ANNALS OF THE FACULTY OF ENGINEERING HUNEDOARA
2006, Tome IV, Fascicole 1, (ISSN 1584 – 2665)
FACULTY OF ENGINEERING HUNEDOARA,
5, REVOLUTIEI, 331128, HUNEDOARA

ADMINISTRATION AND CONFIGURATION OF HETEROGENEOUS NETWORKS USING AGLETS

[1]BALAN Mihai, [2]MUSCALAGIU Ionel, [1]BALAN Mirela - Ramona

[1] DENMARK TECHNICAL UNIVERSITY,
INFORMATICS AND MATHEMATICAL MODELLING,
LYNGBY, DENMARK

[2] "POLITEHNICA" UNIVERSITY OF TIMISOARA,
FACULTY OF ENGINEERING HUNEDOARA, ROMANIA

ABSTRACT
The Mobile Agent Paradigm is the future direction of distributed systems. It offers advantages over conventional mechanisms, which move data to the code thereby wasting available network bandwidth.
Mobile agents' unique properties are well suited to the decentralized management of today's growing heterogeneous networks.
This paper explains how Aglets can be used for network management and configuration operations that allow software installation, upgrading and maintaining.
Several mobile agents are built for this purpose. They allow information retrieval and different task executions on all network nodes accordingly to the desired state of the system in a secure way.

KEYWORDS:
mobile agents, IBM Aglets, heterogeneous networks, configuration, security

1. INTRODUCTION

This paper describes the solution of a network administration and configuration aglet-based tool. The goal is to develop a tool for a network administrator to configure and maintain a network made of heterogeneous nodes in a secure way. This tool has to determine the current and desired configuration of all network nodes and perform the necessary software installations and updates.

2. MOBILE AGENTS IN NETWORK ADMINISTRATION

A mobile agent is an autonomous software object that acts in the name of user. An agent can migrate in a network from one node to another at any time during their execution, performing different tasks e.g. transporting, collecting data, queering network nodes, etc. It does not need permanent network connection. It performs its tasks on the local system where it is dispatched.

When network connection is restored, the mobile agent can continue to migrate to other hosts and perform its tasks there. **Mobile Agent Paradigm** considers that local interaction for gathering data is more efficient than the remote one as in case of the **Client-Server Paradigm**. They can be used successfully in case of the decentralized and cooperative architectures. This is because today's networks cannot longer be managed by using just centralized or highly coordinated management strategies because of the heterogeneity and size issues.

IBM Aglets technology was created at IBM research laboratory in Japan. Later on, this becomes an open source project. They are built around a peer-to-peer network with no centralized node responsible for managing the Aglets servers. Aglets can migrate between network nodes by using three strategies such as **itinerary**, **forwarding**, and **ticket**. The communication among aglets takes place by means of message passing such as: **Now-Type Messaging**, **Future-Type Messaging**, and **One-Way-Type Messaging**. [1]

3. ANALYSIS AND DESIGN OF THE SYSTEM

This section discusses different choices taken during the analysis stage of the network administration and configuration tool. It depicts the identified issues and their solutions. Reasons for using mobile code are also depicted.

3.1. SEVERAL REASONS TO USE MOBILE AGENTS FOR NETWORK MANAGEMENT APPLICATIONS

The **Mobile Agent Paradigm** provides a better approach in case of distributed network management than the **Client-Server Paradigm**. Mobile agents hide the complexity and heterogeneity of the network infrastructure; provide local interaction for communication and mobile logic facilities while the traditional approach uses local transparency.[2]

Mobile agents migrates to remote hosts, perform their tasks accordingly to the their mobile logic they incorporate and uses local interaction to access the distributed information (they migrate where the data is stored). They ensure a better use of network resources and low communication costs through the migration mechanism and minimum network traffic due to their mobility.

Mobile agents are very flexible. They improve the scalability of distributed applications. They can assure reliability and high quality of service by acting autonomously in a temporal disconnected environment. Real time interactions can be performed between mobile agents and network devices because they are moved close to the information source. Network latency can be overcome by taking advantage of alternate routes around any communication link problem. [3]

This technology has also a few drawbacks. Mobile code can introduce new types of vulnerabilities into network such as: agent-to-agent, agent-to-platform, platform-to-agent, and other-to-agent platform threats. Mobile agents can roam into the network forever without an effective mechanism to terminate the agents. Complex agents need transport support in order to transport them from one network node to another in a short period of time.

3.2. ISSUES IN CASE OF THE NETWORK MANAGEMENT TOOL

Several issues are identified in case of the network administration and configuration using an aglet – based tool, such as:

- **network topology discovery mechanism** (How can an agent discover the network topology i.e. node types and status?);
- **node information gathering mechanism** (How can the information be gathered from different nodes and protected against malicious attackers? What happens if one node is down?);
- **installation task mechanism** (How many agents are going to be dispatched to perform the required tasks? How can the installation or update files be transported and used on the remote machines? How to keep the network traffic at minimum?);
- **communication mechanism among agents** (How do the agents communicate among them in order to fulfill the required tasks and inform the administrator about the result?);
- **an efficient mechanism to terminate agents** (How can the network be protected against "ghost" agents?);
- **security concerns** (What kinds of threats one can encounter in such a system? How can the aglets and servers be protected against attackers and new vulnerabilities introduced in the network?)

