Automatic Resumption of Streaming Sessions over WiFi Using JADE

Alvaro Suárez, *Member, IAENG,* M. La-Menza, Elsa M. Macías, *Member, IAENG* and Vaidy Sunderam

***Abstract*—Media streaming technology consists of simultaneously downloading and reproducing multimedia objects from the Internet without the need to store all the information in the client's memory. This technology is very appropriate for client applications running on portable computing devices (cell phones and hand-held devices) due to their limited memory and bandwidth resources. A serious problem that has not been fully examined by other researchers is the control of temporary and intermittent disconnections due to lack of coverage, and its influence on the performance of media streaming. The aim of this paper is to describe an efficient solution based on agents running over portable computers interconnected by *Wireless Fidelity* (*WiFi*) link. In our practical experiments, the agents are in two different *Java Agent DEvelopment Framework* (*JADE*) platforms (installed in the portable computers). They encapsulate standard session control message of *Real Time Streaming Protocol* (*RTSP*) and multimedia datagrams in order to intercept the original traffic between RTSP client and server.**

***Index Terms*—Multimedia, Software Agents, Streaming, Wireless Communications, WiFi, JADE MTP Performance, Proactive Buffering.**

I. Introduction

Media streaming is a technique used to download media objects from the Internet to any kind of terminal device. Since multimedia objects need not be stored in the device's local memory, it is especially appropriate for new portable communication gadgets: mp3 players, mobile phones, *Personal Digital Assistant* (*PDA*), navigation devices based on *Global Positioning Systems* (*GPS*), and so on. These gadgets usually include wireless communications from fabric.

A standard protocol used for media streaming is *Real Time Streaming Protocol* (*RTSP*) [1]. But other standards exist, including *HiperText Transfer Protocol* (*HTTP*) [2], *TCP Friendly* [3], *Stream Control Transmission Protocol* (*SCTP*) [4] or proprietary protocols like *Microsoft© Media Services* (*MMS*). All of them are session oriented protocols (while the user is downloading a media object, a session is opened in the media server). If the communication between the client (user) and the server is broken during transmission for a certain periods of time, the server closes the session.

The unpredictable behaviour of wireless channels over time [5] (all technologies suffer this problem and specially *WiFi* [6]) provokes unpredictable loss of coverage impeding the client terminals (users) communication abilities temporarily. This means there is a high probability that the client would be disconnected during transmission if coverage is lost for certain time periods. In this case, the communication is broken and the server decides to close the session (affecting time-off and jump distance [7] negatively). Otherwise, the server will maintain open sessions for an indeterminate amount of time (it cannot know when the client will reconnect or even if it will do so) with the consequent expense of server resources. The worst scenario is that when the client reconnects it must start receiving again the media object from its beginning. Then the media streaming technique contradicts itself producing a high loss of performance.

We think this is a challenging problem that must be efficiently solved: it must be ensured that the client would receive the lost information (while it was out of coverage) and continue receiving at the point the server is reproducing the media object when reconnects.

Some application level solutions to this problem have been explored. In the Janus Project [8] a software framework is proposed that detects and predicts impending "trouble spots" when a mobile device is moving through a wireless network measuring signal strength and quality, network latency, and packet loss. Applications may use this framework to detect regions of degraded network quality and take compensatory action, resulting in enhanced effectiveness. In [9] is presented a distributed software architecture that tries to predict the behaviour of wireless network that parallel distributed applications use in order to adapt their execution to avoid communication failures when processes use synchronous messages and exhibit data dependencies (only after receiving

Manuscript received August 31, 2006. An earlier version of this paper was published in IWWN'06 (IMECS'06): "Automatic Resumption of Streaming Sessions over Wireless Communications Using Agents", pp. 926-931, Hong Kong, June 2006. This work was supported in part by the Spanish CICYT (MEC) and European Research Development Fund (FEDER) under Grant TSI2005-07764-C02-02, and FEDER and The Canaries Regional Education under contract PI042004/164.

Alvaro Suarez is with GAC (Grupo de Arquitectura y Concurrencia), Departamento de Ingeniería Telemática, Universidad de Las Palmas de G. C., Campus Universitario de Tafira, 35017 Las Palmas de G.C, SPAIN. (phone: 34 9 28 451239; fax: 34 9 28 451380; e-mail: asuarez@dit.ulpgc.es).

Mario La-Menza is with INERZA, Las Palmas de G.C., SPAIN (e-mail: mario.lamenza@gmail.com).

Elsa Macias, is with GAC (Grupo de Arquitectura y Concurrencia), Departamento de Ingeniería Telemática, Universidad de Las Palmas de G. C., Campus Universitario de Tafira, 35017 Las Palmas de G.C, SPAIN. (e-mail: emacias@dit.ulpgc.es).

Vaidy Sunderam is with Department of Math and Computer Science, Emory University, Atlanta, USA (e-mail: vss@mathcs.emory.edu).

data, they can proceed computing new data). Recently, in [10] the authors present ongoing work focusing on a way of achieving better video service quality based on an adaptive layer, taking into account the movement of the user and the constraints of their new location. They try to adapt the rate when a mobile user changes from one network technology, for example UMTS, to another one, e.g. WiFi.

