Mathematical Problems of Computer Science 25, 2006, 64-70.

Intelligent Agent Server (NetInt) System

David A. Karapetyan

Institute for Informatics and Automation Problems of NAS of RA
e-mail david@dm-lab.sci.am

Abstract

The paper describes Intelligent Agent Server system, which employs software agents. A detailed description of software agents as well some examples of the system application, particularly intrusion detection in local networks is given.

1. Introduction

A research project **NetInt (Networked Intelligence)** has been carried out, aimed to develop an application software product to operate in computing networks. The use of software mobile and intelligent agents enables the system to solve a variety of applied problems, such as network management and maintenance, network dynamic optimization and security control.
NetInt system is an extension of SPARA (Security Policy Adaptation Reinforced through Agents) system designed within the framework of IST-12637 "Security Policy Adaptation Reinforced through Agents" project, 5th Framework Programme (FP5) European Community Framework Programme for Research, Technological Development and Demonstration, 2000-2001.

2. System architecture

The NetInt system is basically organized as follows: NetInt agent platforms also referred to as Agent servers are installed on a number of computers bounded together to organize a network. We call the computers, where the servers are installed *host*. Agent servers permit access to system resources and utilize them. Mobile agents travel from one host to another and perform the thread of execution. They may assemble and swap information, analyze it and later interchange the analysis results. In this way they try to adjust system parameters, perform system troubleshooting (detect faults in the system or its misbehaviour and take steps to eliminate them when such are found).

To be able to solve problems mentioned above software agents should posses the following characteristics:

- Autonomicy – agent ability to act by its own initiative, i.e. without meddling of other person or program;

[1] The research is supported partly by NetInt: 04-77-7173 project, http://www.intas.be and State Principal Program of Armenia on Scientific Computations.

- Mobility – agent's ability to travel within the network to search information necessary for task execution;
- Interoperability – equal possibilities to interoperate between various software agents;
- Liability – the ability of the agent to perform the thread of execution for which the agent is liable for;
- Flexibilty – ability to act in response to the changes of the execution environment

Currently two classes of software agents are specified depending on their major characteristics: mobile and intelligent agents and, respectively there are two promulgated standards OMG MASIF (The Object Management Group's Mobile Agent System Interoperability Facility) and FIPA (Foundation for Intelligent Physical Agents), which standardize the above-mentioned agent classes.

Below we'll give description of these agents in more detail.
Intelligent or stationary agent executes only in the host where it begins execution. If the agent needs information that is not on the host, it addresses to mobile agent to interact with remote hosts.

Mobile agent is not bounded to the host to which it belongs and where it begins execution. It has the ability to transfer itself from one host to another in a network. This specification is the basic and the most important with mobile agents. When a mobile agent migrates to another host, it locates itself in that host by its own initiative and interacts with the host.

When an agent travels, its state and program code are transported with it. In this context, the agent state can be either its execution state, or the agent attribute values that determine the thread of execution, resumed to perform at the remote host. The agent attribute values may also include some other values associated the agent e.g., time to live and so on.

3. Description of NetInt agent environment

NetInt agent environment is a dialogue system, implemented in Java programming language, which supports transferability of software code across any Java Virtual Machine containing operating system. The NetInt Agent environment operates within local networks. It includes 4 subenvironments:

- AgentServer
- RegistryServer
- CodeBaseServer
- Graphical interface Dam_UI and Dam Explorer

Figure 1 sketches out NetInt agent environment built up in accordance with OMG MASIF standard.

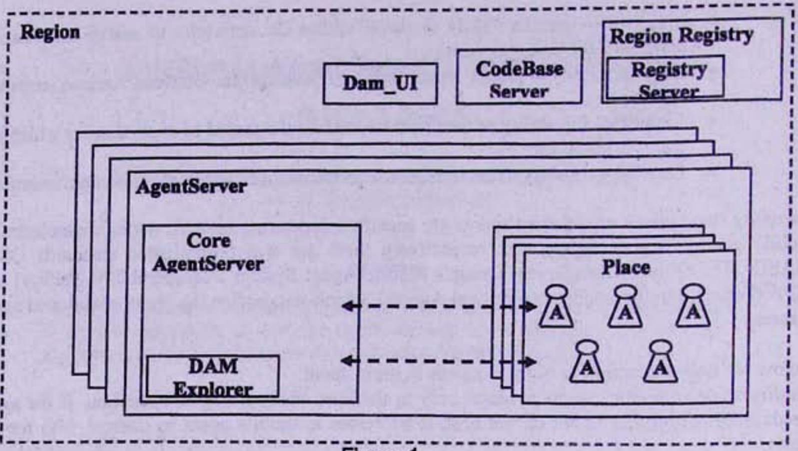

Figure 1.

