Agent Based Computing Environment for Accessing Previleged Servics

Agarwal N.

Department of Information Technology, National Institute of Technology, Durgapur, India
Email address: navin0706@gmail.com

Dutta A.

Department of Information Technology, National Institute of Technology, Durgapur, India
Email address: animeshnit@gmail.com

Abstract

In this paper we propose an application for accessing privileged services on the web, which is deployed on JADE (Java Agent Development Framework) platform. There are many Organizations/ Institutes which have subscribed to certain services inside their network, and these will not be accessible to people who are a part of the Organization/Institute when they are outside their network (for example in his residence). Therefore we have developed two software agents; the person will request the Client Agent (which will be residing outside the privileged network) for accessing the privileged services. The Client Agent will interact with the Server Agent (which will be residing inside the network which is subscribed to privileged services), which will process the request, and send the desired result back to the Client Agent.

Keywords*: JADE, Internet Surfing*

1 Introduction

Many Organizations/Institutes have subscription to certain services inside their network, for example here at NIT Durgapur there is subscription of IEEE and ACM. When outside the network, these services cannot be accessed. We plan to address this problem and also automate the whole process so that so that human effort is reduced.

To solve the problem we will build an agent based system, where multiple agents will interact with each other to solve the problem. When we talk about multiple agents interacting, the system becomes a Multi-Agent system, descriptions of which are given below.

A. Agent

An agent is a computer system or software that can act autonomously in any environment. Agent autonomy relates to an agent’s ability to make its own decisions about what activities to do, when to do, what type of information should be communicated and to whom, and how to assimilate the information received. An agent in the system is considered a locus of problem-solving activity; it operates asynchronously with respect to other agents. Thus, an intelligent agent inhabits an environment and is capable of conducting autonomous actions in order to satisfy its design objective [1-5]. Generally speaking, the environment is the aggregate of surrounding things, conditions, or influences with which the agent is interacting. Data/information is “sensed” by the agent. This data/information is typically called “percepts”. The agent operates on the percepts in some fashion and generates “actions” that could affect the environment. This general flow of activities, i.e., sensing the environment, processing the sensed data/ information and generating actions that can affect the environment, characterizes the general behavior of all agents.

B. MAS (Multi Agent System)

Multi-agent systems (MASs) [2, 5] are computational systems in which two or more agents interact or work together to perform a set of tasks or to achieve some common goals [5-8]. Agents of a multi-agent system (MAS) need to interact with others toward their common objective or individual benefits of themselves. A multi-agent system can be studied as a computer system that is concurrent, asynchronous, stochastic and distributed. A multi agent system permits to coordinate the behavior of agents, interacting and communicating in an environment, to perform some tasks or to solve some problems. It allows the decomposition of complex task in simple sub-tasks which facilitates its development, testing and updating.

The client agent outside the network which is subscribed to certain services need to interact with some agent residing inside the network, which will do the work on behalf of the user and send him back the result. In this paper we propose the whole architecture of the system, and how different agents will interact with each other.

To develop the MAS, we will use JADE which is a software framework fully implemented in Java language. It simplifies the implementation of multi-agent systems through a middle-ware that claims to comply with the FIPA specifications and through a set of tools that supports the debugging and deployment phase. The agent platform can be distributed across machines (which not even need to share the same OS) and the configuration can be controlled via a remote GUI. The configuration can even be changed at run-time by creating new agents and moving agents from one machine to another one, as and when required. The only system requirement is the Java Run Time version 5 or later.

The communication architecture offers flexible and efficient messaging, where JADE creates and manages a queue of incoming ACL messages, private to each agent. Agents can access their queue via a combination of several modes: blocking, polling, timeout and pattern matching based. The full FIPA communication model has been implemented and its components have been clearly distinct and fully integrated: interaction protocols, envelope, ACL, content languages, encoding schemes, ontologies and, finally, transport protocols. The transport mechanism, in particular, is like a chameleon because it adapts to each situation, by transparently choosing the best available protocol. Most of the interaction protocols defined by FIPA are already available and can be instantiated after defining the application-dependent behavior of each state of the protocol. SL and agent management ontology have been implemented already, as well as the support for user-defined content languages and ontologies that can be implemented, registered with agents, and automatically used by the framework.

