Experiments with Virtual Network Management Based on Ontology

Carlos R. Senna, Daniel M. Batista, Edmundo R. M. Madeira, and Nelson L. S. da Fonseca
Institute of Computing, University of Campinas - UNICAMP
P.O. Box 6196, Campinas, São Paulo, Brazil
{crsenna, batista, edmundo, nfonseca}@ic.unicamp.br

Abstract — **This paper presents an agent-based platform to support the development of a management system for the network architecture proposed by the Horizon Project. The platform consists of a substrate network, software for creating virtual networks and a set of agents that support ontology to enable the development of a knowledge plan. We also show the prototype built to evaluate a simple ontology for the exchange of information between agents.**

Index Terms — **Network Virtualization, Intelligent agent.**

I. Introduction

One of the goals of the Horizon Project is to propose an automatic pilot with intelligent mechanisms able to understand the context of the network and provide the necessary protocol adaptations [1]. Support for this pluralism is made by an intelligent architecture based on a pilot plan that allows the network elements to have their own vision and to take decisions for network optimization. In this context, it is proposed that the autonomy-oriented architecture should be based on isolated virtual domains running its own protocol stack. This paper presents an agent-based platform to support the development of a management system for network architecture towards the objectives of the Horizon Project.

II. Platform Architecture

The platform consists of a network substrate, a set of software tools for creating on-demand virtual network and a set of agent-based tools that support ontology. The following subsections describe the main components of the platform.

A. *Substrate Network and Software Tools for Network Virtualization*

The first step to enable the execution of agents is to construct an infrastructure that enables the creation and removal of virtual networks. We decided to adapt a setting that could be implemented on real machines with no need to modify the existing configuration. Thus, we created virtual machines and networks on the operating system Debian GNU / Linux version squeeze using the free software `qemu` [2] together with the networking tools `bridge-utils` and `iproute`. The `qemu` uses special kernel modules that enable that a set of instructions, which can be virtualized and carried out directly on the real processor. Moreover, unlike other platforms such as `VMWare` [3] or `VirtualBox` [4], the access to the console of `qemu` is much lighter, allowing virtual machines to be managed more flexibly through open and standardized remote protocols. As the real machines, virtual machines also have the operating system Debian GNU / Linux version squeeze installed. Figure 1 summarizes the configuration of virtual networks that were created to serve as a prototype for testing the system agents.

In the prototype created to evaluate the platform, A, B, C,

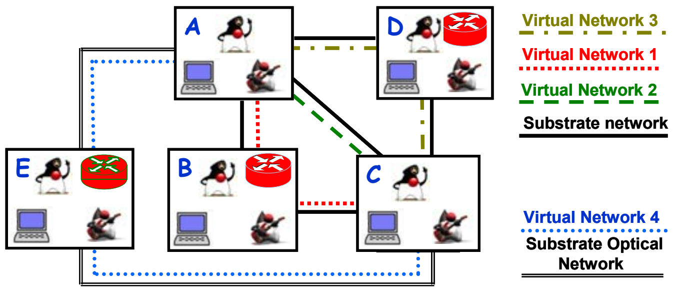

Figure 1. The Virtual Networks Prototype.

and D are real computers located in the same local network. Each of these computers have instantiated a virtual machine that can play the role of both end node of the network (as in computers A and C) or network router (as in computers B and D). Virtual networks are created defining routes and virtual network interfaces on the virtual machines as well as on the real machines. Each virtual network has an interface that allows control by the agents. The creation and removal of virtual interfaces, routes and instantiation of virtual machines are made by scripts that have been implemented by the authors. These scripts allow agents to obtain information about the network and act in a transparent manner. As noted in Figure 1, there is a fifth node (Computer E) defined in the configuration of networks. It is planned that this computer, which is located remotely to the local network via a dedicated optical connection, to be integrated on the virtual networks in the future.

