Configuration Management for Multi-Agent Systems

Joseph A. Giampapa
Robotics Institute, Carnegie Mellon University
Pittsburgh, PA 15213
garof@cs.cmu.edu

Octavio H. Juarez-Espinosa
Robotics Institute, Carnegie Mellon University
Pittsburgh, PA 15213
ojuarez@cs.cmu.edu

Katia P. Sycara
Robotics Institute, Carnegie Mellon University
Pittsburgh, PA 15213
katia@cs.cmu.edu

ABSTRACT

As heterogeneous distributed systems, multi-agent systems present some challenging configuration management issues. There are the problems of knowing how to allocate agents to computers, launch them on remote hosts, and once the agents have been launched, how to monitor their runtime status so as to manage computing resources effectively.

In this paper, we present the RETSINA Configuration Manager, *RECoMa*. We describe its architecture, how it uses agent infrastructure such as service discovery, to assist the multi-agent system administrator in allocating, launching, and monitoring a heterogeneous distributed agent system in a distributed and networked computing environment.[1]

1. INTRODUCTION

Managing the allocation and run-time status of even tens of software agents with different operating requirements in a heterogeneous, distributed, and networked computing environment can be a complex task. Part of the problem lies in what is called the *connection problem*[1] — finding the computer with the resources that match an agent's configuration requirements. One way of solving this problem is to use matchmaking techniques that can automatically match the description of an agent's requirements for its runtime environment with the descriptions of the resources offered by the different computers under control of the multi-agent system administrator[2, 3]. Inherent in this solution are the difficulties of anticipating which descriptions to provide, and how they should be represented. Once agents have been allocated to their host computers and launched, there is then the problem of how to monitor the runtime status of the agents, and how to monitor the constantly-changing availability of the platforms on which to run them. Doing all of the above from a single point of control and with a uniform logical interface adds further complexity to the problem.

In this paper we present the RETSINA Configuration Manager, *RECoMa*, which is our first attempt at addressing these problems of multi-agent system configuration management by matchmaking agent descriptions with the descriptions of the computer platforms on which they can be launched and executed. Through the use of an agent control language, agents and RECoMa are able to dialogue so as to provide the multi-agent system administrator fast and accurate knowledge of the runtime status of the MAS under his administration. And by configuring computers to load the RECoMa launcher every time they startup, the administrator can know, in real time, the availability and description of the computers on which he can allocate, launch and run agents.

2. RECOMA ARCHITECTURE

The RETSINA Configuration Manager is organized in a client-server architecture. RECoMa servers are installed one per computer and are automatically launched whenever the computer is started. The server provides a profile that describes its host computer's resources, and receives requests to launch agents with platform-dependent operating parameters. The profiles represent features of the execution environment that could be requirements for agent execution, such as available memory, the versions and paths of the interpreters and virtual machines, graphics libraries, or even the type and resolution of the screen. The profiles are encoded in XML so as to permit a structured representation of the data and so that fields may be added, deleted, renamed, or restructured, as needed.

When the RECoMa server receives a request to launch an agent from the RECoMa client, the request arrives in the form of an XML environment description. The environment description contains the definition of environment variables, launch parameters, and the command line specification for how to launch the agent on that computer. The server translates this XML description into a platform-specific execution file, such as a shell script or batch file, which it then uses to spawn a subshell from which the agent is launched.

The RECoMa client provides the multi-agent system administrator with a single point of control for the whole, distributed system. The client provides profiles that describe

[1]The authors would like to acknowledge the significant contributions of Matthew W. Easterday in his earlier and exhaustive implementations of configuration management programs and CM design proposals. This research has been sponsored in part by DARPA Grant F-30602-98-2-0138 and the Office of Naval Research Grant N-00014-96-16-1-1222.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
AGENTS'01, May 28-June 1, 2001, Montréal, Quebec, Canada.
Copyright 2001 ACM 1-58113-326-X/01/0005 ...$5.00.

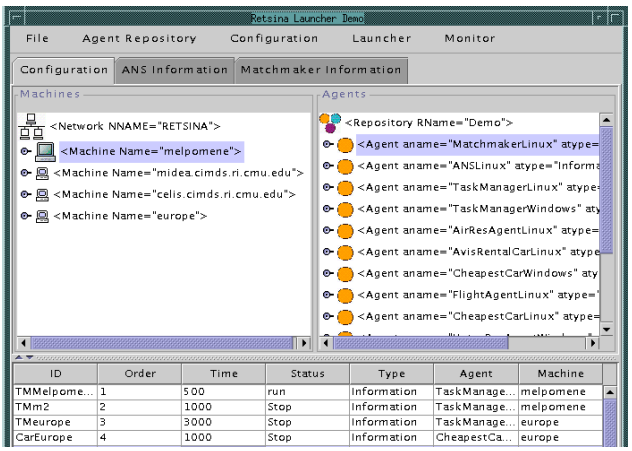

Figure 1: The RECoMa Graphic Interface

all of the agents known to him, that he may or may not be able to launch on a target machine. Although the agent profiles represent requirements for host platforms, they do not contain the specific launch environment descriptions, such as path setup, environment variables, and command lines. Those are generated when the client matches and merges the host platform description with the agent profile description.

