Journal of the Nigerian Association of Mathematical Physics
***Volume27*, (*July, 2014*), pp 521 – 526**
© J. of NAMP

Development of a Mobile Agent Operational Algorithm

[1]Imianvan Anthony Agboizebeta and [2]Akinyokun Oluwole Charles

[1]Department of Computer Science, University of Benin, Benin City. Nigeria.
[2]Department of Computer Science, Federal University of Technology, Akure. Nigeria.

Abstract

Mobile agent research is fast evolving. However, most mobile agent researchers often ignore presenting the underlying algorithm in their report. This creates great problems for fresh developers. Besides, mobile agent developers are often over bored by the problem of mobility and autonomy. In this paper, an operational technique of a typical mobile agent for the assessment of the resources of computer network is unveiled. The algorithm provided will ease the nerves of future mobile agent developer.

Keywords: Mobile Agent, Mobility, Autonomy, Schema, Formal Methods, Zed notations.

1.0 Introduction

Mobile agents are autonomous and intelligent programs that are capable of moving through a network, searching for and interacting with the resources on behalf of the network administrator. Mobile agent is an executive program that can migrate at times of its own choosing from one machine to another in a network**[1 - 6].**

The technology of computer memory is very high. With the high cost of computer memory in the 90's, there was a conscious effort made in **[7]** to develop a utility program to police the economic utilization of the memory of Microcomputers. A program was written in PASCAL to evaluate the extent of utilization of Random Access Memory and hard disks of computers. The program was implemented and tested at the computer center of the Federal University of Technology, Akure, Nigeria. Computers used for the experiment were stand-alone (that is, not connected to form a network). The utility program is capable of been activated and run on a stand-alone computer to:

a. Keep track of the users' files in a computer hard disk.
b. Give notice of each of the users file that:
    i. Is more than one month old.
    ii. Has not been accessed and updated within a month.
    iii. Occupies more than a specified memory space.
c. Raises an alarm and recommends backup of the offending files by the operations manager.
d. Automatically delete the offending user files at the end of the third alarm.

That work can be described as a mobile agent within a computer system. Although the author foresees the usefulness of the developed package in a network environment, yet, it was not adopted and adapted to computer network environment.

Mobile agent could be activated and launched from one computer to another for the purpose of autonomously searching for and interacting with network resources on the network administrator's behalf. A conscious effort at developing a mobile agent for the assessment and evaluation of computer networks with emphasis on throughput, utilization and availability have been proposed**[3]**. The proposed algorithm, which serves as the driving force for this mobile agent research was simulated analytically. Analytical simulation is mathematical and algorithmic in nature. The analytical simulation is not an adequate tool for implementing a real-life system typical of mobile agent. This research is aimed at providing an algorithm for mobility of agent that can serve as a takeoff point for mobile agent developers.

2.0 Mobility Software

In theory any language can be used to implement mobile agents. The only necessary requirement is that an execution environment on the host supports the language. A wide variety of languages have been used to write mobile agents which include Telescript, Java, Obliq, Agent Tcl **[8]**.

Corresponding author: Imianvan A.A. E-mail: tonyvanni@yahoo.com Tel.: +2347069742552.

Telescript is a proprietary system developed by General Magic **[8]**. The Telescript language is specifically designed for implementing mobile agent systems. Telescript was designed with the vision for the computer networks to become a programmable platform. Basically, Telescript in communications is synonymous to Postscript in printing. Telescript is an Object Oriented Language which supports objects, classes and inheritance.

The object oriented model and the syntax of Telescript is in many way similar to that of C++. Telescript has a library of built-in classes for writing mobile agents. The Telescript language has a set of built-in commands for agent migration and inter-agent communication. The Telescript system includes notions of which authority the agent is representing. Telescript programs are compiled into a portable intermediate representation, called *low Telescript*, analogous to Java byte code. Telescript programs can run on any computer with a Telescript execution engine. The Telescript execution engine was designed to be able to run on even small communication devices. The Telescript language has had a great influence on the development of mobile agents, and mobile agent languages. Infact, the features of Telecript made General Magic to first coined the term *mobile agent*.