On the other hand, at the application level software agents [11], proactive buffering [12] and proxies [13] can also be used to control intermittent disconnections. The seminal paper [14] presents a distributed software agent architecture in which a client terminal agent communicates with a fixed proxy agent to negotiate some parameters of Internet access, moving the client agent to the fixed network when a disconnection is detected. In [15] it is described the implementation of a web-based multi-agent system specifically designed for users on the move, to stream multimedia content to a hand-held device. They don't set a particular network infrastructure. In [16] the authors propose a mobile agent based ubiquitous multimedia middleware to solve intermittent disconnections provoked by handover of WiFi devices. They propose a two-level proactive buffering (one buffer in the edge wired proxy and the other in the mobile device). To our knowledge there has been no effort that encapsulates RTSP messages using the multiagent transport message system and also that uses proactive buffers to solve the intermittent disconnections problem. In this paper we explore the usage of JADE [17] that is an open source *Multi Agent System* (*MAS*) that fulfils totally the recommendations of *Foundation for Intelligent Physical Agents* (*FIPA*) [18]. JADE is an interesting platform since it includes the *Lightweight Extensible Agent Platform* (*LEAP*) that is being used to program PDA and mobile telephones [19]. Up to our knowledge, JADE performance has not been studied for resuming streaming sessions automatically. Another important topic is to study the performance of the JADE platform and its *Message Transport Protocol* (*MTP*) that could help to provide a high level solution to intermittent disconnections.

In this paper we present a proactive control mechanism of intermittent disconnections of a WiFi link based on proactive buffers managed by JADE agents (one for the client and another one for the RTSP server). We have demonstrated our results executing the software on two portable computers interconnected by a WiFi link. Initially we observe that JADE's performance is suitable and that the behaviour of the MTP is good enough.

The rest of the paper is organized as follows. In section 2 we outline the important characteristics of MAS and our motivation to use it to solve problems derived from the intermittent wireless disconnections. Section 3 is devoted to present our proposed software solution. In section 4 is presented deeper details of the implementation and some performance measurements. Some advantages are presented in section 5. Finally, we summarize our conclusions and present directions for further research.

II. THE BENEFITS OF USING MAS

To our knowledge the main characteristics of software agents (autonomy, learning, proactiveness, scalability, flexibility and reliability) have not been studied to obtain smart streaming distribution methods in wireless networks using their natural message passing systems to encapsulate the RTSP control messages. We think agents could collaborate in predicting the behaviour of the wireless communication to avoid (or at least to reduce) the impact of communication performance degradation in the presence of intermittent wireless disconnections.

One of the benefits of using a MAS is that it simplifies the application design. This is because the services offered by the MAS platform allow a high level of abstraction for the design of the messenger details and/or the *Remote Method Invocation* (*RMI*). In this way the application designer has only to address the implementation of the business rules (video streaming in our case).

Other benefits are the principal advantages that MAS offers: modularity (reduces the complexity of software design), scalability, flexibility (adding or removing agents is very simple) and pre-established behavior of the agents (permits the agents to exhibit a behavior both proactive and reactive).

In order to take advantage of the above benefits it is important to use a MAS platform that includes all of them. JADE is a single platform consisting of one or more containers that can be located on different hosts with a unique main container executing the RMI service. It offers different implementations depending on the availability of resources of the device to be used, which makes it an attractive option. A solution which works well on *Java 2 Platform, Standard Edition* (*J2SE*) has a good chance of working equally well on *Java 2 Platform, Micro Edition* (*J2ME*), using LEAP. An additional benefit is that the MTP of JADE guarantees communication between resident agents in distributed containers, thereby offering failure recovery services. In case of disconnection, it carries out a periodic checkup to monitor the reconnection. Once the reconnection is achieved, it communicates the state change by sending messages to the agents; so that they can restart the dialogue from where it was left off (we tested this in our practical experiments).

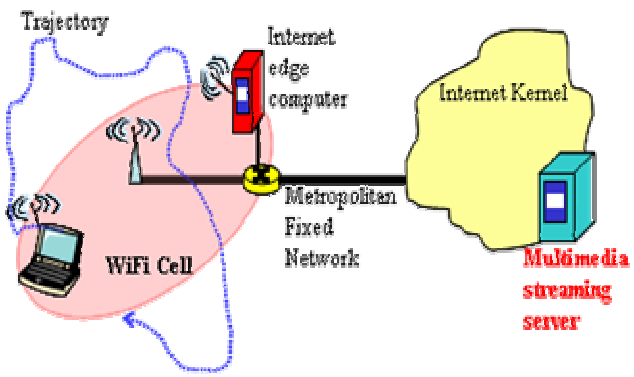


Fig.1. General architecture for the mechanism.

III. THE AUTOMATIC RESUMING MECHANISM

Fig. 1 shows the general architecture for the implementation of our mechanism. The access network consists of a wireles cell in which any of the mobile gadgets could be connected (for example using WiFi technology). We suppose an Internet edge router to which a WiFi Access Point and a computer (to which we denominate Internet edge computer) are connected. The edge router is connected to a Metropolitan Fixed Network that accesses to the Fixed Internet in which the multimedia streaming server is connected. We consider a portable computer that implements the multimedia streaming client using RTSP signaling and *Real Time Protocol* (*RTP*) for media datagram transmission. This client uses the WiFi link to access the multimedia streaming server.

One of the objectives of this paper is to evaluate the performance of the MTP of JADE to observe the efficiency of the recovery of the connection in case of failure of the wireless channel or loss of the connection due to the trajectory of the mobile gadget. In order to be able to recover the original RTSP/RTP Client-Server communication using the MTP is necessary to encapsulate the communication in messages using FIPA *Agent Communication Language* (*ACL*). Re-programming all the commercial clients and servers (in order to encapsulate the communication) which exist in the market is not possible and either desirable because its source code is not open or the reprogramming task could be very arduous. A better, more efficient and quicker viable solution consists of programming independent agents which intercept the client-server messages. They will be in charge to detect wireless communication failures and to recompose the messages suitably.

