Deploying new QoS aware transport services

Exposito E. [1,2] , Sénac P. [1,2] , Garduno D [2]., Diaz M. [2], Urueña M. [3]

[1] ENSICA, DMI 1 Place Emile Blouin 31056, Toulouse Cedex, France
{ernesto.exposito, patrick.senac}@ensica.fr
[2] LAAS du CNRS, 7 Avenue du Colonel Roche, 31077, Cedex 4, Toulouse, France
{dgarduno,michel.diaz}@laas.fr
[3] Universidad Carlos III de Madrid, Leganés 28911, Madrid, Espagne
{muruenya}@it.uc3m.es

Abstract. Traditional protocols as TCP and UDP propose a very restricted vision of the quality of service notion. These limitations that restrict the spreading out of distributed multimedia application led to us to define a new generic transport protocols generation instantiable from the applicative quality of service requirements. However, the introduction of a new transport protocol has to answer the wide scale deployment questions. This paper proposes a networking architecture based on the concept of active networks that makes possible the automatic and transparent deployment of advanced end to end communications services. The proposed approach has been successfully experimented on top of a large scale European Networking Infrastructure designed in the framework of the GCAP European project.

Key words: Quality of Service, transport protocols and services, active networks, multimedia applications.

1. Introduction

Advances in software engineering techniques open the door to a promising industry of distributed multimedia applications and software components. However, this parallel and steady evolution of hardware and software has not been followed by a corresponding progress of transport protocols and services. Therefore, there is at the present time a wide gap between applications requirements and services delivered by TCP and UDP that are the transport widely used today on top of IP. Moreover, the introduction of quality of service (QoS) oriented network mechanisms by the Integrated and Differentiated architectures underline the lack of a new generation of transport protocols that would make possible a mapping of application layer QoS needs onto an efficient combination of QoS aware transport-network services. As a result, current multimedia applications implement their own quality of service control mechanisms at the expense of a great increase in programming complexity.
Standard transport protocols (.i.e. TCP and UDP) propose a very restrictive vision of the quality of service notion. Indeed, the quality of service offered by these protocols follows an everything-or-nothing approach, based on two fundamental parameters of the quality of service that are the order and the reliability. However, multimedia flows such as MPEG streams as well as SMIL or MPEG-4 multimedia components, have

reliability and continuity constraints which adapt badly as well to a fully reliable and ordered service, as to a service that offers no guarantee of reliability or order [9]. Indeed, we have previously shown that fundamental QoS parameter of multimedia components and flows can be modeled with partial order relations [10]. We have proposed a formal framework that allows not only the order and reliability constraints associated to the multimedia flows to be modeled, but also the temporal constraints of these media. This formal approach make possible the derivation of fundamental application layer QoS constraints towards transport services and protocols. This new generation of generic transport protocol is formally instantiated from application level requirements and delivers to users a service that complies with application layer requirements while improving, in a very sensible way the use of network resources (i.e. bandwidth and buffers). This new approach has been at the origin of new transport protocols proposed in the context of the IETF, such as SCTP in order to offer a reliable and partial order service [14]. Nevertheless, except the works presented in this article, it does not exist presently a transport protocol that take into account, simultaneously and in a complete way, the fundamental constraints of time, reliability and order associated to the multimedia components and flows.

This new approach that we propose to assure the end-to-end quality of service management led a new generation of generic transport protocols, called "Fully Programmable Generic Transport Protocols" or FPTP, that can be simply and directly instantiated from the applicative quality of service constraints [13]. With regard to the space covering all the services susceptible to be delivered by the FPTP protocol, a specification of service defines the subspace of all the acceptable services that conforms to the service user needs (see figure 1).

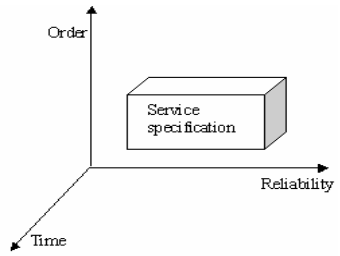

Fig. 1. Services specification in the FPTP protocol space

The gains provided by these new transport protocols generation, compared to TCP and UDP, have been demonstrated by means of two C and Java implementations designed respectively as libraries or packages accessible in the user mode [1]. From an experimental viewpoint, we have shown that the FPTP approach allows the quality of service perceived by a user acceding to MPEG remote flows to be sensibly improved compared to TCP and UDP.

