

RESERVATION MODELS: FROM AREQUIPA TO SRP

WERNER ALMESBERGER, SILVIA GIORDANO and JEAN-YVES LE BOUDEC
Institute for Computer Communication and Applications (ICA)
Ecole Polytechnique Fédérale de Lausanne (EPFL)
<http://icawww.epfl.ch>

September 14, 1998

Abstract

We analyse and illustrate two approaches that we have developed for providing reservations to flows. The first approach uses the explicit reservation model: a traffic profile is negotiated between users and the network. ATM and RSVP are two examples for this approach. In this paper we describe our experience in the ACT project EXPERT (AC094) with the explicit reservation model in the context of ATM. We show that the integration of ATM in a TCP/IP oriented operating system is easy and does not require developing applications specially for ATM. The method we have designed is called Arequipa and is described in RFC 2170. It is fully implemented in Linux.

However, we are confronted with the complexity required to support a single reservation for each flow. Our analysis is that it is well suited for segregating flows in applications such as private virtual networks, but that, for multimedia networks, the cost of handling a large number of flows is too high. This and other factors led us to develop an alternative solution, based on an implicit reservation model. Our solution is called the Scalable Reservation Protocol (SRP). SRP aggregates flows inside the network: routers other than edge routers performing policing do maintain only aggregate information per port. SRP is being developed and implemented in the framework of the ACTS project DIANA (AC319).

Keywords QoS; Aggregation; Arequipa; ATM; IP; SRP

1 Introduction

The integration of voice, data, and video services provides a new objective for networking technologies. Instead of simply providing best effort service, the network now also has to deal with the integration of services and, as a consequence, must provide some Quality of Service (QoS). There is a large interest for network solutions able to support QoS, and the user is often confronted with the difficulty of understanding which solution will be able to suit their needs. Several papers give technical overviews on the competing integrated services network solutions [1], and this is also the content of the work of the NIG G3 IP ATM Integration Chain Group of the European Community [2]. A conclusion that arises from [2] that nowadays only very few protocols offer the possibility

of specifying or managing QoS. Moreover those protocols are faced with severe scalability issues.

In this paper we report on our experience in developing solutions for providing reservations to flows. Our work has led us to evolve from a first approach, the explicit reservation model, to a second one, the implicit reservation reservation model.

One network technology which uses the explicit reservation model is the Asynchronous Transfer Mode (ATM) [3, 4], which promises to provide a scalable network architecture. The design of ATM considered reservation mechanisms from the very beginning. ATM networks therefore now offer reliable reservation mechanisms and well-understood traffic management concepts. Corporate and public ATM networks are already a reality in many places. Applications that are specifically written to use ATM, so-called *native* ATM applications, can use benefit of end-to-end QoS guarantees already today. However, one major problem is that the vast majority of networked applications is written to TCP/IP service interfaces, not to ATM. If you want to use TCP/IP applications in such environments, the standard solution is to run IP over ATM; however, with IP over ATM today, there is no simple way yet to benefit from the end-to-end QoS guarantees of ATM.

Another network technology which uses the explicit reservation model is based on enhancements to the Internet. The current Internet does not provide reservations, but the IETF has identified the need for supporting integrated services already years ago ([5], [6]) and has been working on the design of reservation mechanisms for TCP/IP. One major result of this activity are the Resource reSerVation Protocol (RSVP [7]) and the corresponding mappings to specific link layers, which are currently still in draft status. This approach is based on the concept of *integration*: network nodes (here: routers) need to be upgraded in order to support an additional set of functions required by the reserved services. Once and if RSVP is deployed across the Internet, it is possible to use TCP/IP applications with some end-to-end quality of service.

In Section 2, we report on the feasibility of an approach which solves the problems of integrating ATM with the TCP/IP host environment. It is called Application REquested IP over ATM (Arequipa) [8, 9]. We claim that, with Arequipa, we showed that providing the quality of service of ATM to TCP/IP applications is straightforward, with minimum changes to the TCP/IP implementation in hosts. For two end-systems to communicate using Arequipa, it is necessary that they are connected (1) to a common ATM network and (2) to the Internet or to the same Intranet. However, there is no cooperation required between the two types of networks. We say that our approach is based on a concept of *segregation*, in contrast to RSVP which uses integration. Arequipa allows applications to establish direct point-to-point end-to-end ATM connections with a given QoS at the link level. These connections are used exclusively by the applications that requested them. After setup of the Arequipa connection (namely, the ATM connection that is used for Arequipa), the applications can use the standard TCP/IP service to exchange data.