3.3. PROPOSED SOLUTION

The solution to the network administration and configuration aglet-based system can be stated by considering the advantages of mobile computing in network management. The system is made of three types of agents:

The Manager Agent (MgA) - a stationary agent on the management station. It controls all the others outgoing agents, provides them with itineraries and tasks to execute. It also collects and analyzes the results.

The Data Collector Agent (DCA) - created and controlled by the MgA, travels in the network following the route stored in its itinerary (set up by the MgA), and collects the data of interest from each remote node. The collected data is reported to MgA when returns on the management station.

The Task Executor Agent (TEA) - is also created and controlled by MgA. It travels in the network according to its itinerary, and on each remote node it executes the task stored in its task list.

The manager agent controls all other working agents (DCA and TEA). It has access to a list of itineraries stored in a database as URLs pointing to all nodes in the network. The information gathering mechanism is done by using one worker agent that visits all nodes, one at a time, and collects the data. It returns the gathered data to the manager agent after visiting all nodes. The manager analysis and stores the collected data. It takes the appropriate measures in order to obtain the desired system configuration by dispatching only one agent to deal with all identical tasks and a different agent for each unique task.

The agents transport the files needed for installations to the remote nodes. Agents communicate by using asynchronous message passing mechanisms. In case of a failure in the normal functionality of an agent, the agent informs the manager and disposes itself. To run the management tasks, including installation or removal of applications i.e. to create and run processes, agents need high security privileges. To grant high privileges to an autonomous mobile code is a high-risk operation. Therefore, security schemes to authenticate and use mobile agents in a safe and secure way need to be implemented i.e. the multi-agent system must provide the following security services: **Authentication of the Sender**, **Authorization of the Agent (or Its Owner)**, **Secure Communication between Agent Systems**, and **Non-repudiation and Auditing**. [4, 5]

The diagram of the proposed solution for the network management agent based system is depicted in fig. 1. **The Manager Agent** (**MgA**) is located on the server. It has access to an **Itinerary list** that contains all nodes in the network. The first step consists in dispatching a **Data Collector Agent** (**DCA**) that collects local data, encrypts the collected data on that node, stores it in a structure in DCA, and moves to the next node.

When DCA finishes collecting data on all network nodes it returns home and hands in the collected data to the MgA that stores it on the server. MgA analysis the collected data and takes the appropriate measures. It dispatches one or many **Task Executor Agents** (**TEA**).

They receive a new itinerary from the Manager. TEA goes to the first node in the itinerary, decrypts the task list, executes the tasks, gets the tasks' output, and moves to the next node until it returns home. Once it arrives at home, it hands in the results of the tasks to the manager.

There can be several TEAs each of them performing a unique task on one or many network nodes e.g. TEA 1 performs a Windows update SB07722 task on Node 1 and n, while TEA 2 performs a MS Office update on Node i and j.