However, the introduction of a new protocol comes along with the critical problem of its dissemination. This paper proposes an original solution to this problem based on the concept of active networks. This approach has the merit of being completely transparent to the applications without demanding any changes of the traditional development technologies.

Next sections are structured as follows. Section 2 gives a short introduction of active networks and introduces the SARA active platform. In section 3, we propose a networking architecture capable to assure the transparent deployment of FPTP services on the SARA active platform. In the last section, we describe several experiments, done in the framework of the GCAP European project (Global Communication Architecture and Protocols), aiming to validate the proposed architecture.

2. Active networks

Traditional data networks passively transport payload bits from one end-system to another. In the current Internet, the user data is transferred opaquely, i.e., the network applies only routing decisions and checksum processing on the messages it carries between end-systems. Active Networks extend this role by allowing the network to perform customized computation on user data. For example, a user of an active network could send a customized compression program to a node within the network (e.g. a router) and request that the node executes that program for filtering purpose when processing its packets [15]. Another variation could consist of adaptive rate control in active router by dropping some type of packets to adapt to network conditions.
Such networks are “active” in two ways:

- Switches perform computation on the user data flowing through them
- Users can inject programs into the network, thereby tailoring the node capabilities to the user and application profiles.

The implementation of an active network can be done with a more or less important granularity and dynamics of services. From a conceptual point of view, active network architectures are formed by two principal components: Active Nodes and Active Packet (Cell). Active nodes architecture deals with how packets are processed and how local resources are managed. The functionality of the active node is divided among the Node Operating System (Node OS), the Execution Environment (EEs) and the Active Applications (AAs) [4]. The general organization of these components is shown in figure 2.

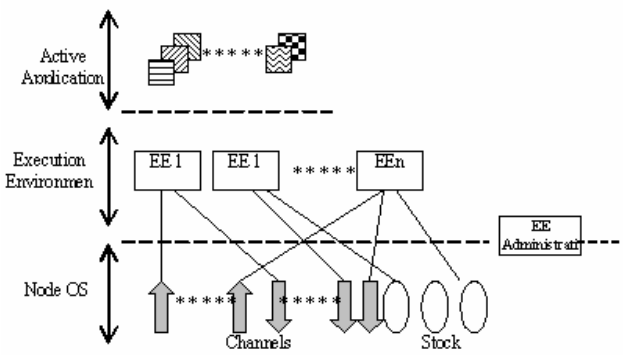

Fig. 2. Active node architecture

For the moment, there exist many implementations of Active Networks which explore different NodeOS, as ANTS [16] or BOWMAN [7]. We have chosen the SARA platform (Simple Active Router Assistant) [19] for dynamically and transparently deploying the FPTP service.

2.1 SARA

SARA implements the NodeOS and the EE over a dedicated processor, called Assistant, linked to an enhanced router. This approach permits to add active node functionalities to the shelf routers in a safe and high-performance way.
The SARA platform can operate over any router in the middle of the network without the explicit knowledge of the user. This means that users send their active packets to the final destination and the routers recognize and process them following the corresponding code, all this while taking normal routing decisions for non active packets.

2.1.1 SARA active node architecture

As explained before, the active applications are not processed by the router, but by a dedicated processor called Assistant, linked to the router. This approach reduces the overhead in the router to the simple identification and redirection of the active packets. SARA is an active node prototype developed using JAVA and is able to process, transparently, the active packets passing through the router.

