Contents

Session I

This is a techical session

FLEXIBLE SOFTWARE AGENTS FOR THE AUTOMATIC PROVISION OF PVCS IN ATM NETWORKS

Morsy M. Cheikhrouhou
Pierre O. Conti
Jacques Labetoulle

Institut Eurécom, Corporate Communications Dept.
BP 193, 06904 Sophia-Antipolis Cedex - France
{Morsy.Cheikhrouhou | Pierre.Conti | Jacques.Labetoulle}@eurecom.fr

Abstract: This paper describes an agent-based approach to the automatic provision of Permanent Virtual Channels (PVC) in ATM networks. The agent framework described fosters flexibility and seamless evolution by the definition of capability skills. Skills are plugged into the agent's brain which is responsible for the coordination of the agent's behavior. The application of this agent framework to the problem of PVC configuration resulted in the definition of a set of skills, among which, only one skill is device-dependent. Therefore, the implemented application can easily support new types of ATM switches. Moreover, it is efficient in terms of performance and bandwidth saving.

Keywords: Distributed Software Agents, Distributed Processing, ATM

1 INTRODUCTION

Currently, there is no standard protocol to automate the provision of Permanent Virtual Channels (PVC) in ATM networks [5]. Moreover, PVC provision is a tedious task. The establishment of a PVC involves many parameters that must be carefully selected and configured. For example, the Virtual Channel Identifier (VCI) used at the output of a switch must be the same as that used at the input of the next switch on the PVC route. Otherwise, cells will be lost. Also, if the Usage Parameter Control is not configured with exactly the same values on every switch, the traffic may be affected in a way that is hardly diagnosable. The problem is even more complex in heterogeneous ATM environments, since the management interfaces offered by different ATM switch providers are often not compatible. Therefore, a management application that automates the management of PVCs would be of a great use for ATM network operators.

In our paper, we present an approach based on the use of flexible software agents [4, 6, 2] which are a new rapidly evolving paradigm for developing software applications [3]. Within the DIANA (Distributed Intelligent Agents for Network Administration) project , we are developing an agent framework for Network Management (NM). Our architecture aims to foster agent flexibility and ease of development with a modular approach.

In the following section, we describe the agent framework and its concepts. Section 3 presents how the case study of PVC configuration is designed using our agent framework. Implementation related results and result assessment is presented in section 4. Finally, we conclude the paper with remarks and future directions.

2 DIANA AGENT ARCHITECTURE

Our agent architecture aims at providing flexible and dynamic software agents. Agents are wanted to be able to acquire new capabilities and skills seamlessly without interrupting their operation. This is essential for network management purposes where network elements may need to be upgraded frequently and therefore, the management application needs to be easily adaptable. For these reasons, our agent architecture is based on two major component types: the Brain and the skills. Skills provide the agent with capabilities and behaviors, while the Brain is the “headmaster” that accepts and manages agent skills.

The Agent’s Brain. The Brain (Figure 1) offers basic and innate facilities necessary for the agent operation. These facilities are either local facilities, i.e. for the agent’s local operation; or inter-agent facilities, i.e. responsible for communications and social interactions with the other agents.

Locally, the Brain is responsible for maintaining the agent’s *information database*. This holds network management information as well as information about the other agents and about the agent itself. The agent’s information can be accessed concurrently during its operation. The *information manager* ensures the coherent access to the information database and maintains its integrity.

The Brain is also responsible for the management of the agent’s skills. Skills can be downloaded on-the-fly and integrated into the agent inside its *skill base*. Skills that are no more used are disposed off so that to keep the agent as small as possible in size. Newly loaded skills may require pre-requisite skills, and the Brain is responsible for checking whether or not these skills are available and are already loaded into the agent. The brain may fetch skills either on the local machine, or on other machines with agents running on.

Moreover, the brain acts as an information and request broker for the running skills. Skills make use of the agent’s information database by queuing or updating pieces of information. A skill operation may depend on the information maintained or generated

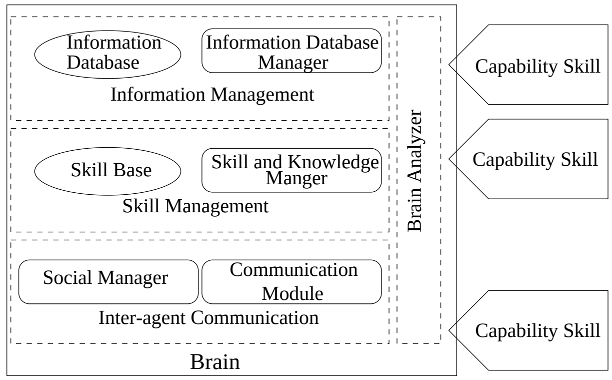

Figure 1 DIANA agent architecture

by other skills, and the Brain is therefore in charge of dispatching asynchronously this information and its updates to the interested skills. These facilities are provided by the *knowledge manager* which holds the necessary information about the skills in the *skill base*.

