Innovative Technologies for the Management of Large-Scale Computer Networks

Marcos Dias de Assunção [1] , Fernando Luiz Koch [2] , Carlos Becker Westphall [2] , Tom Spindola [2] , Edison Alessandro Xavier [2]

[1]CETEC - Technological Center
UNOESC - University of Western Santa Catarina State, SC, Brazil

[2]Network and Management Laboratory
Federal University of Santa Catarina, SC, Brazil

{assuncao,koch,westphal,tom,xavier}@lrg.ufsc.br

Abstract. *In this paper we present our research on the use of agents-based computing, mobile code and grid computing for the problem of scalability in computer network management systems.*

1. Introduction

The management of large networks requires computational resources beyond the available in traditional environments. Decentralized network management is a possible solution and we have explored the *Distributed Artificial Intelligence* approach in our previous work [Koch and Westphall, 2001].

This paper presents our research on innovative solutions for the problem of scalability in network management systems. We are exploring two main distinct solutions: (i) the use of *mobile agents* architectures [Xavier et al., 2003], by tentative and analysis of different configurations, and; (ii) *grid computing*[Assuncao et al., 2004], where the development of a *grid of agents* provides the infrastructure to build highly distributed, agent-based network management solutions.

In section 2, we motivate the use of agent-based computing for the problem of network management. Section 3 describes the *mobile agent* approach and results and section 4 describes the use of grid computing. Section 5 ends the paper with the conclusions.

2. Motivation

The use of Distributed Artificial Intelligence (DAI) (see [Koch and Westphall, 2001]) proposes an alternative for the server-centric scenario imposed by traditional network management architectures (such as SNMP). The motivation for the development of multi-agent based network management solutions is related to the ability of provisioning: (i) **high degree of adaptability**, which is inherent to the agent technology, being one agent an environment aware and responsive piece of software; (ii) **code mobility**, as the self-contained agents represent a simple abstraction for software move between element; (iii) **module reusability**, as each agent can implement a module function and multiple agents interact during problem resolution, and; (iv) **self-generation**, due to the agent's self-contained features it is easier to implement agents that create new agents customized for specific jobs.

In the early days of our research, we have tried to create multi-agent software applications composed by autonomous agents loaded with the rules and capabilities for

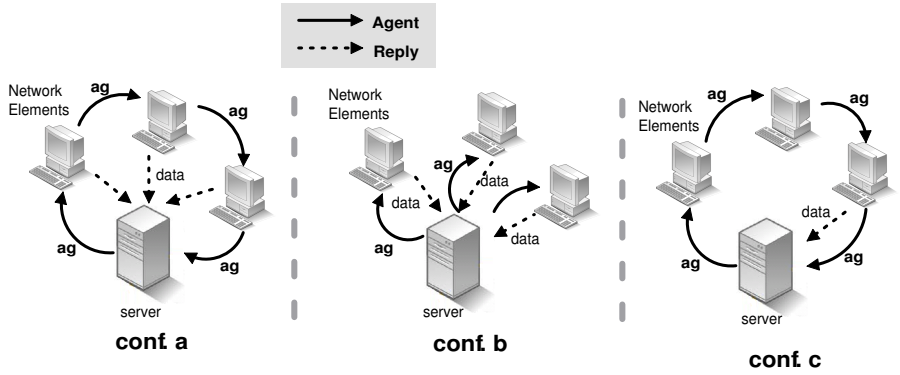

Figure 1: Mobile Agent Configurations in Network Management Solutions

each of the phases of the network management process: data collection, store, analysis and report generation. Although we had initial success and could produce several small-scale solutions, the approach started to present problem of complexity and scalability when applied to larger scenarios. Hence, we went back to research board and started to look for *alternative* ways of applying DAI to the network management problem.

In the next sections, we present the evaluation of *mobile agent configurations* and the implementation of *grid of agents*. These are separated efforts that contribute to the problem in different ways.

3. Mobile Agents Approach

The use of *mobile agents* contributes to the problem of network management by providing a method to build distributed agent-based applications for the network management task. There are several possible distribution configurations in this scenario and the code displacement heavily impacts on the solution's results. In [Xavier et al., 2003], we proved that different configurations present distinct metrics and proposed a method to analyse these results. Figure 1 presents three possible configurations, which are described below.

- **(conf. (a))**: the agent is created in the *manager station* and dispatched to the first network element, where it collects data, executes designated actions and, then, forwards the collected data back to the manager station. Next, the agent migrates to the next network element and the process repeats.
- **(conf. (b))**: the *manager station* dispatches one agent to each network element, where it collects data, executes designated actions and replies to manager station.
- **(conf. (c))**: the agent is created in the *manager station* and dispatched to the first network element, where it collects data, executes designated actions and stores the collected data in the body of the agent. The agent migrates to the next network element and the process repeats till the last visit, when the agent returns back to the manager station carrying all the collected data in its body.

We compared the behavior of these configurations, taking into account the *number of transmitted bytes* (number of bytes transferred through the network during the operation). We concluded that the **conf. (a)** has the best behavior when intermediate results are welcomed, combined with small amount of transmitted bytes. The **conf. (b)** resulted in a larger amount of data transmissions with no significative performance or solution robustness improvement, since it uses the same request/response model as the traditional network management model (e.g. SNMP). The **conf. (c)** has the best behavior in an environment with instable or high-latency link between the manager and network elements, since the replies to the manager happen just once, at end of the management.