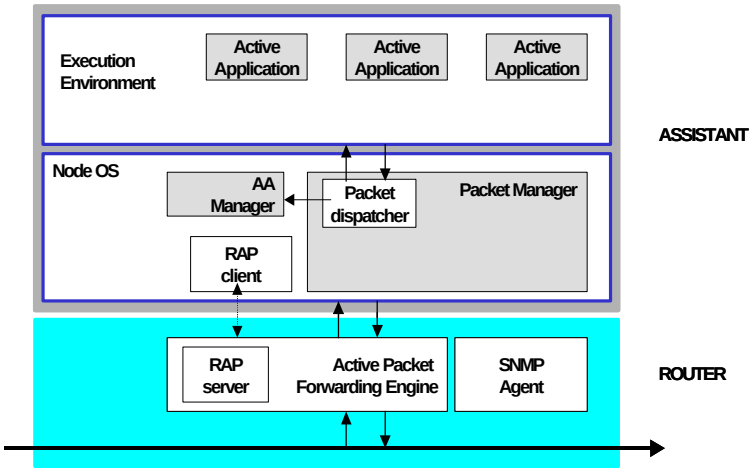

Fig. 3. SARA architecture

In agreement with the active general architecture described above, the SARA platform is formed by two principal modules, the Execution Environment (the execution support for the active applications), and the NodeOS. This NodeOS controls the entire system, administrate the applications and distributes the packets coming from the routing machine to the concerning applications (Figure 3).

2.1.2 SARA Transparency

Transparency of active node implementation is a desirable characteristic for allowing active services to be easily developed without disrupting distributed applications development practices.

The active packet concept in SARA is transparently implemented using the “router alert” bit within the IP header [5]. The active packets are simply UDP datagrams encapsulated into IP packets with the router alert bit activated. The SARA header is shown in figure 4.

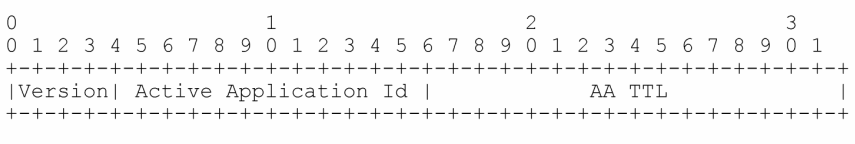

Fig. 4. SARA header packets

SARA provides packet encapsulation for requesting an active service by an UDP socket available in the SARA API.

3. A Proxy architecture to ensure QoS

We have explained before how the FPTP approach allows the quality of service for the multimedia applications to be assured more efficiently than with standard transport protocols. However, pragmatic considerations led us to take into account that most of current time critical multimedia applications are developed over the RTP/UDP/IP protocols. So, even if FPTP provides an API similar to the standard socket API, its integration within legacy applications would requires some expensive and dissuasive modifications and adaptations. This problem relates not only to server applications, but also client ones.

In order to facilitate the use of this new family of protocols, it is necessary to design a mechanism able to deliver FPTP services transparently on top of existing network infrastructures. This requirement led us to define an architecture based on a FPTP PEP (Protocol Enhanced Proxy). The FPTP PEP represents a flexible way to extend the actual Internet architecture with new services to solve problems and introduce services not expected in the initial network design [6]. The implementation of a transport level PEP introduces, potentially, certain number of problems related to security and end-to-end principles [3]. According to the end to end principle, for efficiency purpose, the network must have a limited intelligence restricted to packet routing, all complex services and treatments are pushed aside at the network periphery in end systems [11].

Nevertheless, an architecture based on a transport PEP does not try to replace the end-to-end functionality of the applicative layer. Indeed such an approach tries to add a performance optimization in the existing sub-networks between the end systems [3]. This is exactly the goal of our PEP architecture centered on FPTP.

The next subsections explain the architecture of the intermediate transport proxy that aims to improve the multimedia applications QoS; moreover the proposed mechanisms to deploy and maintain such architecture are also detailed.

3.1 Architecture and basics mechanisms

We propose an architecture made up of FPTP proxies placed on edge routers at the interface between different QoS domains. For traditional client server applications this approach distinguishes at least three different QoS domain (i.e. the two LANS that support respectively the client and the server applications and the global Internet) interconnected by two edge devices that support the FPTP proxies. In the server side, the proxy offers QoS oriented error, flow, rate and congestion control adaptive mechanisms. In the client side, the proxy makes error detection and synchronization control and enforcement. This additional level of QoS control enforced by the FPTP connection between the two FPTP proxies is based on the partial reliability, ordering and time constraints related to the multimedia flows or components transmitted by the FPTP connection (figure 5).