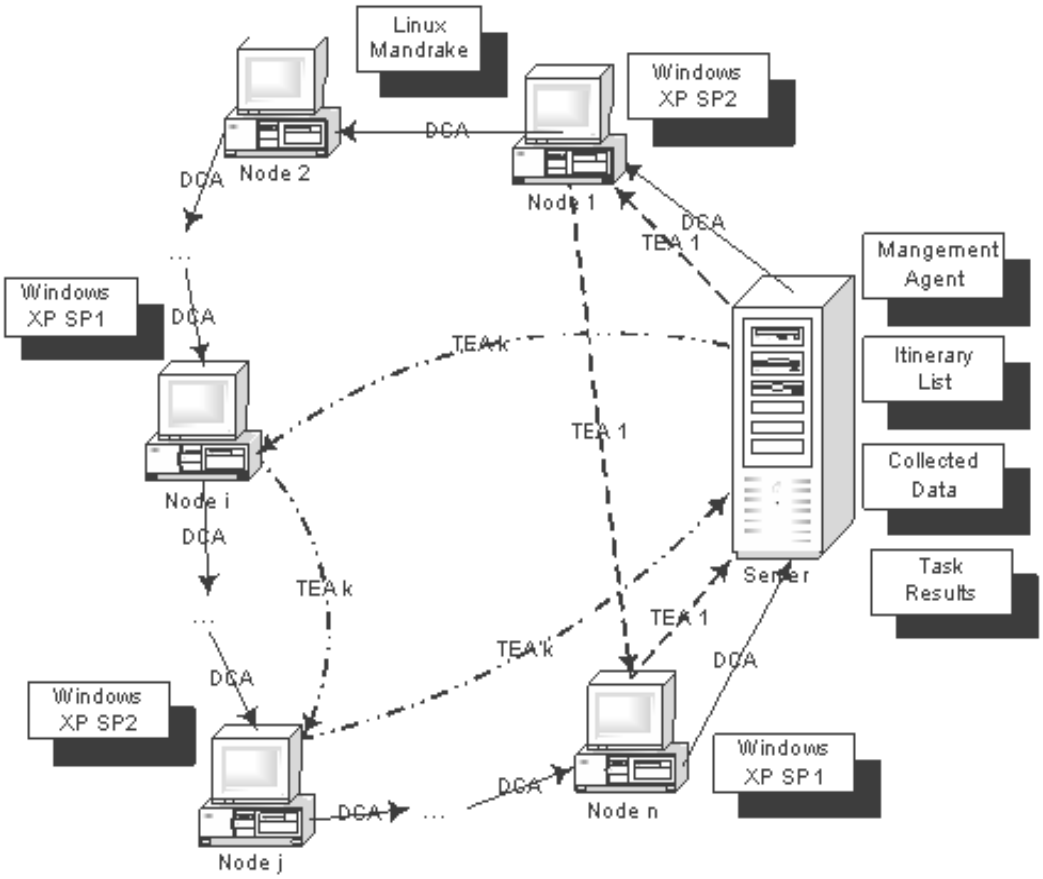

Figure 1. NETWORK ADMINISTRATION
AND CONFIGURATION AGLET - BASED SYSTEM

3.4. THE COMMUNICATION PROTOCOL OF THE AGENTS

The communication between MgA, DCA and TEA is realized by means of message passing. On the creation time TEA and DCA receive a reference to the MgA proxy. As the MgA is a stationary agent, its proxy never changes. In this way the outgoing agents are able to initiate the communication with the MgA at any time.

MgA is the creator of TEA and DCA, and it can communicate with the agents as long as they are not dispatched on the remote hosts.

When the TEA and DCA are dispatched, they will get new proxies, and the communication can take place only from the outgoing agents to manager. The new agent proxy is sent back to the manager when the agents return to the manager station. In this way the manager can take the control over the communication and send further instructions to the agents (e.g. discard the collected data). The communication protocol between MgA and DCA can be summarized in the following steps:

- The MgA creates the DCA. DCA receives the MAg proxy on the creation.
- DCA sends a **READY** message to the MgA.
- MgA replies with **OK**. At this point the communication between two parts is established and each of the parts can initiate the communication with the other one.
- MgA creates an itinerary object and signs it with its private key.
- MgA sends a **LOAD ITINERARY** message to the DCA, containing the signed itinerary.
- DCA verifies the signature of the itinerary with the public key of the manager, and sends back an **OK** message if no problem occurs.
- If the MgA receives the **OK** reply, then a **START** message is sent to the DCA.
- The DCA start dispatching itself according to its itinerary. As long as it is on remote destinations, the DCA is the only part that can initiate the communication. This takes the form of **ALERT** messages when certain problems occur (e.g. invalidation of the itinerary signature)
- When the DCA returns back on the manager station, a **DATA READY** message is sent to the MgA. The new proxy is also sent in the message arguments.
- At the right moment, the MgA sends a **DISCARD DATA** message to the DCA.
- The DCA replies with a **DATA** message that contains the encrypted collected data from the visited node.

At this point the DCA has finished its job and it can be disposed. Then the MgA processes the collected data (including decryption), and initialize the TEA. The communication between MgA and TEA is almost identical with the one between MgA and DCA. The only exception is that a new message is introduced, namely **LOAD TASK LIST**.

4. CONCLUSIONS

As nowadays networks increase in size very fast, there is a need of a reliable and de-centralized secure network management and configuration system that can manage and adapt dynamically to all changes in a network. Different solutions are analyzed and discussed. The architecture proposed in this paper is a step in this direction. Considering the initial state of the system and the desired one, a network administrator can use the system introduced in here and reach the desired state of the system by using several Java mobile agents - Aglets built during the development phase. Security issues in case of code mobility are also dealt with. An agent specialized in a secure key distribution mechanism can be also be considered part of a possible future work.

REFERENCES

[1.] FERRARI, L. (2004), The Aglets 2.0.2 User's Manual, World Wide Web. http://sourceforge.net/projects/aglets

[2.] FIPA 98 (1998) Agent Security Management Specification, Foundation for Intelligent Physical Agents, Part 10 Version 1.0, Geneva, Switzerland, World Wide Web. http://www.fipa.org
[3.] MARTIN-FLATIN, J.P., and S. ZNATY (1997) A Simple Typology of Distributed Network Management, Proceedings of the 8th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management, Sydney, Australia.
[4.] MITSURU, O., K. GUENTER, and O. KOUICHI (1998), Aglets specifications Draft 1.1, World Wide Web. http://www.trl.ibm.com/aglets/spec11.htm
[5.] MANOJ, K., and C.Z. XU (2002), Framework for Network Management using Mobile Agents, Proceedings of the 16th International Parallel and Distributed Processing Symposium.