2 RELATED WORK

Agent-based models have been used since the mid-1990s to solve a variety of business and technology problems. Examples of applications include supply chain optimization [9] and logistics [10], distributed computing [11], workforce management [12], and portfolio management [13]. They have also been used to analyze traffic congestion [14]. In these and other applications, the system of interest is simulated by capturing the behavior of individual agents and their interconnections. In this paper [15] a framework for constructing application in mobile computing environment has been proposed. In this framework an application is partitioned into two pieces, one runs on a mobile computer and another runs on a stationary computer. They are constructed by composing small objects, in which the stationary computer does the task for the mobile computer. This system is based on the service proxy, and is not autonomous. In our work we are building our system based on agents which adds a lot of flexibility and is autonomous. A Multi-Agent system [16] for accessing remote energy meters from electricity board is related to this work. In this the server said to be the host is located in the electricity board, and all the customers are the clients connected with the server. This MAS system helps in automating the task and thus replacing the human agents. It is similar to our scenario where we are automating the task of downloading papers from IEEE/ACM sites, and replacing human agents which can do the task being inside the privileged network. In this paper [17] architecture has been proposed for secure and simplified access to home appliances using Iris recognition, adding an additional layer of security and preventing unauthorized access to the home appliances. This model is also based on server and client approach, where the server will reside inside the home, and client will reside outside the home and send request to the server for performing task on behalf of the user. Advanced method for downloading web-pages from the internet has been proposed here [18], we will be using many concepts from these to improve the working of the server agent and more utilization of bandwidth.

3 PROBLEM OVERVIEW

There are many networks which have privilege accessibility to many sites and servers. For example being inside NIT Durgapur, there is no authentication required for downloading papers and other documents from IEEE and ACM sites.

A user who has the right to access that privileged network, but if he is outside that network he will not be able to. There can be scenarios in which an Institute or Organization, can pay for some services to be accessed inside their network. In such situations the user has to be inside the network to enjoy those services or, they can access the network from outside by means of a Proxy Server (There are some more possibilities).

4 SCOPE OF WORK

The aim of this work is to automate this whole process. Make the work of the user easy and take advantage of the services or privilege that he is entitled to access being a part of the Institute or Organization. Developing the agent in JADE allows us to implement it for Mobile devices also. The only requirement for running any JADE agent is, Java Run Time Environment, which most of the System and Mobile Devices have.

In this project, the user will just need to send the keyword, and all the related documents matching that keyword will be downloaded and sent to the user. The user need not wait for the whole process to finish. He just needs to send the request, and the Multi-Agent System will perform the task for the User.

The main purpose of technology is to ease human work, so that the effort can be put to do more useful work. This project targets that specific purpose, with some added benefits to the user.

5 MULTI-AGENT BASED ARCHITECTURE

The agent system is divided in two parts:

- There will be one single agent called Server Agent which serves the requests of multiple users.
- Multiple Client Agents which will send a request to the Server Agent in form of a Keyword.

A. Server Agent

This Agent will run autonomously inside the network, which has privileged access, or has been authorized to a service. It will always be ready to accept request from client. Then that keyword will be searched in a Search Engine, and the source code of that web page will be downloaded using Java. That webpage will contain many links, and also some documents. If a link is found while searching the source code of that webpage, then its source code will also be downloaded. This can be visualized in the form of a graph as shown in figure 1. Building this graph will help us not to search for same links again. While parsing the source code of the webpage whenever a link is found, the java code for downloading the source will be called again and executed in a different thread. Whenever any document is found then the java code for downloading files from web will be called and executed in a different thread. This process may continue forever, so we will restrict the depth of the graph from the starting page. Whenever we are parsing the source code which is at a maximum depth from the starting page then only documents in that web page will be downloaded.

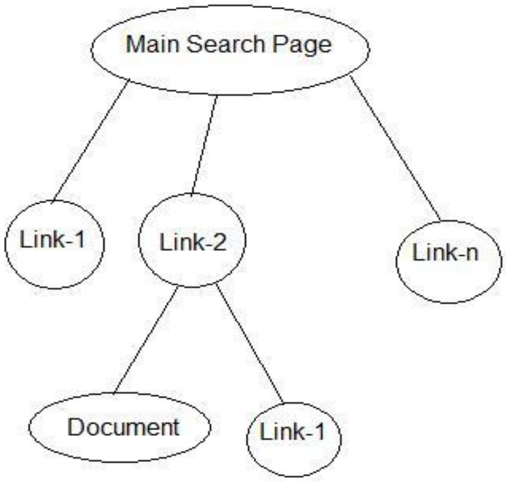

Figure 1: Diagram showing the graph, of the links and documents.

B. Client Agent

This will be a very simple agent, it will perform two tasks.

- Authentication of the user, it will send a request to the server with the credentials. If the user is authenticated then, the user will be able to perform his task.
- Provide a simple GUI to the user for sending his keyword, relevant to which the user requires documents (or papers). The user can also directly send the link of the document, or IEEE page in this situation the keyword will not be searched in a search engine, but the server agent will perform the next step directly.