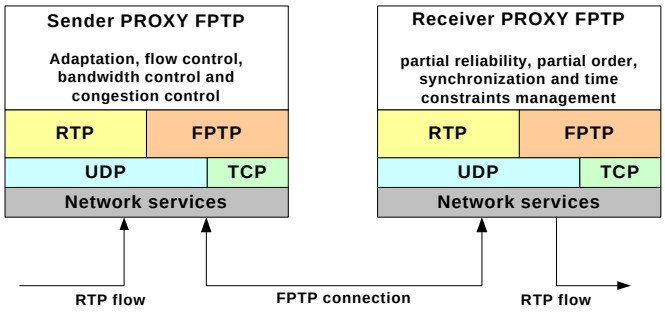

Fig. 5. FPTP proxy architecture

Moreover, transparent protocol translation from RTP to FPTP is done transparently taking account the QoS semantic into the RTP header packets [12]. We have experimentally observed that on flight filtering and analysis of RTP packets induce a negligible delay in active routers. Based on header information, the sending proxy is able to identify the multimedia data type for each packet (H.263 or MJPEG for example), the segmentation of application data units, the order and time constraints, the source address, etc. This set of information is used at the sending proxy level to instantiate a FPTP service compatible with the order, reliability and synchronization requirements of the transmitted application data units.

RTP packets are encapsulated into FPTP packets and the FPTP connection between the proxies allows granting the inferred QoS. The receiving proxy caches the packets and decodes the FPTP packets to reconstruct the multimedia flows to be sent to the receiving application.

As previously mentioned, such architecture supposes that the proxies are located in a strategic place near of the end-users. Therefore, there are not special QoS control mechanisms to be deployed between the end-system and the proxies (Figure 6).

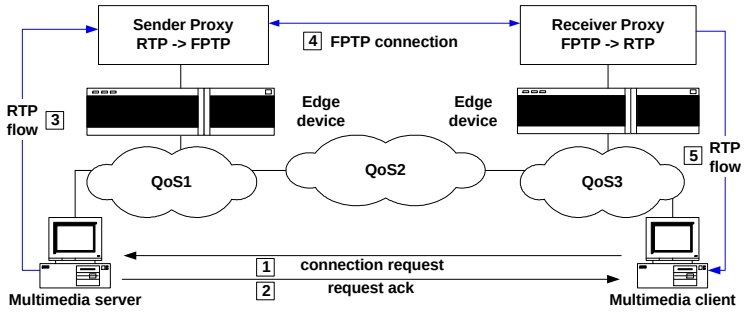

Fig. 6. Proxy architecture deployment

In Figure 6, a multimedia client asks for a video flow from a server placed in the other side of the network. The server answers and starts sending the RTP multimedia data. The sending proxy is able to recognize the flow QoS and to adapt the flow rate to the network behavior so offering the quality of service QoS2. For example, if the video flow is produced at 30 images per second and QoS2 is only 5 images per second capable, the sending proxy adapts the flow to the link possibilities while applying QoS oriented filtering techniques based on ADU semantics. The adapted flow will be transmitted to the receiver through a FPTP tunnel granting a dynamically inferred QoS.

3.2. Proxy transparency

In order to deploy the PEP architecture described previously, a first option consists in implementing explicitly and statically a connection between each end-system and its respective proxy as well as between both proxies. More exactly, this approach consists on replacing the destination address of the data flow on the server application for the sending proxy address and when the data arrives to the receiving proxy, replace it again by the client proxy and so on.
The shortcoming of this approach is that it breaks the end-to-end view of the server. An appreciable characteristic in a proxy is its degree of transparence [3]. A proxy can operate in a transparent way from the end-user point of view, assuring that no modifications should be done over the existing applications and protocols. In order of taking into account the advantages of the proxy's transparency, we have held this option for the FPTP PEP implementation.
In this approach, the sending proxy has to be deployed in one of the intermediate routing nodes present in the link between the multimedia service provider and the end user. In this node, the multimedia flows are automatically filtered and redirected to the receiving proxy via a FPTP connection (dynamically instantiated from the "on the fly" analysis done to the flow). This connection represents a tunnel carrying out a function of QoS guarantee and adaptation between the network services and the applicative requirements. Finally, the receiving proxy redirects the multimedia flow to the end-user.