Subenvironments in the systems interact via Java RMI communication facility. The latter allows a more effective usage of object-oriented programming (OOP) resources.
Figure 2 illustrates NetInt agent environment architecture in a different way.

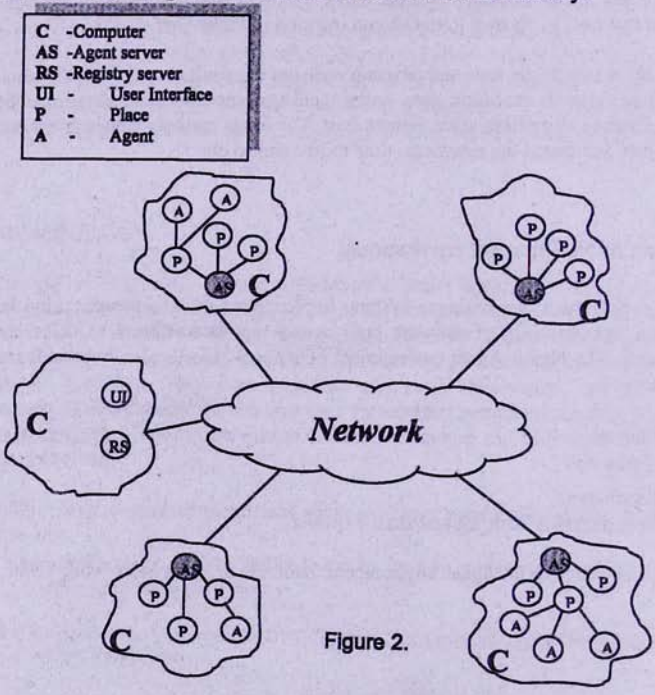

Figure 2.

As it is seen on Figure 2 an agent environment contains the following elements: computers interconnected via network, each running an Agent server (AS). An Agent server has the following architecture (see Figure 3):

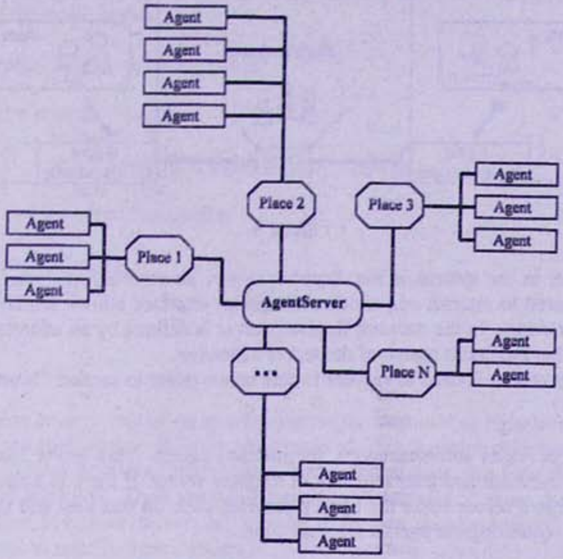

Figure 3.

Agent server is an environment that is located in a host and permits to create /terminate, transfer/receive and execute software agents. An agent can generally execute in a context of Agent server, which is called *Place.*

When loaded in a host, the Agent server registers itself in Register server that can be located on a destination host. One of the basic elements of Agent server is Place, a context in which an agent can execute. Another major element of Agent server is *Communicator*, which supports transfer of data as well as agents themselves between hosts (see Figure 4). Communicators are stationary agents. Agent communicator is the principal Communicator that provides communications services mentioned above. Admin communicator is another Communicator that plays an important part in supporting interconnecton between NetInt environment admnistrator and Agent server.