An important design parameter is where initially install the agents (JADE platforms). They must resolve the WiFi intermittent disconnection and do the automatic resumption of sessions. As is reasonable, installing an agent in the Internet edge computer and another one in the portable computer, they can cooperate discovering when the portable computer is out of coverage simply using MTP signaling. We name the agent in the portable computer as *Agent Client Proxy* (*APC*) that redirects the communications to the *Agent Server Proxy* (*APS*) hosted in the Internet edge computer. The streaming client uses the APC as a navigation proxy. APC intercepts RTSP/RTP messages and encapsulates them on *INFORM* (for RTSP) and *PROPOSE* (for RTP serialized media objects) kind of FIPA ACL messages. APS receives these encapsulated messages (using unrealiable WiFi link) and converts them to the original RTSP/RTP messages sent by the client. These messages are redirected to the multimedia streaming server (using reliable cable link to the egde router). We deal with JADE performance and for that reason, we maintain the communication between agents in this simple way not defining ontology. We only need to install these agents (under JADE platforms) in these computers in a secure way. It is not neccesary to move the agents, so we don't need to consider the security issues explained in [20].

Finally, another important design parameter is what the agents must do when a disconnection is discovered. This implies the temporary proactive buffering of multimedia data in the APS.

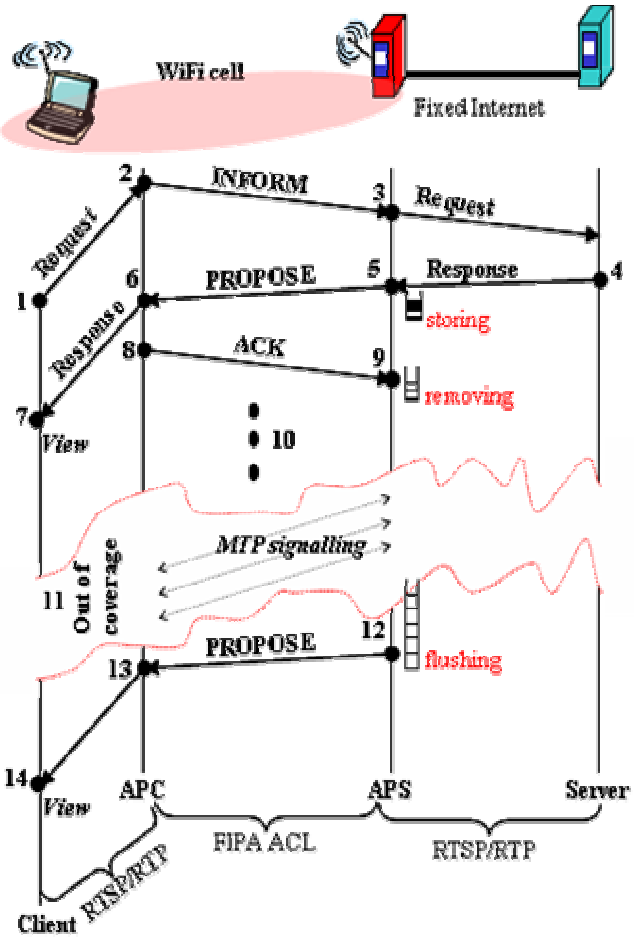


Fig. 2. Actions to automatic resuming streaming sessions.

Fig. 2 shows the main steps of the resumption mechanism to solve the intermittent disconnections described in section I.

Firstly (action marked as 1), the client issues a RTSP request (RTSP "*SETUP URL*" message). Secondly, APC (2) encapsulates it in an INFORM kind of message and redirects it to the APS which sends (3) it to the server (again as a RTSP "*SETUP URL*" message). After the server receives that message, it sends back a response (i.e. RTSP "*200 OK*"). We then suppose that in step 4 the server has received a RTSP *PLAY* order and it sends back data using RTP protocol (after sending a RTSP "*200 OK*" message). Those data are intercepted by APS which stores them temporarily (5) in a well dimensioned buffer and also sends them to the APC using a *PROPOSE* kind of message. In this point the APC receives them (6) and redirects the media datagrams to the client (that could present them to the user, step 7), but also sends back an *ACK* kind of message to the APS (8) to confirm the reception of the last block of media data received. This message is encapsulated in an INFORM message. Once APS receives the

ACK kind of message it will remove the acknowledged data in the proactive buffer (marked as an empty buffer in step 9). These steps could be repeated for the reminder blocks of media data (10). Let us note that actions 7 and 9 are overlapped with the storing of new data in action 5. In this way we hide the possible overhead of our mechanism. This set of actions could be achieved till the end of the streaming process.

Let us examine the actions to deal with intermittent disconnections of the portable WiFi client (11) affecting multimedia data communication.

The MTP periodically polls in steps 8 and 9 the APC and APS to test if they are still alive. The APS adds the temporal storing of media frames in its buffer. The exact maximun size of this buffer depends on the amount of time the client is disconnected (with a powered server machine we can support several minutes of disconecction) and the resolution of media frames (compression technique). In practice an intermittent disconnection could last 1 minute (in some cases, more than this time is not practical because the user can't follow the plot easily). So our mechanism is scalable (a relatively large amount of wireless clients can be supported by the same APS).

Let us take into account that in step 5 the frames APS sends are also stored in its buffer and that they are removed if it will receive an ACK response. When the portable client is out of coverage the ACK response doesn't arrive at APS. In this case and while it is out of coverage the media frames will be stored in the buffer. One of the main benefits of using JADE is that MTP will try to reconnect the APS and APC during some time. MTP allows the programmer to specify a time out whose default value is 1 minute but it can be set to hours or days indeed. This time out also influences efficient buffer dimensioning. Using this time out MTP takes care of message delivery. If the communication is interrupted, the MTP will retry the delivery at regular intervals. When the message is finally received by the APC, MTP sends a message to APS.

Let us note that intermittent disconnections while the APC is sending a RTSP signaling command will be directly recovered by the RTSP semantic. For example, if the client issues a RTSP "*PLAY*" command while it is out of coverage, it will not arrive to the server. After a certain time-out the client will repeat the command because no answer is received. On the contrary, if the RTSP "*PLAY*" command arrives to the server, but its associated answer does not arrive to the client (just in that instant of time it is out of coverage), then the APS will store the associated media data until the client is connected again.