In this way, no modifications are done to the multimedia applications and the QoS is guaranteed according to the FPTP approach in a transparent way from the point of view of the end user.

3.3 Deployment

The approach proposed previously raises the problem of the deployment of the FPTP protocol on active nodes. First of all, it is necessary to define the sending proxy location and configuration. The sending proxy may be statically placed and configured by the service provider. Actually, the implementation of this infrastructure can be considered as an add-on service offered by the multimedia provider aiming at improving the QoS delivered to its clients.
On the other hand, the main inconvenient lies in the receiving proxy deployment. In fact, the potential users of the services offered by the FPTP PEP are not known *a priori*. Thus, the router in the receiver side is not necessarily configured to support FPTP services and QoS control mechanisms. In order to solve this problem we propose an approach based on the dynamic deployment of transport services on receiving active nodes.
Indeed, the implementation of a receiving proxy can be done by sending an active packet over the client-server link. In this way, every active node able to intercept and recognize the packet and being located on a favorable place for the service required, can download, configure and launch the proxy services.
The active packet aiming to configure the receiving proxy can be sent by the client application when opening the connection multimedia. Nevertheless, this method obliges the modification of the receiving application. The same inconvenient results from sending active packet with the connection acknowledgement issued from the server. In contrast, transparency of the dynamic proxy deployment and configuration can be assured by the sending proxy that is in a privileged situation for sending active packets. Furthermore, from the interception of the first multimedia flow packet sent by the server, the sending proxy knows the network address destination where the active packet should be sent to.

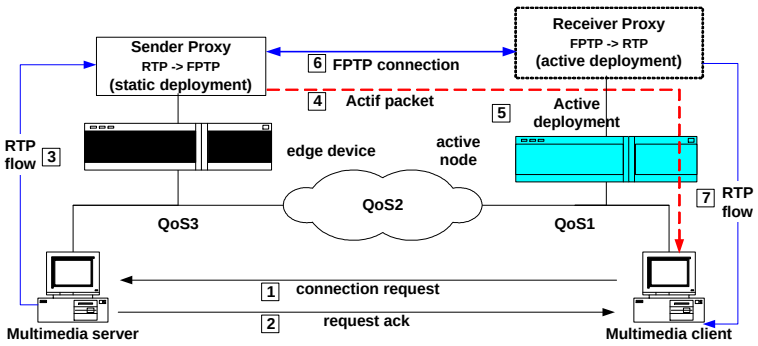

Fig. 7. Active deployment architecture

Figure 7 summarizes this methodology of deployment. First of all, from an application level protocol such a RTSP, a demand of multimedia component is

received and acknowledged by the multimedia server. As a consequence of this acknowledgement, the multimedia flow is sent to the client and intercepted by the proxy server which identifies the destination sub-network. Then, the sending proxy produces and sends an active packet to the destination sub-network. If an active node is found in the destination sub-network, the packet is intercepted, the receiving proxy is deployed and a FPTP connection is opened between the client and server proxies. It is advisable to note that the limit of this approach lies in the need of having an active router in the edge of the client sub-network. However, if no active node is found in the path from the proxy server to the multimedia client, the sender proxy can just perform a function of QoS control and adaptation on the multimedia flows delivered to the client.

4. Deployment and results

Within the framework of the GCAP European project [18], we have performed some experiments at a European Internet scale to evaluate the profits obtained from the proxy architecture described previously. At the application level, a video server and a client application accessing live or stored video flows were developed in JAVA by using the JMF support (Java Media Framework). Java and JMF permit the diffusion of multimedia flows over the RTP protocol, in different standard formats such as H.263 or MJPEG [17][2]. We have chosen JAVA as our development language due to the multi-platform compatibility requirements joined to several concluding performance tests done in preceding studies [1]. Therefore, the receiving and sending proxies, and the FPTP protocols, have been developed in JAVA.
Initially, we have done several tests locally using an emulation environment based on Dummynet [8]. During this emulation based phase, the proxies have been deployed statically over two industrial edge routers [20]. Between these two machines we have placed a third FreeBSD system that routes packets among the two others and support the emulation environment. This test-bed, shown in figure 8, allows, in a simple and powerful way, the fundamental network layer quality of service parameters such as bandwidth, end-to-end delay and distribution of losses to be dynamically modified.

