Matchmaking and Brokering

Keith Decker, Mike Williamson, and Katia Sycara
The Robotics Institute, Carnegie Mellon University
5000 Forbes Ave, Pittsburgh, PA 15213
(decker,mikew,sycara)@cs.cmu.edu

One of the basic problems facing designers of open, multi-agent systems for the Internet is the connection problem—finding the other agents who might have the information or other capabilities that you need. *Matchmaking* and *brokering* behaviors are both common in the literature. Matchmaking refers to the act of providing a capability-addressable name service or "yellow pages". A requestor is free to choose any potential server using any of several criteria. Brokering refers to the act of an agent (the broker) taking on a request, choosing a server or servers, and eventually providing the result. We show that brokers provide a much simpler load balancing mechanism, but can more easily become single points of failure, require more commitments on the part of servers, and in an open system need to work with a matchmaking/yellow pages service anyway. We have defined these behaviors in terms of simpler request actions because the semantics of requests are well-understood and allow us to easily compose hybrid systems with both matchmakers and brokers.

Performance tradeoffs: some experimental results

Our experimental results are reported using our real networked implementation of the WARREN multi-agent portfolio management system. First, a simple queuing theory model of the system is validated against the basic matchmaker and broker systems. From this it is clear that the load balancing of the brokered system confers a response time advantage over the matchmade system. We then investigate the effect of server failure and recovery on our two systems. We begin with three servers and a system overloaded for two servers. After five minutes, we kill one server, and after five more minutes, we kill another. Five minutes later, we bring one sever back, and then after ten minutes the third.

Figure 1 shows the results of this experiment. Each point represents the completion of a request. The dashed line represents the number of servers active. The response-time superiority of the brokered system stems from the difference in behavior when the failed servers come back online.

Matchmaker-based organizations can be expanded into forms such as decentralized markets. They offer the most flexible response to arbitrary agents entering and exiting the system, and each agent keeps total control over its own control decisions. However, pure matchmaker systems present a single point of failure (mitigated by local caching or multi-matchmaker structures), and each agent must be smart

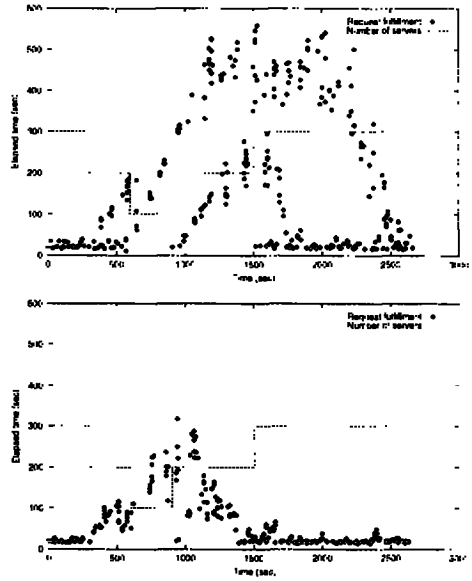

Figure 1: Effect of server failure and recovery. Top: matchmade. Bottom: brokered.

enough to construct a meta-query and evaluate the resulting alternative server choices. No easy load balancing is possible. Using a matchmaker has slightly higher overhead due to the extra queries.

Brokered organizations can become centralized markets or traditional bureaucratic managerial units. They provide easy load balancing, centralized ontological translation, or even simple integration facilities. However, they suffer from the need of agents to have static knowledge of the brokers and also form a communication bottleneck (since all requests and replies need to go through the brokers). Worst of all, they form a single point of failure that cannot be mitigated via local caching.

The solution that we are currently working on is a hybrid system with both matchmakers and brokers. By design, our specified agent behaviors work interchangeably with both organizational roles. A hybrid system allows us to capitalize on the lower overhead and efficient load balancing of a brokered system while retaining the dynamic naming capabilities and greater robustness of the matchmaker system.

A complete version of this paper with references, and several related papers, can be retrieved from the WWW at:
`http://www.cs.cmu.edu/~softagents/warren`