Once the connection between agents has been re-established, APS will flush the buffer (12) reading the ordered media frames in the buffer and sending them to the APC. APC simply redirects the frames to the client (13). In this way, the wireless client could recuperate the stream just at the point it lost the vision (14).

Finally it is important to remark that the server does not receive a RTSP "*TEARDOWN*" command from the client when it goes out of coverage (APS is always alive and receiving from the server). So the original session is still open and the client can continue with the streaming.

IV. Performance Evaluation of Experimental Results

We have adapted the code for the client and server found in [21]. In this implementation the server held a single thread for attending only to one client. The integration of these codes into JADE platform was guided by some necessary changes: the integration with agents and the migration of some of the server functionality to the APS behavior. In particular, APC must receive the client RTSP messages, encapsulate them into ACL messages (using MTP) and then send them to the APS like a normal RTSP client. On the other hand APS has a client part that receives messages from the server and a server part to attend APC messages. It has a single thread to store the media frames in main memory. We tried to maintain the standard behavior of the client and server unchanged. This is important to connect our agents to standard RTSP clients and servers.

We used a simple movie format named *Motion Joint Photographic Expert Group* (*MJPEG*) [22]. In this format the video is stored as a sequence of JPEG images. Therefore, the payload of packages RTP consists of a JPEG image. This enormously simplifies the design of the presentation of the multimedia object in the client. The reason of using MJPEG format is that it is a very simple format and it is possible to be used in a high rank of devices: practically all of them have the capacity to decode images in JPEG format (and therefore a set chained of them).

In the scheme of the general architecture of the Fig. 1 it is shown that in order to accede to the multimedia streaming server it would be necessary to cross the Kernel of Internet. Determining mechanisms to provide good performance of the streaming technique in the Kernel of Internet with wireless access is a problem that not yet has optimal solution [23]. Since we are interested in the study of the performance of the streaming technique in the portion of wireless access network in the presence of sporadic disconnections, we installed the APS and the multimedia streaming server in the Internet edge computer (that it is also a portable computer). The portable computer that hosts the client and the APC and the portable computer that hosts the server and the APS, communicate using the WiFi link. This allows us to focus on the study of our problem avoiding the influence of the fixed network. The hardware characteristics of the client portable computer are: 1.6 GHz Pentium Centrino Processor, 512 MB of *Random Access Memory* (*RAM*), 54 Mbps WiFi *Network Interface Card* (*NIC*). It is connected to a 54 Mbps WiFi *Access Point*. The server and the APS were installed on a portable computer with the following characteristics: 1.2 GHz Pentium mobile Processor, 512 MB RAM and 11 Mbps WiFi NIC. We used the *Integrated Development Environment* (*IDE*) Eclipse 3.0.1 [24] for implementing the Java code over *Java Virtual Machine* (*JVM*) 1.4.2 [25]. JADE was installed in both portable computers using Eclipse. MTP takes care of ACL message passing between agents. We have used the *Internet Inter-Object-request-broker Protocol* (*IIOP*) *HypertText*

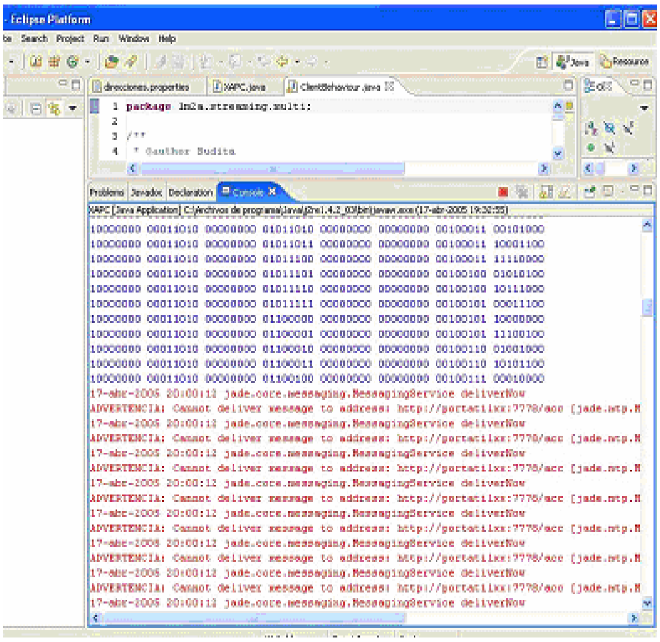


Fig.3. Eclipse IDE showing that the messages could not be delivered.

Transfer Protocol (*HTTP*) for JADE interplatform communication among different machines because it provides good performance [26].

In JADE 3.3 MTP could produce an *MTPException* when communication errors occur. In Fig. 3 is shown, on the *Graphical User Interface* (*GUI*) of the Eclipse IDE, that exactly takes place a series of warning messages indicating that the communication has been interrupted ("*cannot deliver message to address* …").

From the agents point of view this exception is translated into a FAILURE ACL message sent back to the agents. We had to find a new version of MTP because the original one did not throw all the errors. In particular when an agent sent messages to a platform that was not on-line, HTTP MTP produced a *SocketException* which was not transmitted to the platform. The result was that agents had no way to decide if the message was received or not. Using the new version of MTP implementation all failures cases are transmitted. In Fig. 4 we show a graphical example of that. It is shown the GUI for debugging applications of JADE denominated *Remote Monitoring Agent* (*RMA*). The left part shows the agents included in the GUI by default: *Agent Management System* (*AMS*) that provides an integrated environment with several services (formulated like agents also) for the agents of the platform, the *Sniffer* (that carries out the debugging of the application) and the *Directory Facilitator* (*DF*) that provides a service of "yellow pages" for the agents known by the platform. The agent *XAPC* that corresponds with the APC is marked for debugging. In the right part of the GUI are the different messages of error and recovery of the communication. Note that these messages are classified between the messages sent or received by APC and the other agents. After several errors of communications (and corresponding retries of MTP) the reconnection is achieved when wireless connection is recovered (as indicates the Windows warning message).

