the contract net protocol is a situation where a contractor agent awards a contract to a subcontractor but later refuses to accept the results and deliver payment. Finally, *systemic exceptions* describe unintended emergent dysfunctional behaviors, which are usually due to inherent weaknesses of the protocol. For example, one potential systemic dysfunction of the contract net protocol that has been reported in the literature is resource poaching: a situation where all subcontractors are tied up with low-priority tasks and several high-priority contractors remain unsatisfied (Chia, Neiman and Lesser 1998).

In our previous work, we have developed methodologies and tools that assist the systematic discovery and classification of exceptions (Dellarocas and Klein 1998; Klein and Dellarocas 1999). Using these methodologies, we have performed a systematic analysis of contract net exceptions, a subset of which are listed in Figure 2.

2.2 Design of Candidate Institutions

For each exception type identified by the previous analysis, we then design one or more *handling mechanisms*, i.e., processes for anticipating it, avoiding it, detecting potential symptoms, and, finally, resolving it. This collection of processes forms the blueprint for the social institutions of that society. Figure 3 presents a partial summary of the analysis performed for a sample exception type, while Figure 2 gives a partial overview of identified handling mechanisms.

Exception Class	*Examples*	*Handling Mechanisms*
Infrastructure failures	– Agent death – Slow host – Slow links – Lost/garbled messages – Matchmaker provides false positives/negatives	– Monitor for failures and notify interested parties – Use redundancy for fault tolerance – Adaptive load balancing and mobility techniques (for avoidance/recovery) – Track reliability history of various components and make it available to interested parties
Protocol violations	– Fraudulent bid – Subcontractor advertises fraudulent skills – RFB canceled or changed – Agent skills change – Contractor refuses payment	– Monitor for protocol compliance and apply penalties to violators – Maintain reputation server and report violators – Expel repeat violators from society (by removing access privileges to matchmaker and message router)
Systemic exceptions	– No available agents due to resource poaching – No capable agents – Better bid appears after award – Better candidate appears after award – No bids from one or more good candidates – Biased bid selection	– Mechanisms depend on specific dysfunction. See Figure 3 for examples of mechanisms for coping with resource poaching.

Figure 2. Partial List of Contract Net Exception Types and Handling Mechanisms.

Exception	Resource poaching
Definition	One or more high-priority tasks are unable to access needed subcontractors because they already have been "grabbed" by lower priority tasks
Criticality	This exception can have a high fairness impact when there is a significant variation in task priority and the available subcontractors population can be oversubscribed.
Anticipation Processes	**P1**: The potential subcontractor population is currently busy with low priority tasks, and a set of high-priority tasks is expected.
Detection Processes	**P1**: The priority of the tasks that have the resources they need is less than the priority of those that do not. **P2**: A high priority contractor does not get any bids for an offered task within time-out period.
Avoidance Processes	**P1**: Require that subcontractors collect several request-for-bids before bidding, and respond preferentially to higher-priority bids.
Resolution Processes	**P1**: Instruct subcontractors to suspend lower-priority tasks and bid on later higher-priority tasks.

Figure 3. Exception Handling Rules for the "Resource Poaching" Exception Type

2.3 Experimental Evaluation

The final step in our methodology consists of performing simulation experiments in order to evaluate the effects of candidate institutions on the agent society. To facilitate the rapid evaluation of a wide variety of societies and institutions, we have developed a reusable experimental infrastructure, which consists of the following components:

- The problem solving agents, that is, the actual contractor and subcontractor agents in the case of marketplaces based on the contract net protocol.

- An *exception handling service*, which allows the rapid prototyping of a wide variety of social institutions. As described in the previous section, institutions can be specified as sets of rules for anticipating, avoiding, detecting, and resolving exceptions. The exception handling service provides a knowledge base for storing these rules. Upon initialization, the service activates a number of "sentinel" agents, which enact the anticipation and detection rules. Sentinels work by monitoring some of the communication between agents or by proactively querying agents about their status. Whenever they detect a likely symptom, they trigger the diagnostic component of the exception handling service. After the exception cause has been determined, the resolution component selects one of the resolution rules specified for this type of exception and activates "firefighter" agents in order to enact it and bring the society back to an acceptable state. The exception handling service is described in more detail in Klein and Dellarocas.

