RESOURCE MANAGEMENT IN HYBRID WIRELESS NETWORKS USING INTELLIGENT MOBILE AGENTS

Alexandru L. RICOBON[1] Ramona M. TRESTIAN[2]
([1]) *Technical University of Cluj-Napoca, Romania;* ([2]) *Dublin City University, Ireland*
([1]) alex.ricobon@com.utcluj.ro ([2]) ramona@eeng.dcu.ie

Abstract: Wireless networks are no more used only for data transfer and Internet access, but now also support voice and multimedia services. The tremendous demands from the users are pushing the development of mobile communications faster than ever before, leading to plenty of new advanced techniques emerging. Future wireless Internet will consist of different wireless technologies operating together. The key issue is to provide new services and maintain quality of service (QoS) in hybrid wireless networks. This can be done in many ways, one of those using mobile agents to travel across the network and configure the route or the application demands.

Key words: *resource management, hybrid wireless networks, intelligent mobile agents.*

I. INTRODICTION

There have been extensive researches going on in the development of the mobile agent systems. But there are not many efforts in the study of their performance in real world applications and very few in wireless networks. As a result, the spread of mobile agent technology in the real world applications cannot be seen yet. The main obstacles faced by the researchers are the complexity of evaluating distributed applications in heterogeneous networks and expenses of building test beds for their experiments. A computer simulator can overcome these obstacles. Such a simulator is able to provide users with practical feedback when designing real world applications.

II. NETWORK SIMULATOR (NS-2)

Ns-2 is a network simulator that is originally based on the REAL network simulator. It is an open source software and commonly used by network researchers for evaluating and developing network related research. Since it is an open source software and additions are encouraged, many other network researchers have contributed to the development of the simulator. [1].

The simulator is developed and runs in two programming languages, OTcl and C++. OTcl is an object oriented version of the Tcl language, which is an interpreted scripting language. The OTcl part of the simulator is mainly concerned with configuring the network prior to simulation start and the C++ part mainly handles the packet processing within the network when running a simulation. The scripting properties of OTcl lets developers quickly explore many network layouts, while the compiled C++ code effectively executes the large amounts of data processing required by packet handling during simulations.

A simulating environment in ns-2 consists of network elements that simulate the behavior of a network and network connections that generate the data traffic used in the simulation. The network elements in ns-2 mainly consist of nodes and links between the nodes. At a higher detail level, each link and node consists of several internal elements to implement the behavior. Nodes acts as routers and traffic generation points, and links acts as transfer element between the nodes.

III. MOBILE AGENTS

A Mobile Agent is a type of software agent, with the feature of autonomy, social ability, learning, and most important, mobility. When the term 'mobile agent' is used, it refers to a process that can transport its state from one host to another. Mobile agents decide when and where to move next, saving their own state and resuming the execution on the new host from the saved state.

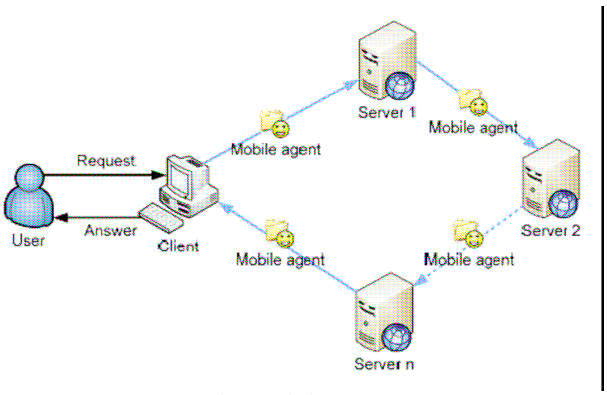

Figure 1. The Mobile Agent concept.

The traditional use for mobile agents is to retrieve information for the user. For this purpose we can use two approaches: the client-server model and the mobile agent model. In the first case, a client needing information from several servers, connects to each of them, makes a request

Manuscript received April 24, 2009, reviewed June 11, 2009

and expects an answer. This approach works just fine, but generates a lot of traffic.