In the above scenario, a question rapidly raised: is it JADE appropriate for implementing the proxies as agents? To answer it we considered basically three different aspects: a) to observe the *JVM* memory consumption, b) to measure the frequency of issuing of frames in the server and c) to measure the jitter.

In the experimental results we measured the above associated parameters (aspects a, b and c) executing up to five times the streaming software considering only Java implementation of the client and the server. After that we repeated the execution of the same client and server (slightly modified) cooperating with the JADE agent proxies. In all these executions we maintained the overall workload of the client, server and proxy machines constant for not distorting the experimental results.

We did several tests moving the client portable computer out of coverage and returning after one minute (at most). We probed that without our software the streaming sessions were never resumed (the client lost definitively the session and a new one had to be restarted). Doing the same movements but using our agent based proxies the streaming sessions always were resumed at just the point the client lost the vision.

In Fig. 5 we show the JVM memory size increasing when the JADE agents executed their actions. The vertical coordinate represents the increase in *Mega Bytes* (*MB*). The percentage of increase in the client portable computer is 44.64 % and in the server is 56.9 %. Let us note that measuring the exact quantification of memory use (with or without JADE agents) does not turn out to be simple given that Java hides the internal details of its memory management system, and the garbage collector. For that reason we observed the overall size of JVM (although it is simple to characterize the maximum size of the frames buffer it can not be observed in a stand alone way). In every measurement we included the size of the Eclipse programming environment (this is the reason why the size of the client and server without agents is 61 and 55.3 MB respectively). Obviously in a PDA or mobile phone using LEAP only these sizes are much smaller.

In Fig. 6 we present the experimental results for frames issuing and jitter when no agents are used. We have chosen 5 movie visualizations. We chose, for each visualization, an interval of consecutive frames issuing samples (the same interval for every visualization). The behaviour for all the samples is similar to the samples of this interval. We measured their issuing time. In Fig. 6.a it is shown that the frames where issued periodically (in vertical axis we present the accumulated instant of time of frames issuing, in the horizontal axis we present the different samples numbering). In Fig. 6.b the jitter is presented (the time the frames are received in the client to be presented). In general the jitter was linear with minor variation due to the WiFi channel properties. It is important to note that almost all visualizations presented the same characteristics (the distance among different curves is very small). The delay is also very small and due to the traffic generated is *Constant Bit Rate* (*CBR*) the *Mean Opinion Square* (*MOS*) is good (we have not measured it quantitatively). We agree [15] that CBR traffic

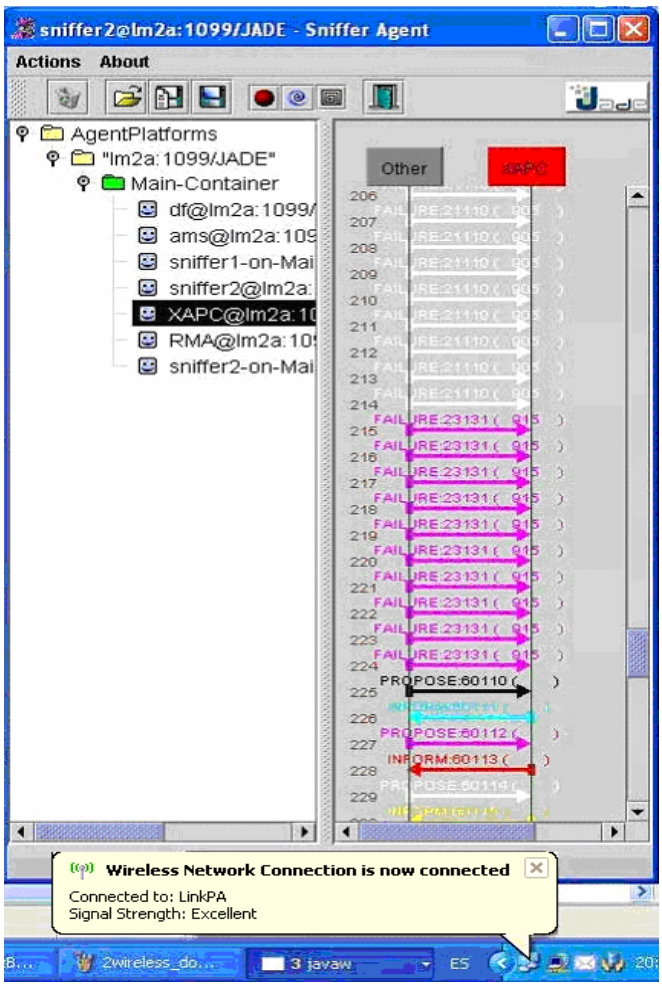


Fig.4. The Platform verifies the connection and communicates the event to the sender agent.

is
the most suitable under certain environment conditions (constant bandwidth and known multimedia object size).

In Fig. 7 we present the same interval of samples for which we measured the same metrics as Fig. 6, but now for the client and server with agents. It is important to remember that now an additional encapsulation of messages in APC and APS is done. Moreover MTP now takes care of disconnections in the client portable computer (remember that for our experiment we hosted the server in another portable computer). For these reasons now the distance among different curves of frames issuing and jitter are greater than the above results. The important results are that the frames issuing and jitter are almost linear also and that the delay is good.

The software, a film coded in MJPEG and related information that we use in our experiments can be found in the server of Internet http://prai.inerza.com/sm/vpa/ (a mirror is: http://guigui.teleco.ulpgc.es:8028/TSI2005-07764-C02-01/otros-doc-y-herram/).