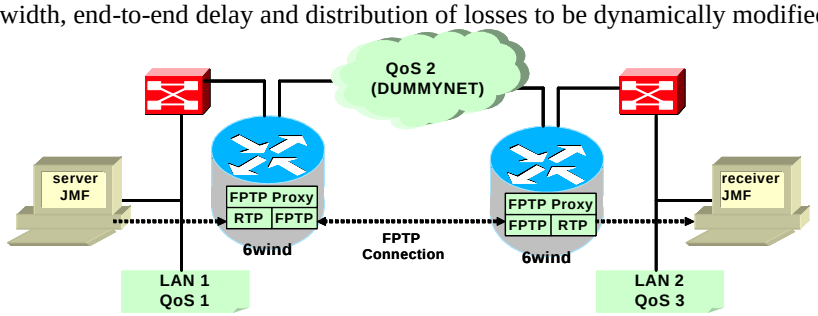

Fig. 8. Local platform

Experiments consisted in transmitting 5 minutes duration MJPEG flows at a approximate rate of 1 frame per second (250 frames per 300 seconds). In a first phase we have emulated a UDP service using a FPTP connection with 0% reliability and no

order constraint on the sequence of application data units. This first test has been done by emulating successively a fully reliable network service and an unreliable one that entails between 10 and 30% of losses. The results have been compared with those obtained from a FPTP connection offering a 70% reliable service. The results, summarized in the figure 9, show that FPTP compared to UDP provides systematically, a more important data unit percentage of data delivered on time.

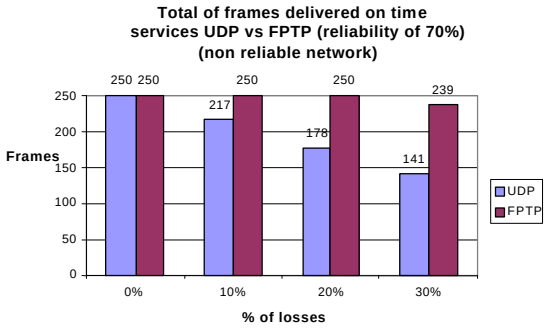

Fig. 9. Comparison of reliable service on UDP and FPTP

Traditionally, time critical multimedia applications such as voice over IP or Video on demand, privilege the use of the UDP protocol because constraints on flow continuity are not compatible with the potentially unbounded discontinuities created by a reliable transport services such as the one delivered by TCP. Figure 10 shows that the partially reliable ordered and timed constraints service offered by FPTP allows an adaptive behavior of the transport mechanisms that result in the increment of the number of frames received by the user in congested network environment.

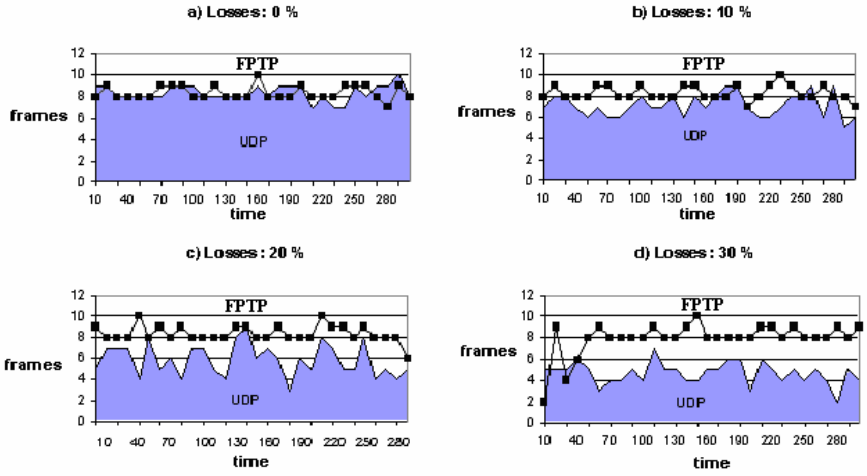

Fig. 10. Number of frames delivered on time (FPTP vs UDP)

In these figures we see that when network conditions are good (case a), FPTP and UDP deliver a similar service which respect the flow continuity. In case of network

congestion (cases b, c, and d), FPTP delivers on time a bigger number of frames per unit of time.