In the mobile agent model, the client sends a self-contained piece of software (the mobile agent) to do the work for him. This concept is illustrated simplified in figure 1. The agent travels in the network from a server to another, collects data and returns to the client. If in the first case the traffic generated from the client was n requests (where n is the number of servers), in this case the client generates only one request.

Another advantage of the second model is that the mobile agent has enough intelligence to return only the relevant information to the user, sparing him the time needed to search thru the results.

The following applications can be potentially deployed more efficiently using mobile agents:

- **Information Retrieval:**

The most prominent application of the mobile agents is in distributed information retrieval. The information available in Internet is growing exponentially at every moment. Also the information is widely distributed. The information that can be retrieved using a search engine has its own limitations. Hence, for getting a certain set of instructions it is necessary to search different network sites. Sometimes this needs querying a series of servers one by one to get a desired result. It may also the case that the query to the next server needs the result from the previous server. Using traditional methods of communication like RPC can result in significant overhead both in terms of wireless bandwidth consumption and latency or total query time. The deployment of mobile agent technology can significantly improve the application's performance. This is because of the ability of mobile agent to roam autonomously in the wired network till the information is gathered and hence no need for intermittent communication through the wireless channel.

- **Filtering Data Streams:**

The second prominent application of mobile agents is in filtering the bulk amount of result data to return only what is relevant to the mobile user. Clearly, the agent avoids the transmission of unnecessary data, but does require the transmission of agent code from client to server. If the agent code is reasonable enough, a great saving in bandwidth and time can be attained.

- **QoS Provision in Wireless Multimedia Applications:**

The development and the deployment of multimedia services should meet the increasing user expectations and the growing requirements for QoS and should face the increasing heterogeneity in access devices. In this context, the traditional end-to-end model is showing its limitations. The network infrastructure should play an active execution role. Mobile agents are highly suitable for the implementation of active services (Baschieri, Bellavista and Corradi 2002), since they provide many active service properties like control decentralization, service tailoring to user requirements and resource availability, and adaptation of services in response to modifications in network resources.

- **Proxy based Personalized Services to Portable Devices:**

A lot of architectures have been proposed for supporting the portable devices for wireless Internet services. Most of them are based on a proxy-based middleware using mobile agents. It is not necessary to run an agent server on the user's device, rather user needs to provide a profile based on his requirements to a gateway server on the wired network acting as a proxy for the mobile device. This proxy then creates and launches agents for the user. In some architectures proxies are themselves deployed as a mobile agent and thus enabling their dynamical installation when and where necessary. Such proxy can follow the mobile device to continue serving it personalized services.

- **Commercial Wireless Services:**

Some of the architectures using mobile agents have also proposed wide range distributed commercial wireless services. Mobile agents can be optimized for product searching and collaborative filtering (recommendation of similar products). The authentication process is more secure if the user data is not exchanged on the wireless link.

IV. TEST SCENARIO

Resource management can be implemented in two ways: from the network point of view or from the application's point of view.

The network uses mobile agents to permanently test available links and store this information in the hops. This way the network is aware of its QoS capabilities (delay, packet loss, jitter, RTT, etc.), being able to accept or reserve resources asked by different applications.

In the other approach the application, before starting, sends a mobile agent to test the network. The agent travels across the network from source to destination and back, configuring the route that packets must follow. If none of the possible paths meet the requirements of the application, the agent can configure it to run with lower resources. The agent can also inform the nodes in his path that the application is about to start, and reserve resources.

Imagine person A wants to start a videoconference with person B at high video resolution and the link bandwidth is not enough. If the streaming application does not use mobile agents, the videoconference will start, but it will need a lot of buffering breaks, making it inacceptable from the point of view of the users. If before starting the application sends a mobile agent to test the link, the agent can change the resolution of the image and run smoothly. The connecting time is longer and the video quality is lower, but A can talk with B without interruptions.