V. ADVANTAGES OF OUR MECHANISM

In this section we present some additional benefits of using agents to program our proxies, some advantages of our mechanism and some comparisons with other strategies.

The use of a MAS for programming proxies is more efficient and also a more elegant programming style compared to the strategy followed in [14].

In [27] it is proposed a RTSP "*PING*" command to prevent the identified session from being timed out. This aproximation is used in [28] where the client usually pools the server liveness. We use the MTP properties to clearly include the semantic of this operation transparently in the behaviour of the agents (but no especial code must be programmed for that). We use this behaviour to control the sporadic intermittent disconnections of the client.

A typical solution to intermittent disconnections (for Video on Demand) consists in always storing a relatively long buffer in the client machine [29]. This technique is difficult to apply to real time streaming and also is impossible to be used for light client machines like current certain kind of PDA and mobile phones with reduced main memory capacity (less than 64 MB both for applications and user data). In this case a proxy like APS is strongly recommended. Moreover, the APS buffer could be shared by different clients in the WLAN that are simultaneously out of coverage.

Our mechanism is directly and easily scalable to heterogeneous client machines: The APS could collaborate with an APC implemented in a light client machine to manage the buffer, but also could delegate this management to an APC implemented in fat clients (portable computers). Also it is easy to include a two-level proactive buffering like the one in [16] taking into account that the management of our buffers is most complex because the profiles of handover between WiFi cells are not available for us.

The authors of [30] could easily introduce our mechanism in their proxies to improve their software component based architecture.

In [31] is proposed a typical end-to-end solution that tries to adapt the speed of transmission between the client (wired or a

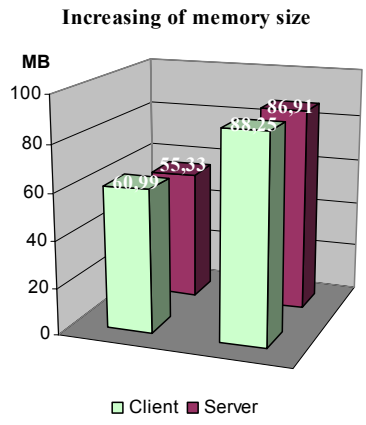


Fig.5. Increase in memory using agents.

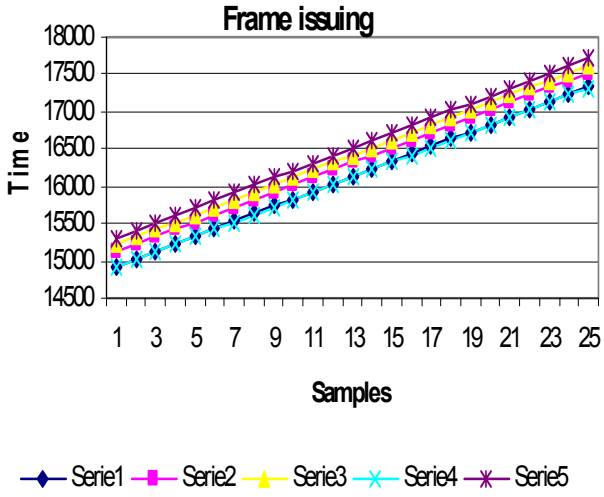
wireless) and the multimedia streaming server to Internet Kernel congestion and wireless channel characteristics (loss of packets, fading, and so on). Although this is a well tested solution it does not consider the case for intermittent sporadic client disconnections, because it is impossible to anticipate information to the model they use for discovering when the portable client could be out of coverage. This is not our case because we use proactive buffering.

Up to our knowledge, there are few papers in the literature that evaluate the JADE performance. All we have found evaluate JADE MTP performance over a 100 Mbps Ethernet, but no one evaluates its performance over WiFi. In [26] they state that JADE MTP scales linearly in several testbed scenarios. Their measurements showed a good JADE message system performance and it was a good candidate for developing heavy-load distributed applications. In [32] (that is presented as a work that tries to complete the [26]), it is concluded that the JADE MTP does not scale well for a big amount of agents in their testbed, and in some cases paradoxical results were obtained. This could be because they use a very big amount of agents that could not be well balanced. We are not interested in using a big amount of agents, because in a WiFi cell it is not recommended to use thousands of WiFi gadgets. In [33] the authors present the practical experiments about start up, agent migration and interaction with FIPA agents under several Operating systems (Linux and Windows) and *Java Runtime Environment* (*JRE*) 1.4.2 and 1.5 showing a linear scalability of the JADE platform. We have tested the JADE performance of interplatforms communication in presence of WiFi intermittent disconections for a streaming application under Windows® operating system.

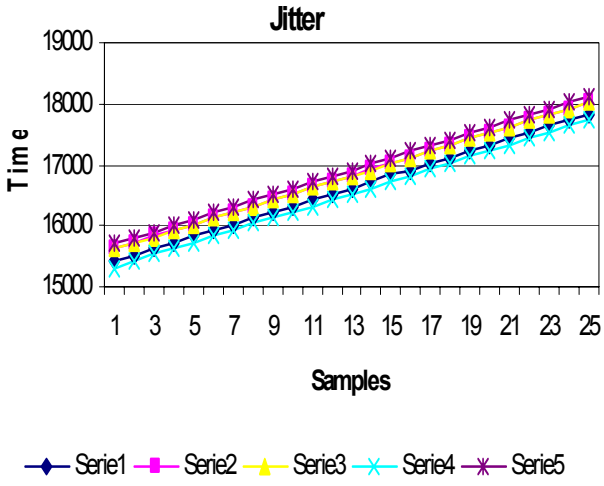
In [34] is presented a transactional system that counts with a software object that senses the channel state. Using a profile the applications can use adaptable time-outs that indicate them when they can send the data and when they must wait until the channel is ready to communicate their data. They test their methodology in a best effort 100 Mbps Ehternet in which long delays were obtained when a big amount of stations are communicating (from the point of view of an individual station this could be equivalent to a logical disconnection). This model can not be applied to multimedia streaming applications and to applications in which there are data dependencies between the client and the server.