3.0 Mobility Algorithm of the Agent

Mobility is concerned with the movement of agent system from one workstation to another in a network environment. Telescript is a language presented in **[9,10]** for mobile agent development and implementation. Telescript supports some base commands each of which calls some procedures or utility programs while, in turn, each procedure or program performs a specific task. The following commands amongst many other commands of Telescript are adopted for the purpose of developing and implementing the mobility aspect of a typical mobile agent for bandwidth assessment.

a. TicketAgent.
b. SelfSendAgent and SelfHomeAgent.
c. LiveAgent.
d. MeetAgent.
e. DumpAgent.

3.1. Ticket Agent

TicketAgent is made up of the following subcommands:

a. TeleAddress.
b. LookUpPlaces.
c. Copy.
d. Here.

TicketAgent operates as follows:

a. Tags the targets using TELE command. For example, use: TELEADDRESS to tag target address.
b. Uses LookUpPlaces command to autonomously view targets workstations.
c. Build a graph of the target workstations.
d. Uses COPY command to collect target address.
e. Uses HERE command to instruct the agent to carry out the instruction from present location.
f. Uses TICKET command to notify the agent of the target workstations.

TicketAgent algorithm is presented in Figure 1.

```
1.  TicketAgent(agent, v, e)
2.      // agent is to be deployed to workstations.
3.      // v is set of vertices of the graph representing workstations.
4.      // e is set of edges of the graph representing distance between workstations.
5.      TargetFound ← false
6.      j ← 1
7.      loop i from 1 to number of workstations
8.          TeleAddress $w_i$       // tag target addresses.
9.          LookUpPlaces $w_i$      // used to autonomously view targets workstations.
10.         // $w_i$ is target workstations
11.         if $w_i$ not in v
12.             v ← v U {$w_i$}
13.         end if
14.         copy(v)     // collect or copy target address
15.         j ← j + 1
16.         if $w_i \neq w_j$ $edge_{i,j}$ ← distance $w_j$ – distance $w_i$
17.             e ← e U $edge_{i,j}$
18.         end if
19.         copy(e)    // collect or copy distance between targets
20.         TargetFound ← true
21.         Ticket (agent, $w_i$)
22.     end loop
23. end TicketAgent
```

Figure 1: TicketAgent Algorithm.

The following are sequence of Telescript commands specification tonotify an agent (for example, Bandwidth Agent) of target workstations:
Teleaddress addr := here@LookUpPlaces.Address.Copy();
Ticket(BandwidthAgent, addr):

3.2 Self Send Agent and Self Home Agent

The movement of the agent is self-triggered and involves the agent sending copies of itself across to target machines along appropriate directions of the target workstations. Sending copy of the agent employs the following procedure:

a. Evaluate and select agent routes to target workstations.
b. Use HERE command to lay hold on the agent at current node.
c. Use SELF-command to activate autonomy.
d. Use SEND command as object of SELF-command to cause the movement of the copy of the agent to the target.

The evaluation of the agent route involves the assessment of the graphical network produced by the target workstations. An edge of the graph of the agent network represents the distance between workstations whereas the vertices represent the workstations (targets). The edge is weighted by the cost of sending the agent from source to target workstations.
After evaluating and selecting the route to take through the network, the mobile agent stores the list of routes in an enumerated variable called Sendlist. The agent then activates the here, self and send commands. For example, to lay hold on bandwidth agent and autonomously send it across to list of targets we issue the commands:
here@ LookUpPlace.BandwidthAgent;
SELF.SEND (SendLIST);
When the agent completes its task, the agent has to go back home. The procedure to return the agent home is activated by SelfHomeAgent. Self HomeAgent is made up of the following subcommands:

a. SELF
b. GO
c. HOME

SelfHomeAgent operates as follows:

a. Use SELF-command to activate autonomy.
b. Use GO command to activate the agent movement.
c. Use HOME command to activate return home.

For example, the following command transports an agent to its home computer.
Self.Go (Home)

3.3 Live Agent

LiveAgent causes the execution of the agent. The LiveAgent calls and executes the procedures that are concerned with the assessment and evaluation of the bandwidth of the network environment. LiveAgent is made up of the following subcommands:

a. Sponsored.
b. Try.
c. Live

LiveAgent operates as follows:

a. Use SPONSORED command to activate start of execution and define operations to be exempted and declare variables.
b. Use TRY command to activate the execution of the procedure body of the agent.
c. Use LIVE command to merge the variables and procedure body.