In figure 2 interactions between Client Agent, Server Agent and Java Codes in Server has been shown. First step is the authentication process, in which client sends the credential to the server agent for verification. If the credentials are verified then the client will be granted access. The client then sends the search keyword to the server agent, which then verifies if the keyword is valid. If it is valid then, the server agent calls the Java code for downloading source code, which will search all the links starting from the mail search page, and process as shown in figure 3. The source code downloader will send the list of all the documents found back to the server agent. Then the agent will send this list to the Document downloader code, which will download all the documents, and save it in a zipped folder ready to be sent to the client. Then it will notify the server agent that the downloading has been done, and finally the server agent will send the zipped folder to the client.

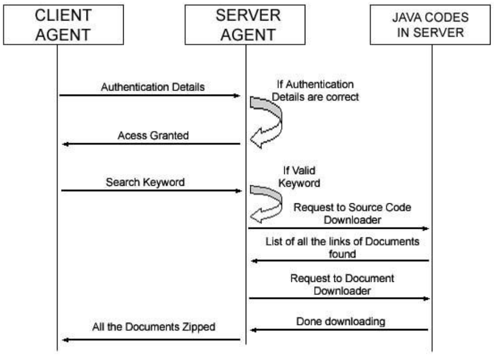

Figure 2: Diagram showing the interaction between Server Agent, Client Agent and Java Codes in Server.

6 PROTOTYPE DESIGN

Figure 3 represents the main JADE prototype elements. An application based on JADE is made of a set of components called Agents each one having a unique name. Agents execute tasks and interact by exchanging messages. Agents live on top of a Platform that provides them with basic services such as message delivery. A platform is composed of one or more Containers. Containers can be executed on different hosts thus achieving a distributed platform. Each container can contain zero or more agents.

A special container called Main Container exists in the platform. The main container is itself a container and can therefore contain agents, but differs from other containers as

- It must be the first container to start in the platform and all other containers register to it at bootstrap time.
- It includes two special agents: the AMS that represents the authority in the platform and is the only agent able to perform platform management actions such as starting and killing agents or shutting down the whole platform (normal agents can request such actions to the AMS). The DF that provides the Yellow Pages service where agents can publish the services they provide and find other agents providing the services they need.

Agents can communicate transparently regardless of whether they live in the same container, in different containers (in the same or in different hosts) belonging to the same platform or in different platforms (e.g. A and B). Communication is based on an asynchronous message passing paradigm. Message format is defined by the ACL language defined by FIPA [19], an international organization that issued a set of specifications for agent interoperability. An ACL Message contains a number of fields including

- The sender
- The receiver(s).
- The communicative act (also called performative) that represents the intention of the sender of the message. For instance when an agent sends an INFORM message it wishes the receiver(s) to become aware about a fact (e.g. (INFORM "today it's raining")). When an agent sends a REQUEST message it wishes the receiver(s) to perform an action. FIPA defined 22 communicative acts, each one with a well defined semantics, that ACL gurus assert can cover more than 95% of all possible situations.

Fortunately in 99% of the cases we don't need to care about the formal semantics behind Communicative acts and we just use them for their intuitive meaning.

- The content i.e. the actual information conveyed by the message (the fact the receiver should become aware of in case of an INFORM message, the action that the receiver is expected to perform in case of a REQUEST message).

In Figure 3, three agents have been shown two clients and one server. Every system that runs a JADE platform will have a main container where all the agents run. The two clients send a request to the server which contains the search query or the link for the paper/document to be downloaded.

Host 3: Server

Host 1: Client

Host 2: Client

Host 3 is the server, where the JADE platform will run, there is only one container called the Main Container where along with the server agent two more agents called AMS and DF will run. Name of the server agent is B@Platform2, and it address is http://host3:7778/acc. When the client agents will communicate with the server agent remotely then host3 must be fully qualified domain name.

There are two clients Host 1 and Host 2, both this will have a JADE platform with one container called the Main Container where along with client agent there will be two more agents called AMS and DF running. When a user wants to send a request to the server agent, the client agent will send a message to the server agent, where the receiver address in this case will be http://host3:7778/acc and the name of the agent will be B@Platform2, along with other necessary details.

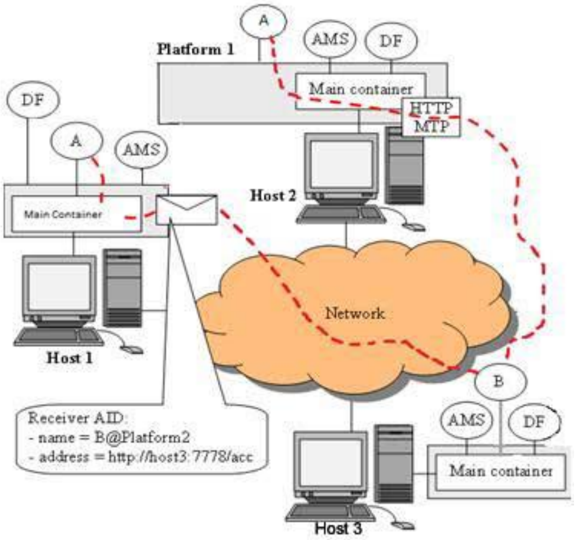

Figure 3: Diagram showing two Client Agents, sending message to the server agent.