In a second time, we have compared the services delivered by the proxy using a FPTP connection ensuring a minimum of reliability of 70% and a TCP connection. This comparison allows to demonstrate that, in any network conditions, the service provided by FPTP offers an optimal trade-off (i.e., in conformity with the application requirements) between the unreliable-unordered service delivered by UDP and the fully reliability-ordered service offered by TCP. In order to reduce the negative impact of the TCP congestion control mechanisms on the experiments, we have emulated TCP by instantiating a fully reliable and ordered FPTP service. Figure 11 shows that in a network with 10% of losses, the partial reliability provided by FPTP allows delivering all the multimedia data in an interval of time that complies with the time constraints of the flow because the difference between the arrival time and the presentation time is positive. On the other hand, the use of TCP in the same network conditions generates a disturbing discontinuous service due to systematic and unbounded packet retransmissions of lost packets. It is important to note that a real TCP implementation in these emulated congestion conditions would have produce a worst service to the user because of the drastic rate reduction entailed by congestion control mechanisms.

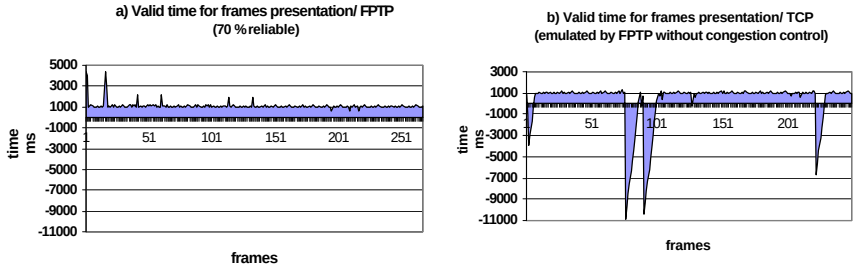

Fig. 11. Arrival and presentation time difference per ADU (FPTP vs TCP)

A second test-bed has been done over the European research Internet between Toulouse (LAAS/CNRS) and Madrid (Carlos III University). In these experiments the multimedia proxy server and application have been placed at Toulouse. The SARA active platform associated to the Telebit router and the receiver application have been placed at Madrid (see figure 12).

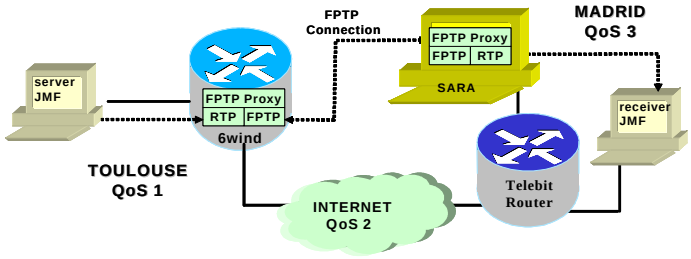

Fig. 12. Active deployment over SARA platform (Toulouse-Madrid)

The receiving proxy has been adapted to be actively deployed over the SARA architecture. The code of the FPTP protocol has been placed on a FTP server working as a repository server. So, after the access demand to the video flow from the client, the sending proxy detects the RTP flows coming from the multimedia server and produces an active packet to be sent to the client. This packet is filtered by the active node placed between the sending proxy and the client (i.e., the Telebit router) who redirects it to the SARA Assistant. The packet is decoded by the assistant who checks whether the corresponding active application is available or whether it is necessary to load it remotely. After the load and execution of the application, the receiving proxy is deployed and the FPTP connection is established with the proxy transmitter. The multimedia flow is so transmitted through the FPTP tunnel and finally directed to the receiving application. After many tests, the results show that the required average time since the active packet is transmitted by the sending proxy till the connection is established with one active node is about of 295 milliseconds, if the code is locally available (i.e. in the active node), otherwise there is an additional time corresponding to the transmission the 42KB of compressed p-code associated to the FPTP protocol (this time is negligible when the code repository is located in the same LAN than the active node).