Our test scenario combines both approaches, meaning that the network is self aware of its parameters (stored in the nodes) and the application sends a mobile agent before starting. Figure 2 shows the network topology used for our simulation. The mobile node (MN) wants to start a communication link (VoIP, FTP, etc.) with a host thru the wired network.

On each server is stored information about the service availability, medium number of dropped packets, jitter and error ratio. The application sends a mobile agent that travels from MN to BS on the radio interface, and from there, tests all four possible links between GW1 and GW2, chooses the best path (based on the information stored on the servers and its own measurements and estimation of RTT), tests the link

between GW2 and the host, and returns back to MN. Here, having all the information (packet loss, global RTT, jitter, average delay) and depending on the type of traffic generated by the application (real time, error free etc.) can configure its parameters.

In order to test our scenario, we used an Intel Celeron M 1.5 GHz computer with 1 GB of RAM running Fedora Core 4 operating system. An extension is necessary to include these features. We chose version 2.28 of Network Simulator, and successfully installed the mobile agents patch developed by Kunal Shah [2] for the version 2.1b9a of NS-2. In the simulation script we created a wireless node, a base station, two gateways, eight wired nodes and the correspondent host. These nodes are connected as shown in figure 2 by links with different bandwidth and delay.

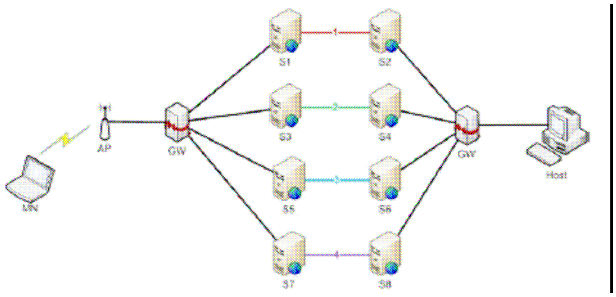

Figure 2. Network topology for the tested scenario.

The scenario simulates a voice and a video stream. The first part of the simulation consists in the network testing mechanism by the agents. After the agent chooses the best link, we start a 25 kbps CBR source, simulating a voice transfer. After 5 seconds, another CBR flow is activated. The second CBR flow has a 1 Mbps throughput, simulating a video stream.

V. RESULTS

Based on the network configuration presented in Table 1, the results given by the script are shown in figure 3. The script displays the minimum RTT (calculated).

The simulation produces two output files: agent.nam and agent.tr. The first one is used to view the network topology and packet flow in the visual network analyzer Nam (Network Animator). Nam is a tool included in the NS-2 distribution.

The second file is a trace file, storing all the events of the simulation. It contains the timestamp, sending node, receiving node, packet type, size, etc. This file can be analyzed with many tools, our choice being Tracegraph. This tool is developed in Matlab and is able to extract any type of information from the file, showing the results as 2D or 3D graphs.

The agent code size overhead is set to 0, representing ideal conditions; its data size is 5kB.

Table 1. Link Configurations

Link	Bandwidth [Mbps]	Delay [ms]
1	2	**3**
2	5	**10**
3	3	**2**
4	**20**	**15**

The time it takes the agent to travel from the mobile node to the host and back on the third link is 1.1 seconds. As one would have expected, the third link has the smallest travelling time.

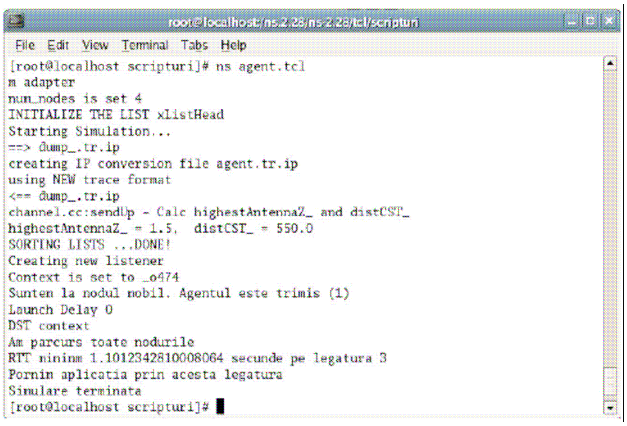

Figure 3. Simulation results.