VI. Conclusions and Future Work

In the last years several protocols and mechanisms have been elaborated to support mobile transactions in the presence of unreliable wireless channels. Nevertheless, the execution of distributed applications with firm real time requirements like those of streaming multimedia on unreliable networks is a problem that has still not been solved efficiently. The difficulty is in controlling the intermittent disconnections because the client can leave coverage at any time and lose the multimedia information reception causing serious problems of inefficiency to certain protocols like RTSP.



(a)



(b)

Fig. 6. Measurement of frames issuing time (a) and jitter (b) for client and server without agents.

The technology of agents offers an attractive environment for programming media streaming applications over unreliable wireless links. It offers a real alternative for controlling information lost during coverage breaks.

In this paper we have proposed an automatic resumption mechanism to recover RTSP sessions based in JADE. JADE MTP is the one in charge to recover the communications after a disconnection of the wireless client. The APS simply must maintain a proactive buffer in which it stores the pending datagrams of being received by the APC. One important advantage is that the semantics of the RTSP is not altered. Our mechanism can be used for any type of RTSP client and server. The experimental results show the convenience of using JADE to solve the intermittent disconnections problem in wireless

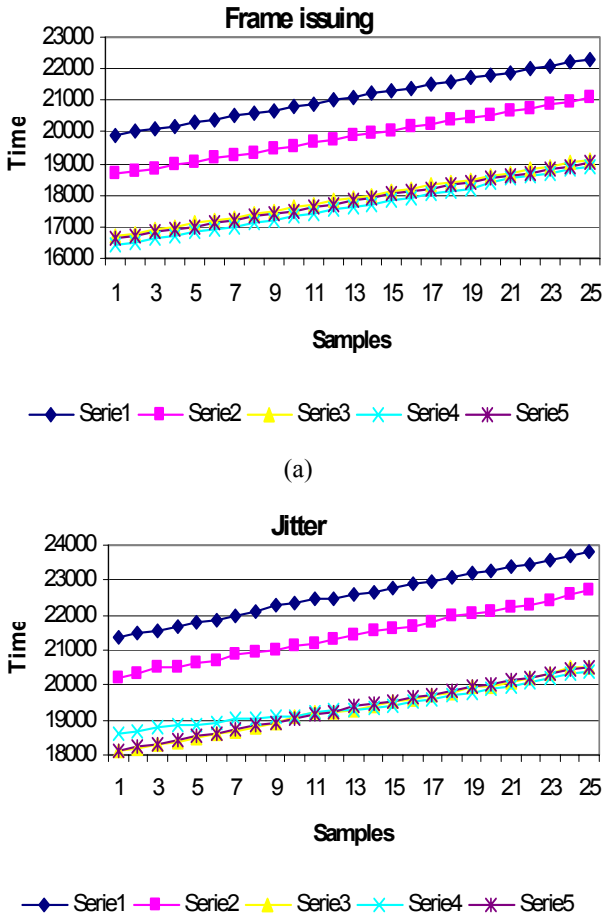


Fig. 7. Measurement of frames issuing time (a) and jitter (b) for client and server with agents.

network.

This seminal work must be improved considerably to definitively show that JADE is appropriate and better than other technologies to solve efficiently this problem.

For example in our implementation the server held a single thread storing the packages directly in main memory. But in a real situation a thread should be created for each client with coverage lost. Based on a study of the RAM/persistent mix, an optimum number of media frames to be stored in memory should be defined.

We have used the MJPEG format, but there are other standards that are more used in practice. The usage of the codecs for this other formats is strongly recommended for improving our software. Others problems will raise like complexity in media frames generation, large latency and large jitter which complicate the storing of frames and its recovery.

REFERENCES

[1] H. Schulzrinne, A. Rao, R. Lanphier, "Real Time Streaming Protocol (RTSP)", Request for Comments: 2326, Standards Track, 1998.

[2] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, "HiperText Transfer Protocol (HTTP/1.1)", Request for Comments: 2616, Standards Track, 1999.

[3] Deepak Bansal, Hari Balakrishnan, "TCP-friendly Congestion Control for Real-time Streaming Applications", MIT Technical Report, MIT-LCS-TR806, May 2000.

[4] Randall R. Stewart, Qiaobing Xie, *Stream Control Transmission Protocol (SCTP): A Reference Guide*. Addison-Wesley, 2001.

[5] David Tse, Pramod Viswanath, *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

[6] IEEE P802.11, "The Working Group for Wireless LANs", Available: http://grouper.ieee.org/groups/802/11/.

[7] Costa Cristiano, Cunha Italo, Borges Alex, Ramos Claudiney, Rocha Marcus, Almeida Jussara, Ribeiro-Neto Berthier, "Analyzing Client Interactivity in Streaming Media", *13th ACM International Conference on Wide World Web Conference*, May 2004, pp. 534-543.

[8] G. Tonev, V. Sunderam, R. Loader, J. Pascoe, "Location and Network Quality Issues in Local Area Wireless Networks", *Proc. Intl. Conference on Architecture of Computing Systems: Trends in Network and Pervasive Computing (ARCS 2002)*, April 2002.

[9] Elsa M. Macías, Álvaro Suárez, Vaidy Sunderam, "Efficient Monitoring to Detect Wireless Channel Failures for MPI Programs", *12th Euromicro Workshop on Parallel, Distributed and Network-Based Processing (PDP 2004)*, IEEE Computer Society, 11-13 February 2004, pp. 374-381.