Figure 1 illustrates the RECoMa client interface. The top left box shows the host profiles and the top right box shows the agent profiles. Each profile entry can be expanded to reveal more detail about the host platform or agent. When the user selects a computer, those agents which can be launched on it are represented by a larger icon. Similarly, when a user selects an agent profile, the RECoMa client shows the computers to which it may be assigned as a larger icon. Once the user assigns an agent to a computer, the assignment appears in the bottom box of the figure, which shows the agent-computer allocation and runtime status. To facilitate the launching of non-RETSINA agents and non-agent components, such as a web browser, we also added launch "order" and launch wait "time" parameters to the table.

The RECoMa clients discover the RECoMa servers via the SSDP service discovery protocol. This dynamic RECoMa server discovery process keeps the clients aware of the current availability of agent platforms. While not a part of the RECoMa system, per se, having the service discovery protocol part of all RETSINA agents dramatically simplifies their management. This is because agent services, such as log facilities, Agent Name Services, and Matchmakers, can be launched in any order relative to each other and other agents. Figure 2 illustrates one way in which agent discovery can occur.

3. CONCLUSIONS

RECoMa has been applied to manage an agent application consisting of 30 components on 8 computers: RETSINA agents, non-RETSINA agents, and non-agent applications. This work is significant for a few reasons. It is possible to reduce some of the overhead in matching agents to computers and vice-versa by using matchmaking techniques to match agent resource descriptions with profiles that describe the computer platforms on which they may be launched. This matching technique tolerates arbitrary descriptions of

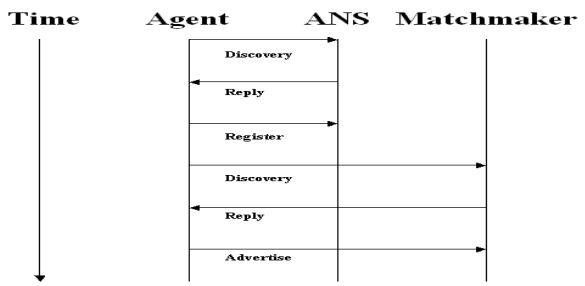

Figure 2: Infrastructure Discovery

agent requirements and profile descriptions, so that either can change *ad hoc* and over time.

We show that agent discovery is helpful to MAS configuration management in three ways. First, it permits the MAS administrator to maintain up-to-date knowledge about which computing platforms are available for running agents, as the computers become available or go off-line. Second, it reduces the number of launch-time parameters that must be passed to the agents by allowing the agents to discover their infrastructure resources, such as Agent Name Services and Matchmakers. A third way in which agent discovery helps CM is that it eliminates the need for launch-order dependencies. An agent can start and wait for other agents that it depends on during runtime execution, without crashing. Needless to say, this feature also renders an agent system more robust to failures or crashes.

A final significant feature of our work involves the human interface to MAS CM. The RECoMa interface makes it easy for the user to visually identify the computers to which he can assign agents, and for identifying the agents that can be run on a computer. The interface also makes it easy for the user to identify the runtime status of the agents, and offers commands so that the agents can be restarted, shutdown, or suspended. Most importantly, a heterogeneous multi-agent system, composed of agents and non-agents written in different computer languages, can be controlled from a single interface, irrespective of the operating system and remote execution environment of the host computer.

As interest in multi-agent systems increases, the resolution of the heterogeneous MAS configuration management problem will also become more critical. We feel that RECoMa is the right model for beginning to address such issues.

4. REFERENCES

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

[2] K. Sycara, K. Decker, and M. Williamson. Middle-agents for the internet. In *Proceedings of International Joint Conferences on Artificial Intelligence (IJCAI-97)*. IJCAI, Inc., 23-29 August 1997.

[3] K. Sycara, M. Klusch, S. Widoff, and J. Lu. Dynamic service matchmaking among agents in open information environments. *Journal ACM SIGMOD Record (Special Issue on Semantic Interoperability in Global Information Systems), A. Ouksel, A. Sheth (Eds.)*, 28(1):47–53, March 1999.