Figure 4 shows the main window of Nam. It displays the network topology and packet flow for the current scenario. This screen print shows the agent travelling between the servers S5 to S6.

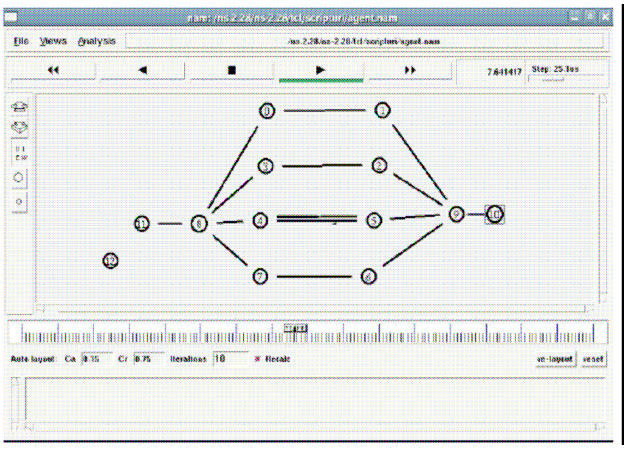

Figure 4. Scenario topology and packet flow in Nam.

The next figure shows the cumulative sum of the number of received packets. As one can see, at time 2, the first CBR source starts to send packets. At time 5, the second CBR source begins to send packets, without interfering with the first one. The simulation lasts 25 seconds, generating a total of 2000 CBR packets of 1kb each.

Figure 6 shows the same information, but as a 3d graph.

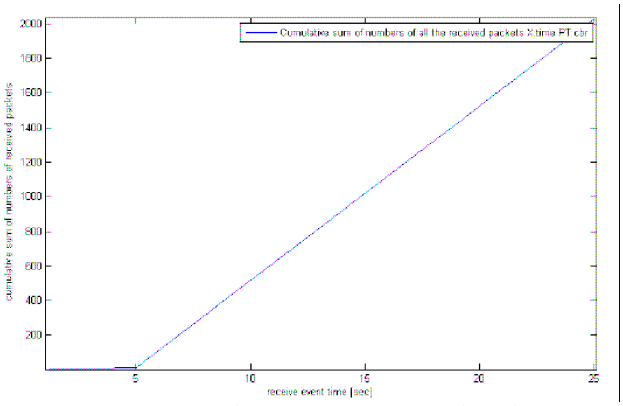

Figure 5. Cumulative sum of received packets.

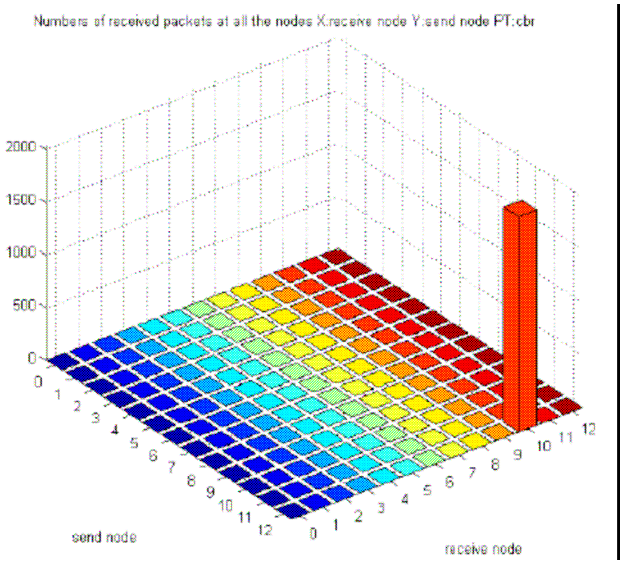

Figure 6. Cumulative sum of received packets, 3d representation.