LiveAgent uses the game of life algorithm presented in [Akinyokun, 1997] to account for the utilization of the bandwidth subscribed by computer network environment.
Figure 2 is an example of sequence of commands pattern for LifeAgent.

```
        Live:   sponsored op (Exception     Nil) =   {
                aPlace: Teleaddress;
    try  {
       self.send (Ticket);
         :
        //   for each agent execute game of life algorithm here.
        :
            self.go(home);
      }
   }
```

Figure 2: LiveAgent Algorithm

3.4 Meet Agent

The MeetAgent command allows the agent to interact with other agents at a meeting place. Meet Agent is made up of the following subcommands:

b. A Petition.
c. Here.
d. Meeting Place.
e. Meet.

Meet Agent operates as follows:

a. List agents to meet using aPetition command.
b. Use HERE command to lay hold on the agent at current location.
c. Use MeetingPlace for agent meeting places
d. MEET with the agent using the command.

```
        agentToMeet := here@MeetingPlace.Meet (aPetition, nil);
```

3.5 Dump Agent

Dump Agent facilitates the disposal of the mobile agent after completing its assignment. The disposal of the mobile agent simply means removing the mobile agent from the universal set of agents. The process of the mobile agent disposal is presented in Figure 3 using Formal methods (Zed notations) **[11,12]**. Telescript command used to dispose agent is DUMP.

Agent

AgentName : seq CHAR

$\forall a, b : Agent \bullet a \neq b \Rightarrow a.AgentName \neq b.AgentName$

DisposeAgent

AgentSet, AgentSet′ : $\mathbb{P}$ Agent

Remove? : Agent

Remove? $\in$ Agentset

$\exists_1$ k : Agentset $\bullet$ k.AgentName = Remove?.AgentName

AgentSet$'$ = AgentSet \ {Remove?}

Figure 3: Specification of the Mobile Agent Disposal Process.

4.0 Conclusion

Operational procedure of a typical mobile agent for network resource assessment has been presented. The proposed system is not only valid for bandwidth assessment alone but other agent developers could find the technique for mobility and autonomy using Telescript infrastructure useful. The system will facilitate the development of future mobile agents.

5.0 References

[1] Imianvan Anthony Agboizebeta and Akinyokun Oluwole Charles(2014), “Formal Characterization of a Mobile Agent Operational Environment”, Accepted for Publication in Journal of NAMP, March 2014.

[2]. Imianvan Anthony Agboizebeta (2009), “Development of a Mobile Agent System for Evaluating the Use of Bandwidth in a Computer Network”, PhD Thesis, Federal University of Technology, Akure, Ondo State. Nigeria.

[3] Aderounmu G. A. (2001), “Development of an intelligent Mobile Agent for Computer Network Performance Management”, *PhD Thesis, Obafemi awolowo University, Ile-Ife, Nigeria.*

[4] Weina He and Gaoyuan Liu (2011), The application of mobile agent in e-commerce, 3rd International Conference on Advanced Computer Control (ICACC), 2011, HARBIN.

[5] Djamel Eddine Menacer, Habiba Drias, Christophe Sibertin-Blanc (2012), MP-IR: Market-Oriented Mobile Agents System for Distributed Information Retrieval, Advances in Intelligent and Soft Computing, Volume 122, pages 379-390, 2012.

[6] Huy Hoang To, Shonali Krishnaswamy, and Bala Srinivasan (2005), Mobile Agents for Network Management: When and When Not! , ACM Syposium on Applied Computing.

[7] Akinyokun O. C. (1997), “Catching and Using the Virus”, The Journal of the Institute of the Management of Information Systems (IMIS), London, United Kingdom, Vol. 7, No. 6, Pages 12-17.

[8] White J. E. (1996), Mobile Agents White Paper, General Magic, Available at: http://www.genmagic.com/agents/Whitepaper/whitepaper.html

[9] General Magic (1995a), Telescript Programming Guide, Version 0.5 (ALPHA), October, 1995a, General Magic Incorporated, Canada.

[10] General Magic (1995b), Telescript Language Reference, October, 1995b, General Magic Incorporated, Canada.

[11]. Diller A., (1994), Z : An Introduction to Formal Methods, (2nd edition), John Wiley and Sons

[12]. Spivey J. M. (1998), “The Z notation: A Reference Manual”, Prentice Hall International, United Kingdom