Mobile Agents Markup Language[1]

Roberto Carlos Cascos Fernández[1] and Jesús Arturo Pérez Díaz[2]

[1] University of Oviedo, Computer Science Department
C/ Calvo Sotelo S/N, 33007 Oviedo, Asturias, Spain
Tel. (34) 985-206118
rcascos@gmx.net
[2] ITESM Campus Cuernavaca, Electronics and Communications department
Av. Paseo de la Reforma# 128-A. Temixco, Morelos, 62589. México
Tel. (777) 329-7163 Fax. 329-7166
jesus.arturo.perez@itesm.mx

Abstract. The aim of this paper is the definition of a data and instructions interchange language for mobile agents which belong to different mobile agents systems with the goal of allowing the information interchange between these applications and improving students learning in distributed environments.

1 Mobile Agents

Mobile agents are autonomous and intelligent programs that roam throughout a net gathering information and interacting with different services desired by its programmer [1]. They are mainly used for: monitoring, data compilation (searching and filtering), massive distribution of information and supercomputer emulation [2].

A mobile agent system is a set of programs and tools that allow the programming, execution and monitoring of mobile agents.

1.1 Communication between Mobile Agents

There are many different mobile agent systems. All of them provide simple tools for communication between their own agents, but also most of them are incompatible with other systems, which generates a lack of interoperability when mobile agents of different platforms try to exchange information.

Since messages in plain text are required to allow communication among agents of different mobile agent systems, a common directory will be used where agents which want to establish communication can send and leave their messages and other who want to get information can retrieve and read their messages.

Although this method is limited, it is similar to the method in any other mobile agent system. Messages can be sent only by the agents programmed to do so, similarly only the agents programmed to listen messages can do it.

[1] This paper has been supported by the ITESM inside the research project of Distributed collaborative systems for teaching.

J.M. Cueva Lovelle et al. (Eds.): ICWE 2003, LNCS 2722, pp. 54–57, 2003.

2 MAML

The acronym MAML stands for "Mobile Agents Markup Language" and it will be the name that we give to the message interchange language.

The first thing to be done is to delimit the information that an agent owns and that could be useful to other agents. This information can be classified in several groups:

The description of this language will be based on a DTD, part of it is given in this chapter. The name of the different DTD elements tries to describe its content.

The XML root element simply includes a reference to each of the five internal elements, being the last four optional. It will be as follows:

```
<!ELEMENT MobileAgent (GeneralInformation, Security?,
Mobility?, Data?, Methods?)>
```

2.1 General Information

In this category, the following information could be included:

- Name of the agent: This name must be a global and unique identifier.
- Author: name of the author/authority and other important information about him.
- Creation date of the agent; hour of execution, lifetime, system which it belongs to.
- Creation date of the XML file.
- Agent description: a brief description of its goal.

```
<!ELEMENT GeneralInformation (AgentName, Author?,
Date?, AgentSystem?, Description?)>
```

2.2 Security

Usually, part of the information of the XML document must be hidden to some agents and this information can only be known by other agents that have been programmed by us. Besides, the authenticity of the information included in the file must be ensured. Finally, modification of these files by other agents or by the server cannot be allowed, in order to avoid that other agents change it and write false information.

To solve these problems, information about the security system used is included in the file. Also, private data will be encrypted. The agents that read these files must know the proper public keys to decrypt these data. These keys will be used according to the specified security policy.

In order to ensure that data has not been modified a digital signature containing the hash of the file can be included, in this way, if the public key of the agent's creator entity is known, the destination can ensure that the file has not been modified.

- Creator entity/authority: it will be necessary only if this field does not match up which the data given in the general information.
- Digital signature: with a hash of the message and information about the entity.
- Used method for the hash: by using this field, any method is allowed.

An attribute which indicates the encryption algorithm, will be used to mark an element as “encrypted”. Agents, which want to read that field, must recognize de encryption algorithm mark and decrypt the field content by using this algorithm.

```
<!ELEMENT Security (CreationEntity, DigitalSignature?,
HashMethod?)>
```

2.3 Mobility Information

The agent can use this element to write the list of the servers it has been through, as well as the list of the servers it plans to visit.

- Visited servers: for each server several data can be considered, for example IP address, URL, if the address has been useful, etc.
- Server to visit: in the same way that the visited servers list, the IP address, URL and additional information can be included.

```
<!ELEMENT Mobility (VisitedServers?, Path?)>
```

2.4 Data

The agent can cooperate with other agents and tell them the data it has found. To make them possible to understand each other, the other agents must know what the searching agent is looking for. Data is one of the elements that should be keep it encrypted so that not all the agents are able to know what has been found it by others.

```
<!ELEMENT Data (#PCDATA)>
<!ATTLIST Data encoded CDATA #IMPLIED >
```

2.5 Methods

Commands that an agent can give to others are included here. An agent should only perform the instructions given by a trusted agent (verifiable by the digital signature).

A programming language could be developed so that we would be able to create our own methods, but in order to simplify the process, it is going to be considered that all functions are already pre-programmed. In this way just a few of the more used functions in mobile agents are going to be described.

A method has three parts (if it is considered that it does not return a value):

- Name of the function.
- Name (identification) of the target of the function (one agent or many).
- List of parameters.

As an example of functions that can be necessary we can consider the following:

- Add the following URL to the list of server that will be visited.
- Indication that the searched information has been found.
- Show an element on the screen.
- Load a file. The parameters would be the name and location of the file to load.

The number of functions that could be specified could be very big and they should be programmed on the agent who receives the command. The four previous functions are an example of four very simple functions that an agent could require from another. Other functions could be of the type “look for this thing”, the destination agent must know how to perform that search. One search that can be executed by mobile agents is the search of other agents (location system). This system could be used to give instructions to other agents by finding them first.

```
<!ELEMENT Method (MethodName, TargetAgent,
ParametersList?)>
```

3 Conclusions

The MAML language is only an example of a possible utilization of XML in the construction of data interchange languages among mobile agents.

MAML represents an efficient solution to the interoperability problem among mobile agent systems since it allows the sharing of information, data and instructions in a secure way to create a secure interchange environment. This environment can be used to develop collaborative systems that allows students to get a higher learning in distributed environments.

In order to make the language completely efficient the agent system must know all methods (commands) that can be used. Agents could then consult to the agent system which it belongs to. A common directory must be included programmatically in all the agent systems. In this place messages are going to be left and read by agents.

Before the language can be used, it would be necessary an extended list of all the commands that agents could require, as well as encryption and hash methods that could be used by the agents, so that they could create the digital signature and the hash of the message.

References

1. Cetus Links, 18,199 Links on Objects and Components / Mobile Agents [http://www.cetus-links.org/oo_mobile_agents.html]
2. Venners, B.: Solve real problems with aglets, a type of mobile agent [http://www.javaworld.com/javaworld/jw-05-1997/jw-05-hood.html]
3. Finin, T., Fritzon R., Mackey, D. and MacEntire, R.: KQML as an agent communication language, [http://www.csee.umbc.edu/pub/ARPA/kqml/ papers/kqmlacl.pdf] (1995)
4. Cabri, G., Leonardi, L., Zamboneli. F.: XML Dataspaces for Mobile Agent Coordination. ACM Symposium on Applied Computing (2000)
5. Pérez Díaz, J. A.: Tesis Doctoral: SAHARA: Arquitectura de seguridad Integral para Sistemas de Agentes Móviles basados en Java. University of Oviedo, Spain (2000)
6. Petrie Charles J.,: Agent-Based Engineering, the Web, and Intelligence. IEEE Expert, (1996)