[10] Daniel Negru, Ahmed Mehaoua, "Adaptive Layer Design for Video Multimedia Services in a Mobile Environment", *Proceedings of the 2005 ACM Conference on Emerging Network Experiment and Technology*, October 2005, pp. 270-271.

[11] Lixin Gao, Zhi-Li Zhang, Don Towsley, "Proxy-Assisted Techniques for Delivering Continuous Multimedia Streams", *IEEE/ACM Transactions on Networking*, Vol. 11, No. 6, December 2003, pp. 884-894.

[12] Bobby Vandalore, Wu-Chi Feng, Raj Jain, Sonia Fahmy , "A Survey of Application Layer Techniques for Adaptive Streaming of Multimedia", *OSU Technical Report, OSU-CISRC-5/99-TR14, 1999, 19 pags. (An extended version published in: Real-Time Imaging, Vol. 7, No. 3, 2001, p. 221-235.)*

[13] Paolo Bellavista, Antonio Corradi, Carlo Giannelli, "Mobile Proxies for Proactive Buffering in Wireless Internet Multimedia Streaming", *Proceedings of the IEEE International Conference on Distributed Computing Systems Workshop (ICDCSW'05)*, June 2005, pp. 297-304.

[14] Heikki Helin, Laamanen Heimo, Raatikainen Kimmo, "Mobile Agent Communication in Wireless Networks", 5th European Wireless Conference, Available: http://www.cs.helsinki.fi/research/monads/papers/wireless99/agent_comm.pdf.

[15] G. Leetch, E. Mangina, "A Multi-Agent System to Stream Multimedia to Handheld Devices", *Proceedings of the Sixth International Conference on Computational Intelligence and Multimedia Applications (ICCIMA'05)*, August 2005, pp. 2-10.

[16] P. Bellavista, A. Corradi, L. Foschini, "Java-Based Proactive Buffering for Multimedia Streaming Continuity in the Wireless Internet", *Proceedings of the Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks (WoWMoM'05)*, June 2005, pp. 448-450.

[17] F. Bellifemine, G. Caire, A. Poggi, G. Rimassa, "JADE, A White Paper", *Journal of Telecom Italia Lab*, Vol. 3, No. 3, September 2003, pp. 6-19.

[18] FIPA, "The Foundation for Intelligent Physical Agents", Available: http://fipa.org/.

[19] M. Berger, S. Rusitschka, M. Slichlitchte, D. Toropov, M. Watzke, "Porting Agents to Small Mobile Devices – The Development of the Lightweight Agent Platform ", *Journal of Telecom Italia Lab*, Vol. 3, No. 3, September 2003, pp. 32-41.

[20] Lotfi Benachenhou, Samuel Pierre, "Protection of a Mobile Agent with a Reference Clone", *Computer Communications, 29, Elsevier*, 2006, pp. 268-278.

[21] Padma Mundur, "Multimedia Networking (CMSC 691C)", Spring 2003, Available: http://www.csee.umbc.edu/~pmundur/courses/CMSC691C/lab5-kurose-ross.html.

[22] Video Formats White Paper, "Understanding Video Formats", BroadWare Technologies Inc., Available: http://www.axis.com/adp_cd/adp_cd8/companies/broadware/Video_Format-White_Paper-v3.pdf.

[23] E. Turhan Tunali, Aylin Kantarci, Nukhet Ozbek, "Robust Quality Adaptation for Internet Video Streaming", *Multimedia Tools and Applications, 27, Springer Verlag, 2005*, pp. 431 - 448.

[24] Eclipse Contributors © (include software developed by Apache Software Foundation), "Eclipse Platform", Available: http://www.eclipse.org/platform.

[25] Tim Lindholm, Frank Yellin, *The JavaTM Virtual Machine Specification Second Edition*. Available: http://java.sun.com/docs/books/vmspec/2nd-edition/html/VMSpecTOC.doc.html.

[26] E. Cortese, F. Quanta, G. Vitaglione, "Sacalability and Performance of JADE Message Transport System", *Proceedings of the 2002 Autonomous Agents and Multiagent Systems (AAMAS) Workshop, ACM Press*, 2002.

[27] H. Schulzrinne, A. Rao, R. Lanphier, M. Westerlund, A. Narasimhan, "Real Time Streaming Protocol (RTSP)", *Internet Draft, Internet Engineering Task Force*, February 2004, Work in progress.

[28] VLC, "Video LAN", Available: http://www.videolan.org/.

[29] Real Networks site, Available: http://www.realnetworks.com/.

[30] O. Layaida, D. Hagimont, "Adaptive video streaming for embedded devices", *IEE Proceedings Software*, Volume 152, Issue 5, October 2005, pp. 238 - 244.

[31] Guang Yang, Tony Sun, Mario Gerla, M. Y. Sanadidi, Ling-Jyh Chen, "Smooth and Efficient Real-Time Video Transport in the Presence of Wireless Errors", *ACM Transactions on Multimedia Computing, Communications, and Applications, Vol. 2, No. 2, ACM Press*, May 2006, pp. 109-126.

[32] K. Chmiel, D. Tomiak, M. Gawinecki, P. Karczmarek, M. Szymczak, M. Paprzycki, "Testing the Efficiency of JADE Agent Platform", *Proceedings of the IEEE Third International Symposium on/Algorithms, Models and Tools for Parallel Computing on Heterogeneous Networks (IEEE Third International Workshop on Parallel and Distributed Computing), IEEE Computer Society*, 2004, pp. 49-56.

[33] F. Lo Piccolo, G. Bianchi, S. Salsano, "Measurement Study of the Mobile Agent JADE Platform", *Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2006), IEEE Computer Society*, June 2006, pp. 638-646.

[34] N. Santos, P. Ferreira, "Making Distributed Transactions Resilient to Intermittent Network Connections", *Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2006), IEEE Computer Society*, June 2006, pp. 627-632.