B. *Agents with Ontology Support*

To support the development of network management that uses the concept of virtual domains with decisions supported by the vision of each network element, we use agents with different levels of functionality. Agents are responsible for obtaining information from the links, and for sharing them with your neighbours in order to implement local actions necessary for the proper functioning of the network. All actions taken by agents are based on ontology. There are two groups of agents: informers and managers. Informer group has three types of agents, which the main function is to keep the information updated on the node and notify the local manager. For each link of a node, a *link-informer-agent* agent is instantiated. This kind of agent obtains basic information from the link, such as capacity and available bandwidth. The *router-inform-agent* and the *switch-inform-agent* monitor the nodes in compliance with their functionalities. The features are implemented through appropriate ontology for each feature. The agent of the type manager (*meta-agent*) is responsible for receiving information from informer agents, for organizing them and taking the necessary actions according to the QoS requirements for the virtual networks. The actions taken by staff managers are also modelled through ontology. To implement the agents, we used the Java Agent Development Framework (JADE) [5]. JADE is a software environment to build agent systems for the management of networked information resources in compliance with the FIPA specifications for interoperable multi-agent systems. In addition, JADE provides a good support to the use of ontology. If agents are to communicate in a way that makes sense for them, they must share the same language, vocabulary and protocols. By following FIPA standards [6], JADE already supports a certain degree of commonality, this is evident in the use of FIPA communicative acts and the Coder/Decoder classes for SLn languages which determine the form of the messages exchanged between agents. In our prototype, shown in Figure 1, we see one *router-informer-agents* on nodes B, D and E. The nodes B, D and E have three *link-informer-agents*, while nodes A and C have four *link-informer-agents*. All nodes have one *meta-agent*.

III. A Brief Case

We carried out preliminary experiments with virtual networks composed of the computers A, B and C of Figure 1. In these experiments, information about the available bandwidth between virtual machines were obtained by scripts that executed a modified version of the program `pathload` [7]. The `pathload` is a free software written in C that estimates the available bandwidth by measuring the variation of the delay in a route (OWD). The code of the tool was modified by the authors so that it could run concurrently on the same computer. Currently, the program is active in all the virtual machines in virtual networks. Scripts developed by the authors run every minute in order to obtain information about the availability of virtual links. This information is then captured by the agents and used for decision making. It is important to note that the infrastructure built to the prototype can be modified without the need to rewrite the code of agents. This is possible by the existence of scripts that have been implemented and that provide information to the agents properly.

It is also important to note that during the experiments, we observed that the virtual machines show significant performance degradation when users run heavy I/O bound processes in the real machines (e.g., experiments simulating high-speed networks). These facts justify the use of virtualization solutions that ensure the access of virtual machines directly to the actual hardware of computers, thus preventing the scheduler of the host operating system to be unfair" with the virtual machines.

IV. Conclusion and Future Work

We show a platform to support the development of network management within the objectives of the Horizon Project. The platform consists of a substrate network, tools for creating virtual networks and software agents, to experience-based management strategies implemented in virtual domains described by ontology. Routing changes due to variations in available bandwidth along the paths, creation of a new route to account failures in a network link, experiments with different switches, insertion of delay as an additional attribute to the virtual links and virtual network routers are examples of experiments that can be done with the platform. As future work we intend to offer an alternative to ontology, by using a Rule Engine as a part of our platform.

Acknowledgment

This work was supported by FINEP, FUNTTEL, CNPq, and CAPES.

References

[1] Horizon Project: A New Horizon to The Internet, http://www.gta.ufrj.br/horizon/.

[2] Bellard, F., Brook, P., et al. (2010) Qemu: open source processor emulator. http://wiki.qemu.org/Main_Page (Accessed at 10 Mar 2010).

[3] VMware. (2010) VMware Virtualization Software for Desktops, Servers & Virtual Machines for a Private Cloud. http://www.vmware.com/ (Accessed at 10 Mar 2010).

[4] Oracle. (2010) VirtualBox. http://www.virtualbox.org/ (Accessed at 10 Mar 2010)

[5] F. Bellifemine, G. Caire, A. Poggi, G. Rimassa, "JADE: A White Paper" in EXP in search of innovation, Vol. 3, No. 3. (2003), pp. 6-19.

[6] Foundations of Intelligent Physical Agents (FIPA), FIPA Agent Management Specification, www.fipa.org/.

[7] Jain, M., Dovrolis, C. (2003) End-to-end available bandwidth: Measurement methodology, dynamics, and relation with TCP throughput. IEEE/ACM Trans Netw (TON) 11(4):537–549.