7 Conclusions

In this work, we have developed an Agent based system for accessing privileged services in a network remotely. The service in this scenario is subscription to IEEE and ACM sites that do not require authentication being inside the network. We have also automated the process of downloading papers/documents from the web that match the search keyword. This application is the first implementation of this type, so there is a lot of scope of improvement in it. We plan to improve the search and give better results, by considering the semantics of the search keyword. This work addresses one such privileged service, this model can be used as a base and expanded to include a lot more of such services and even provide automation wherever possible.

References

[1] Christopher A. Rouff, Michael Hinchey, James Rash, Walter Truszkowski, and Diana Gordon-Spears (Eds), "Agent Technology from a formal perspective" (Springer-Verlag London Limited 2006).

[2] G. Weiss, Ed., "Multiagent systems: a modern approach to distributed artificial intelligence", (MIT Press, 1999).

[3] N. J. Nilsson, "Artificial intelligence: a new synthesis", (Morgan Kaufmann Publishers Inc., 1998).

[4] S. J. Russell and P. Norvig," Artificial Intelligence: A Modern Approach", (Pearson Education, 2003).

[5] M. J. Wooldridge, "Introduction to Multiagent Systems", (John Wiley & Sons, Inc., 2001).

[6] A. Idani, "B/UML: Setting in Relation of B Specification and UML Description for Help of External Validation of Formal Development in B", Thesis of Doctorat, The Grenoble University, November 2005.

[7] G. W. Brams, "Petri Nets: Theory and Practica"l, Vol. 1-2, (MASSON, Paris, 1982).

[8] M-J. Yoo, "A Componential For Modeling of Cooperative Agents and Its Validation", Thesis of Doctorat, The Paris 6 University, 1999.

[9] Jiun-Yan Shiau, Xiangyang Li, "Modeling the supply chain based on multi-agent conflicts", Service Operations, Logistics and Informatics, 2009. SOLI '09. IEEE/INFORMS International Conference on, Publication Year: 2009, Page(s): 394 – 399

[10] Yan Wang, YinZhang Guo, JianChao Zeng, "A Study of Logistics System Model Based on Multi-Agent", Service Operations and Logistics, and Informatics, 2006. SOLI '06. IEEE International Conference on, Publication Year: 2006, Page(s): 829 – 832

[11] R. Al-Khannak, B. Bifzer, Hezron, "Grid computing by using multi agent system technology in distributed power generator", Universities Power Engineering Conference, 2007. UPEC 2007. 42nd International, Publication Year: 2007 , Page(s): 62 – 67

[12] [Online]. Available:http://en.wikipedia.org/wiki/ Workforce_management

[13] V. Krishna, V. Ramesh, "Portfolio management using cyberagents", Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on, Issue Date : 11-14 Oct 1998 Volume : 5 , On page(s): 4860 - 4865 vol.5

[14] Application of Agent Technology to Traffic Simulation. United States Department of Transportation, May 15, 2007.

[15] A. Hokimoto, K. Kurihara, T. Nakajima, "An approach for constructing mobile applications using service proxies" , Distributed Computing Systems, 1996., Proceedings of the 16th International Conference on , Issue Date : 27-30 May 1996 , On page(s): 726 - 733

[16] C. Suriyakala, P.E. Sankaranarayanan, "Smart Multiagent Architecture for Congestion Control to Access Remote Energy Meters" , Issue Date : 13-15 Dec. 2007 , Volume : 4 , On page(s): 24 - 28

[17] A. Mondal, K. Roy, P. Bhattacharya, "Secure and Simplified Access to Home Appliances using Iris Recognition" , Computational Intelligence in Biometrics: Theory, Algorithms, and Applications, 2009. CIB 2009. IEEE Workshop on, Issue Date : March 30 2009-April 2 2009 , On page(s): 22 – 29

[18] A. Kundu, A.R. Pal, Tanay Sarkar, M. Banerjee, S. Mandal, R. Dattagupta, D. Mukhopadhyay, "An Alternate Downloading Methodology of Webpages" , Artificial Intelligence, 2008. MICAI '08. Seventh Mexican International Conference on , Issue Date : 27-31 Oct. 2008 , On page(s): 393 – 398

[19] [Online]. Available : http://www.fipa.org