The mobile node is node 12 in this graph, and the correspondent host is node 10. For more details on node numbering, see figure 4.

We tested the time it takes for an agent to travel across the network, depending on the number of links. Figure 7 displays the results.

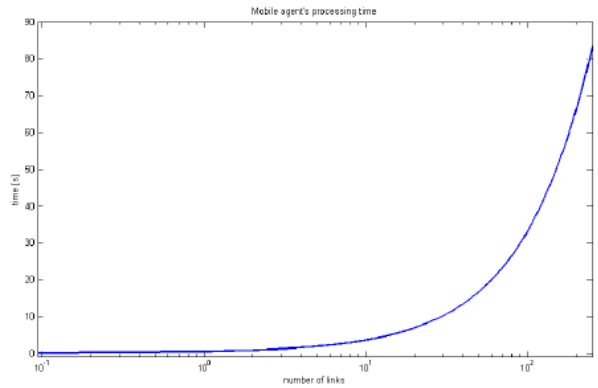

Figure 7. Mobile agent's processing time.

VI. CONCLUSIONS

This study tries to demonstrate how useful mobile agents are for managing resources in a hybrid wireless network. Mobile agents can test links before the actual application starts, selecting the optimum route. Furthermore, based on agent's measurements, the application can be reconfigured.

The most time-consuming task was installing the mobile agents patch. The patch was developed in 2003 on a different Linux version, which lead to a lot of incompatibilities. Once installed, we discovered the way a mobile agent is created and programmed.

Since we haven't analyzed jitter effects, we can compare only bandwidth and delay. As the results show, in real-time applications link delay is more important than bandwidth. The agent chose the third link which has the smallest delay even if that link had the second smallest bandwidth.

In real-life applications, there are more than four possible routes. The time it takes the agent to check all the routes and choose the best one may be too long (see figure 7) to be practical. In this case, the agent should check only a limited number of routes, even if it is possible to miss the best one.

In this phase of the project, we are able to simulate or test different network parameters, demonstrating that mobile agents can be used in network management. In order to compare the benefits of the mobile agent described before over the client-server one, we will have to simulate both scenarios and analyze trace files. Such analyses should compare the amount of traffic on the wireless interface or communication interruptions.

REFERENCES

[1] "NS-2 wiki" http://nsnam.isi.edu/nsnam/index.php/Main_Page

[2] K. Shah, "Performance Analysis of Mobile Agents in Wireless Internet Applications using Simulation", Lamar University, 2003.

[3] R. Babbar, A. Fapouwo, B. Far, "Agent-Based Resource Management in Hybrid Wireless Networks", CCECE-CCGEI 2004.

[4] H. De Meer, A. La Corte, A. Puliafito, O. Tomarcio, "Programmable Agents for Flexible QoS Management in IP Networks" IEE Journal on Selected Areas in Communication, vol. 18, No.2, 2000.

[5] S. Youssef, M. Ismail, S. Bassiouny, "Integrating Mobile Agents and Swarm Optimization for Efficient QoS management in Dynamic Programmable Networks", IEEE MECECON, 2002.

[6] S.Papavassilou, A. Puliafito, O. Tomarcio, "Mobile Agent-Based Approach foe Efficient Network Management and Resource Management: Framework and Applications", IEEE Journal on Selected Area in Communications, Vol. 20, No. 4, 2002.

[7] S. Manvi, P. Venkataram, "QoS Management by Mobile Agents in Multimedia Communication", IEEE, 2002.

[8] M. Cohen, D. Rouffet, L. Brignol, M. Coupechoux, "High speed Internet in sparsely populated areas", Alcatel-Lucent Telecommunications Review, 2004/3.

[9] S. Franklin, A. Graesser, "Is it an Agent, or just a Program? A taxonomy for Autonomous Agents", Proceedings of the Third International Workshop on Agent Theories, Architectures and Languages, Springer-Verlag, 1996.

[10] D. Roddy, "Satellite Communications 3rd edition", McGraw-Hill, 2001.