5. Conclusions and perspectives.

In this paper we have demonstrated that FPTP is able to improve the QoS delivered to the multimedia applications by doing an efficient adaptation between application layer QoS needs and the services offered by the network layer, this is in particular true for Best Effort networks such as the current Internet.

We have also shown how a new generation transport service can be transparently and efficiently deployed in an active network infrastructure. This active deployment represents an efficient and inexpensive approach to implement advanced QoS oriented network services.

The experiments developed in the framework of the GCAP European project have allowed the contributions of this new generation of protocols to be successfully experimentally validated.

Studies should be done to analyse the contribution of this approach in a large scale multi domain network context with different QoS. Additionally, some studies are being done to propose congestion control mechanisms not only fitting to the network behaviour but also in accordance with the QoS required by the applications. The integration of temporal constraints in the error control, flow control, congestion control and even in the order control mechanisms implemented in the routers or active nodes represents another interesting research feature. Some experiments over the new generation of the differentiated network services should also be done.

6. Bibliographie.

1. Apvrille L., Dairaine L., Rojas-Cardenas L., Sénac P., Diaz M., “Implementing a User Level Multimedia Transport Protocol in Java”, The Fifth IEEE Symposium on Computers and Communication (ISCC’2000), Antibes-Juan les Pins, France, July 2000.
2. Berc L., Fenner W., Frederick R., McCanne S., Stewart P., “ RTP Payload Format for JPEG-compressed Video”, RFC 2435, October 1998.
3. Border J., Kojo M., Griner J., Montenegro G., Shelby Z., “Performance Enhancing Proxys Intended to Mitigate Link-Related Degradations”, RFC 3135, June 2001.
4. Calvert K.L., “Architectural Framework for Active Networks”.. University of Kentucky; 1999.
5. Katz D., “IP Router Alert Option”, RFC 2113, Network Working Group, February 1997
6. Knutsson B., Architectures for Application Transparent Proxys: A Study of Network Enhancing Software. DoCS 01/118, 119 pp. Uppsala. ISSN 0283-0574.
7. Merugu S., Bhattacharjee S., Zegura E., Calvert K., “Bowman: A Node OS for Active Networks”. DARPA data base.
8. Rizzo L., “Dummynet: a simple approach to the evaluation of network protocols”, ACM Computer Communication Review, Vol. 27, no. 1, January 1997.
9. Rojas L., Chaput E., Dairaine L., Sénac P., Diaz M., “Transport of Video on Partial Order Connections”, Journal of Computer Networks and ISDN Systems, 1998.
10. Rojas L., Sénac P., Dairaine L., Diaz M., Towards a new generation of transport services adapted to multimedia applications, published in Annals of Telecommunication, December 1999
11. Salter J.H., Reed D.P., Clark D.D., “End-to-end arguments in system design”, In ACM Transactions on Computer Systems. ACM, 1984.
12. Schulzrinne H., Casner S., Frederick R., Jacobson V., “RTP: A Transport Protocol for Real-Time Applications”, RFC 1889, January 1996.
13. Sénac P., Exposito E., Diaz M., “Towards a New Generation of Generic Transport Protocols”, Lecture Notes in Computer Science 2170, Springer, Eds. S.Palazzo, September 2001.
14. Stewart R., Xie Q., Morneault K., Sharp C., Scwarzbauer H., Taylor T., Rytina I., Kalla M., Zhang L., Paxson V., “Stream Control Transmission Protocol”, RFC 2960, October 2000.
15. Tennenhouse D.L., Wetherall D., “Towards an Active Network Architecture”.. Multimedia Computing and Networking, 1996.
16. Watherall D., Guttag J.V., Tennenhouse D.L., “ANTS: a toolkit for building and dynamically deploying network protocols”.. OPENARCH’98.
17. Zhu C., “RTP Payload Format for H.263 Video Streams”, RFC 2190, September 1997.
18. GCAP : Global Communication Architecture and Protocols, IST-1999-10 504, home site: http://www.laas.fr/GCAP/
19. SARA home site. http://matrix.it.uc3m.es/~sara.
20. 6WIND home site. http://www.6wind.com