- The *SimHazard agent world exception simulator* (Shue 1999). SimHazard simulates the nodes, links, and other resources of a large-scale computer network and can be used to inject instability to the system in a controlled, robust manner. SimHazard can simulate node and link failures, slowdowns due to overload, and other effects inherent in large-scale networks, such as the Internet. Problem solving agents, as well as "sentinel" and "firefighter" agents created by the exception handling service, run on top of SimHazard. SimHazard thus provides a test-bed for studying the effects of various classes of exceptions as well as the effectiveness of the candidate social institutions in addressing them.

The purpose of the experiments is to evaluate the effectiveness of the institutions designed during the analysis phase. Our experiments consist of creating several interesting configurations of the target agent marketplace with and without social institutions, letting the agents interact, and measuring a number of dependent variables (such as efficiency, failure rate, and so on), which provide measures of the society's ability to cope effectively with an unreliable infrastructure, limited trust, and systemic dysfunctions.

3. CURRENT PROJECT STATUS

We have completed a systematic analysis of contract net protocol exceptions. We have also designed a set of social institutions (specified as rules for handling each exception type) and have added this information to our exception handling service. A comprehensive description of this analysis appears in Aryananda (1999).

Independent Variables	**Dependent Variables**
– *Scale:* # of agents and # of transactions per agent – *Infrastructure dynamism:* parameters that characterize how failure-prone the underlying computer infrastructure is – *Task complexity:* average/maximum length of contractual relationship chains needed to achieve top-level tasks; degree of variation in task priority (i.e. the extent to which task completion is "important"); time criticality (i.e. the extent to which contract satisfaction is time critical). – *Agent heterogeneity:* # of variations of contract net protocol present in the agent population; some agents have deliberately been designed to violate the protocol in various ways – *Presence of specific social institutions:* enabled vs. disabled	– *Quality of service (speed):* ratio of delay in task completion for successful tasks – *Quality of service (robustness):* ratio of outright failures – *Tunability:* task-priority-weighted versions of the speed and robustness QoS metrics above – *Overhead:* ratio between time spent on domain actions and coordination with social institutions

Figure 4. Summary of Independent and Dependent Variables of Planned Simulation Experiments

We have completed a prototype implementation of contract net agents and have specified in detail the simulation experiments to be performed. Our experiments will vary important dimensions of the agent society configuration (Figure 4). Our goal is to evaluate the effect of the various institutions on a number of dependent variables that act as measures of overall marketplace efficiency.

4. REFERENCES

Aryananda, L. "An Exception Handling Service for the Contract Net Protocol," Master's Thesis, MIT Department of Electrical Engineering and Computer Science, May 1999.

Chia, M. H.; Neiman, D. E.; and Lesser, V. R. "Poaching and Distraction in Asynchronous Agent Activities," in *Proceedings of the Third International Conference on Multi-Agent Systems*, Paris, France, July 1998, pp.88-95.

Dellarocas, C., and Klein, M. "A Knowledge-Based Approach for Handling Exceptions in Business Processes," in *Proceedings of the Eighth Workshop on Information Technologies and Systems (WITS'98)*, Helsinki, Finland, December 1998.

Jennings N. R.; Sycara K.; and Wooldridge M. "A Roadmap of Agent Research and Development," *Autonomous Agents and Multi-Agent Systems* (1:1), 1998, pp. 7-38.

Klein, M., and Dellarocas, C. "Exception Handling in Agent Systems," in *Proceedings of the Third International Conference on Autonomous Agents*, Seattle, Washington, May 1999, pp. 62-68.

Maes, P.; Guttman, R. H.; and Moukas A. "Agents that Buy and Sell," *Communications of the ACM* (42:3), March 1999, pp. 81-91.

Smith, R. G. "The Contract Net Protocol: High Level Communication and Control in a Distributed Problem Solver," *IEEE Transactions on Computers* (29:12), December 1980, pp.1104-1113.

Shue, D. "SimHazard: An Agent-world Exception Simulator," Master's Thesis, MIT Department of Electrical Engineering and Computer Science, June 1999.

Waldrop, M. "Computers Amplify Black Monday," *Science* (238), October 30 1987, pp. 602-604.

Wellman, M. P., and Wurman P. R. "Market-aware Agents for a Multiagent World," *Robotics and Autonomous Systems* (24), 1998, pp. 115-125.

Zacharia, G.; Moukas, A.; and Maes, P. "Collaborative Reputation Mechanisms in Online Marketplaces," in *Proceedings of Thirty-second Hawaii International Conference on System Sciences (HICSS-32)*, Maui, Hawaii, January 1999.