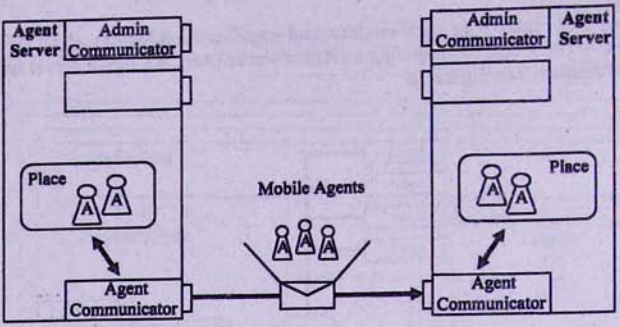

Figure 4.

An important part in the system is the *Register server*, an environment where all the Agent servers are registered to interact one another. Graphical interface allows display of information stored in Register server. In the network Register server is defined by an address of special type called *Multicast* that facilitates search of the server's location.
A special naming system is used to register in this server (refer to section "Naming system" for more information).

CodeBaseServer provides software codes, for instance agents. This server loads on a host in NetInt agent environment and later registers in Register server. If there is a need to execute an agent, then the Agent server seeks the agent's program code on that host and then addresses to CodeBaseServer requesting the agent's program code.

The NetInt environment is globally managed by Graphical interface and two elements represent its core:

- Dam_UI
- DamExplorer

Admin comunicators of Agent server provide interaction between the above-mentioned two elements of Graphical interface and Agent servers.

Dam_UI interface includes Agent servers that pemit embedding of software agents of different types into the network. Another important feature of Dam _UI is to retrieve information on the state of the whole system and display it graphicaly in its own environment.

DamExplorer is another tool that enables creation and termination of software agents.Unlike Dam_UI it doesn't contain an agent server, and provides a possibility to connect to any Agent server in the network via comunicators, retrival data concerning the agents and Places from this server and display them on the monitor, as well as to create and terminate agents and Places.

4. Naming system

Agent servers, as well as software agents, Communicators and Places executed in the server are identified by their names that are globally unique in the system. These names can be used as addresses to determine their location in the network.
Each of these names presents an URL containing internet address. It is advantageous to use Intenet addresses for mobil agents.

The following syntax is used to create names:

```
url         := schema":"location
schema      := "rmi"
location    := "//"[hostport"/"]agentsystem["/"place]
hostport    := host":"port
host        := hostname | hostnumber
port        := digit+
agentsystem := uchar+
place       := uchar+
```

5. Application

An agent system can be very helpful as an environment for data mining algorithms.
In many cases, data subjected to analysis are structured, that is, fixed columns represent data, where data have to have the same type of value. Network activities, log files, database outputs, etc. present data of such type. There are two approaches to analysis of these data: the first is to pass data to the host computer (server) and process them there in a centralized manner. This is rather inconvenient, as usually these data are huge and there may arose difficulties when passing the data to the analyser. The second approach is to analyze data locally and sent already analysed data to the user.

By monitoring the operational data, as well as using output of their analysis one may detect intrusions in the system. An environment is designed, which supplies distributed-structured data to software agent framed facilities and through these data to mining algorithms thath may serve as a tool for finding solutions to network managing and security problems.

References

[1] FIPA (Federation of Intelligent Physical Agents - Home Page http://www.cselt.stet.it/fipa/fipa_rationale.htm.

[2] Michael W. and N. Jennings (1995), "Agent Theories, Architectures, and Languages: a Survey," in Wooldridge and Jennings Eds., Intelligent Agents, Berlin: Springer- Verlag, pp. 1-22.

[3] Hayes-Roth, B. (1995). "Architecture for Adaptive Intelligent Systems" Artificial Intelligence: Special Issue on Agents and Interactivity, 72, pp. 329-365.

[4] Kulin A., Salmonsen G. and Aslanyan L., Security policy adaptation reinforced through agents, international seminar "Conversion Potential of Armenia and ISTC Programs", Yerevan.

[5] Aslanyan L., Markaryan K., Pasic A., Sahakyan H., Intrusion detection intelligence, Proceeding of the conference Computer Science & Information Technologies, Yerevan, pp. 429-433.

Բանականային ագենտ սերվեր համակարգ

Դ. Կարապետյան

Ամփոփում

Աշխատանքը նկարագրում է Բանականային ագենտ սերվեր համակարգը, որը հիմնված է ծրագրային ագենտների վրա: Բերված են ծրագրային ագենտների հատկությունների նկարագրությունները և ստեղծված համակարգի կիրառության մի քանի օրինակներ, մասնավորապես ներխուժման հայտնաբերումը հաշվողական լոկալ ցանցում.