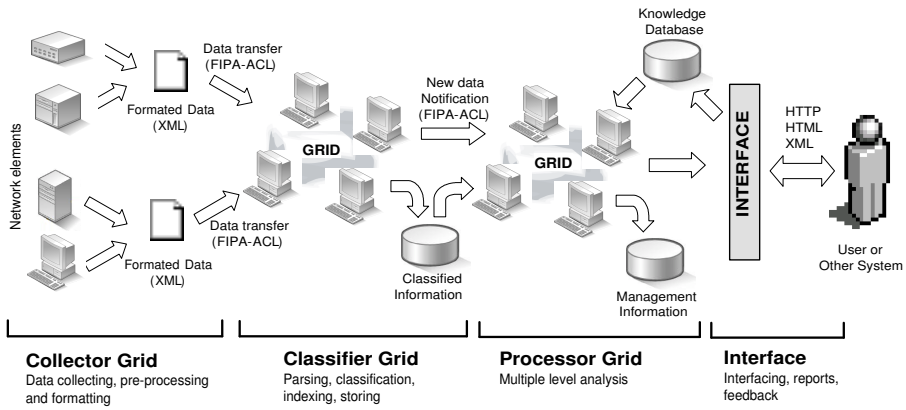

Figure 2: Grid Computing Architecture for Network Management Solution

4. Grid Computing Approach

Grid computing contributes in provisioning the computing resources for the processing of large amounts of data. Hence, *grid computing* is applicable to the problem of network management to provide the infrastructure to process large batches of collected data into compiled management information [Assuncao et al., 2004]. We used agent-based computing to implement autonomous grids nodes, leveraging on its ability of local interaction and coordination.

Figure 2 presents our architecture, which is is based on the four stages of the traditional management workflow [Koch and Westphall, 2001]: *data collection*, *data classification*, *inference* and *report presentation*. We grouped each functionality in a different *grid group* that does not represent individual grids in the system but, rather, they are group of grid nodes that share similar functionality. The components of our architecture are:

1. **collector grid**, for *data collection* stage, that extracts data from the network devices. The data is represented in a standard internal format (XML) and transmitted to the classification grid for further processing.
2. **classifier grid**, works the *data classification* stage, where the collected data is grouped per affinity. The classified data is stored in the *classified data* database. Once new data arrives, a notification is triggered to the processing grid element for further processing.
3. **processing grid** works the *inference* stage, where the data sets are analysed and compiled them into management information. The analysis follows a set of rules, stored in the *knowledge database*, and the results are stored locally and delivered to the interface grid component.
4. **interface grid**, works the *report presentation* stage, by implementing the external interface for the generated management reports. We opted for ubiquitous interfaces based on HTML, XML, and HTTP standards.

4.1. Load Balance in Grid of Agents

The grid computing architecture requires a method to distribute the processing workload throughout the components of the grid. Due space limitation we do not describe the load balancing model in detail. We refer to the description presented in [Spindola et al., 2004].

We developed two load balance methods. The *first method* consists of a central scheduler (CE), which performs resource allocations based on information collected from the resources. We set up four events that trigger resource load information collection: (i) when the resource registers itself in the grid; (ii) when a task is assigned to a particular

resource; (iii) in time intervals; (iv) when a event considerably impacts the load upon the resource. The shortcoming of this first method is the centralization of the scheduling decision. In the *second method*, the CE is responsible for the allocation, but the resource availability information is supplied directly by active agents executing on the grid node.

The normal operation is to have the arriving tasks allocated by the *grid root node* to other nodes with available resources. However, execution exceptions might occur and the solution must be equipped to handle them. We use two allocation policies to deal with exception situations: (i) *receiver-initiated balancing*, where a resource with low processing load acts directly by selecting the tasks that it will execute instead of waiting for the resource allocation, and; (ii) *sender-initiated balancing*, where a node with high workload initiates the process of transferring part of its load to other node.

From our laboratory tests we concluded that the mentioned methods result in the even distribution of resource utilization.

5. Conclusions

In this paper we presented our research on the use *grid computing*, *mobile agents* and *distributed artificial intelligence* in the implementation of network management solutions. In the area of mobile agents, we proposed three distribution configuration and analysed their consequences individually, presenting the results of our experiments about the mobile agent distributions. On the *grids computing* effort, we propose an architecture for the creation of *grid of agents* and described its components.

We argue that only through the distribution of the workload we can supply the infrastructure to implement *more intelligent* solutions (*computational intelligence* is indirectly related to the number of inference rules the a system can process per time). The innovative solutions proposed in this work contribute to the optimization and availability of computing resources towards that goal.

In our future efforts, we will proceed with developing enhanced prototypes of grid computing solutions. One area of interest is to determine the *cutting point* where it is an advantage to use a grid approach (in alternate to a simpler multi-agent solution). Moreover, we intend to extend our studies on load balance in grid computing and the integration of mobile agents to grid of agents scenarios.

References

Assuncao, M. D., Koch, F., and Westphall, C. B. (2004). Grids of agents for computer and telecommunication network management. *Concurrency and Computation: Practice and Experience*, 16(5):413–424.

Koch, F. and Westphall, C. B. (2001). Decentralized network management using distributed artificial intelligence. *Journal of Network and Systems Management*, 9(4):375–388.

Spindola, T., Westphall, C. B., Assuncao, M. D., and Koch, F. L. (2004). Load balance in grids of agents for computer network management. In *9th Workshop on Management and Operation of Networks And Services*, pages 117–128, Gramado, Brazil. Brazilian Computer Society. In Portuguese.

Xavier, E., Koch, F., and Westphall, C. (2003). Automations in computer network management utilizing computational intelligence. In *The Third IEEE Latin American Network Operations and Management Symposium - LANOMS*, Foz do Iguaçu, BR.