Finally, the *brain analyzer* is responsible for the parsing of the messages that the Brain receives, either from the skills or from the inter-agent communication. The inter-agent communication facilities allow skills from different agents to interact in a transparent way. A *communication module* allows to send to and receive requests from the other agents. Another module, the *social manager*, holds information about the other agents, such as the hosts on which they run as well as their addresses. Therefore, skills only deal with the symbolic names of the distant agents they want to interact with, and they are not aware of distribution-related details.

The Capability Skills. An agent skill is a piece of software specialized in a network management area and can be plugged-in dynamically into the agent to enrich it with a new capability. It offers new services and more elaborated pieces of information to other skills. For this, it may use the information and the services offered by other skills that are supposed to operate at a lower level.

3 AGENT DESIGN FOR PVC PROVISION

The design of the agent system was performed in two phases. In the *macro design* phase, we identified the required agent roles. We identified two main roles: the *User Agent* (UA) role and the *Switch Agent* (SA) role. The UA is in charge of perceiving the user requirement for a connection to a remote site with a determined Quality of Service. It may also capture the user constraints such as the price range he prefers or imposes for the connection billing. The UA may then negotiate these requirements with the ATM network provider. The latter is represented by Switch Agents. The switch agents are located directly on the ATM switches, hence their appellation. They accept PVC requests from different UAs and do their best to establish them. When a switch agent receives a connection demand from a UA, it will be the responsible for the establishment of the corresponding PVC. *Relatively to a particular PVC request*, we call such a SA the *Master Agent*. Indeed, it will be up to it to globally coordinate with the other SAs the action sequences that should be taken for the PVC creation. These other SAs are then called *Slave Agents*. A Switch Agent may therefore have different roles according to whether it is the responsible for a PVC creation or not.

In the *micro design* phase, we identified the different skills needed to implement the defined agent roles and we established the interactions between them. The UA role could be implemented using two skills. The *Contract Negotiation Skill* is responsible for sending PVC requests to the SA and negotiating the service contract as well as the price. The *User Interface Skill* is responsible for capturing user requests for a connection establishment and to formulate them for the Contract Negotiation Skill.

The Switch Agent role was ensured by four skills: the *Switch Skill*, the *Slave Skill*, the *Master Skill* and the *Topology Skill*. The *Switch Skill* provides a logical and common view of the ATM switch. It allows to access the switch via a uniform interface. It ensures the translation of this interface to proprietary management commands that are dependent on the switch specifics. However, this should be a temporary situation until standard ATM management MIBs (e.g. [1]) are deployed in the future ATM devices.

The *Slave Skill* is responsible for the local configuration operations that create or delete a PVC fragment on a switch. For example, it is up to the slave skill to decide whether to create a new local VP in order to convey the PVC within, or to use an already existing VP with sufficient available bandwidth. Also, it determines which VPI/VCI couples are to be assigned to the newly created VPs and PVCs.

The *Master Skill* is responsible for the global supervision of the PVC establishment. Once a physical end-to-end route is found between the source and destination end-systems, the master skill contacts the slave switch agents on that route in order to ask them to perform the necessary operations to create the PVC. It is also responsible for handling creation errors that might occur on switches.

The *Topology Skill* helps the master skill to identify a physical route between the source and the destination. The physical route identifies which switches must be traversed by the PVC and via which ports.

Running the system for the creation of a sample end-to-end PVC lead to the interaction scenario described in Figure 2 that shows the interactions between a user agent (on the left) and two switch agents (symbolically identified by “baltazar” and “douchka”). The PVC establishment requires three major steps that are coordinated by the master skill.

Finding a physical route. The master skill queries the topology skill for a physical route that links the source to the destination. The interaction between the master and topology skills is performed through the second and the third messages in Figure 2.

PVC reservation. In this phase, the master skill asks the slave skills on the switch agents (including its own agent) to reserve the PVC (messages 4 and 5). Each slave

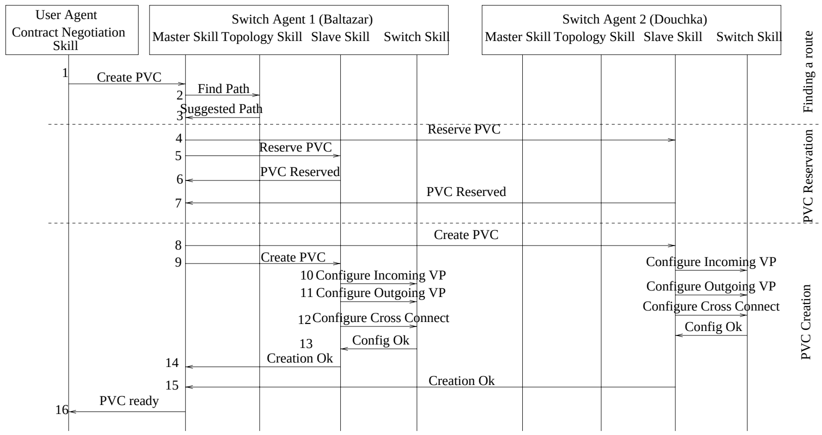

Figure 2 PVC Creation Process

agent then determines whether it is possible or not to accept the PVC (messages 6 and 7). If it can be accepted, then all the parameters (e.g. VC and VP identifiers) of the PVC are determined at this phase.

PVC creation. If the SA that is responsible for the global creation of the PVC receives positive acknowledgments from the other SAs, then the PVC can be effectively created. Again, the master skill sends creation commitments to the switch agents where the slave skills execute the commitment using the services of the switch skill.

Finally, when all the creation positive acknowledgments are received, the master skill can send back a message to the UA indicating that the PVC is now created and can be used to communicate with the remote host (message 16).

4 RESULTS

We have deployed the developed agent system on an experimental ATM network (Figure 3) which is composed of two FORE switches (FORE Runner LE and FORE ASX 200 switches).

The experiment showed the gain we obtain on the cost of management traffic compared to a centralized approach. If a centralized approach was used, all the SNMP requests and responses would be transmitted on the network, between the management station and the ATM switches. In our implementation, only high-level agent communications are exchanged via the network. All the many SNMP primitives are performed locally on the switch.

This makes our agent-based solution scalable. There is no apparent bottleneck for the processing capacity since every switch is running its own agent, and there is no bandwidth bottleneck since there is no central control of the system and the management traffic is distributed throughout the ATM network.

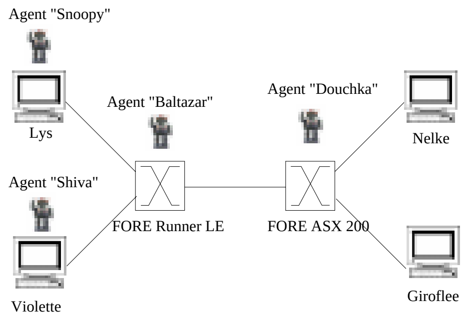

Figure 3 Experimental ATM network and Agent distribution

The only part that is switch-dependent in the whole application is the switch skill. All the other skills are directly portable. This allows to integrate new kinds of ATM devices easily. Moreover, the agent is able to dynamically integrate new skills seamlessly during its operation. Therefore, the network operator need not stop the agent system in the case of a management software upgrade in ATM switches.

Finally, our approach has an interesting property of graceful degradation. The agent system can operate even though one of the switch agents crashes for some reason or another.

References

[1] M. Ahmed and K. Tesink. Definitions of managed objects for atm management version 8.0 using smiv2. RFC1695, August 1994. ftp://ftp.nus.sg/pub/docs/rfc/rfc1695.txt.

[2] Morsy Cheikhrouhou, Pierre Conti, Raul Texeira Oliveria, and Jacques Labetoulle. Intelligent agents in network management, a state of the art. *Networking and Information Systems*, 1(1):9–38, 1998.

[3] Nicholas R. Jennings and Michael J. Wooldridge. *Agent Technology, Foundations, Applications and Markets*, chapter Application of Intelligent Agents. Springer Computer Science, February 1998.

[4] Hyacinth S. Nwana. Software agents: An overview. *Knowledge Engineering Review*, 11(3):205–244, October/November 1996. Available at http://www.cs.umbc.edu/agents/introduction/ao/.

[5] Bernard Pagurek, Yanrong Li, Andrzej Bieszczad, and Gatot Susilo. Network configuration management in heterogeneous atm environments. In Sahin Albayrak and Francisco J. Garijo, editors, *Intelligent Agents for Telecommunication Applications - IATA'98*, number 1437 in Lecture Notes in Artificial Intelligence, Paris, France, July 1998.

[6] Michael Wooldridge and Nicholas R. Jennings. Intelligent Agents: Theory and Practice. *Knowledge Engineering Review*, 10(2):115–152, 1995.