We report on Arequipa demonstrations conducted in the EXPERT project [10, 11]. In these demonstrations, the user was able to modify the ATM peak cell rate (PCR) offered to the video-conference application (*Vic*), at run time, using ATM renegotiation capabilities.

However, we are confronted with the complexity required to support a single

reservation for each flow of the explicit reservation model. Our analysis is that it is well suited for segregating flows in applications such as private virtual networks, but that, for multimedia networks, the cost of handling a large number of flows is too high. This and other factors led us to develop an alternative approach, called the Scalable Reservation Protocol (SRP). SRP aggregates flows inside the network: routers other than edge routers performing policing do maintain only aggregate information per port. SRP is being developed and implemented in the framework of the ACTS project DIANA [12].

SRP uses the implicit reservation model. With SRP, we distinguish traditional best-effort traffic from traffic that requires a better service. The global network is therefore seen as divided in two (independent) virtual networks: the current Internet and a "contracted network". The contracted network is monitored and the traffic that exceeds its profile is dropped or not assured. Note that this is different from priority schemes where, under congestion, lowest priority traffic is discarded even if it respects its contract; here traffic is discarded only if it does not respect the contract. Implicit reservations provide a means for high scalability. Section 3 describes the essential elements of SRP.

2 The Explicit Reservation Model: Arequipa

2.1 The principles of Arequipa

The ATM Forum and the ITU [4, 13] standards for UNI signalling offer a rich set of features to support guaranteed quality of service (QoS). In order to use ATM in a TCP/IP environment, we need to map IP over ATM. Here we are confronted with a limitation of the classical model used by the Internet. According to the Internet Protocol (IP), a host *A* should, in principle, not use direct end-to-end connections to some other host *B*, unless both *A* and *B* have been configured to belong to the same IP subnet. This restriction is underlying the "classical IP over ATM" solution [14]; with this solution, hosts *must* use routers in between if they are not in the same subnet, and *must* establish direct end-to-end ATM connections if they are the same subnet. In order to overcome these limitations, RFC 1937 suggests modifying the Internet Protocol implementation in hosts in order to let the host decide what is more appropriate. With *Arequipa* [8, 9], we have designed and implemented a solution which enables the IP host to use host-to-host ATM connections across subnet boundaries, assuming ATM connectivity exists between the hosts. This is illustrated in Figure 1.

The approach taken by Arequipa is possible because TCP/IP implementations do not follow a strict layer separation: the IP destination tables in hosts are typically set per socket pair, rather than per IP destination address. This makes it possible to select a given ATM connection for one specific application flow, instead of for one IP destination address. Service integration in hosts rather than in the network makes it possible to use QoS immediately, since ATM commercial networks are already in operation.

Arequipa allows ATM-attached hosts that have direct ATM connectivity to use the ATM explicit reservation model to obtain QoS. These hosts set up end-to-end IP over ATM connections, within the reachable ATM cloud, on request from applications and for the exclusive use by the requesting application. The QoS is guaranteed by the fact that each of these connections is used exclusively

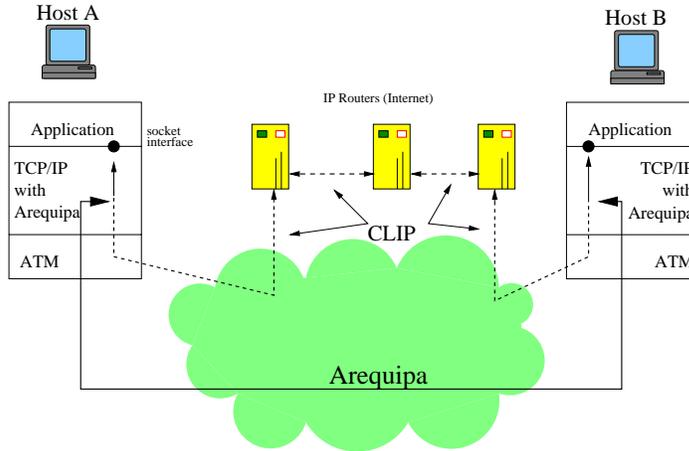


Figure 1: CLIP solution (dashed lines): host *A* and *B* cannot use the ATM end-to-end connection. Arequipa solution (solid line): IP hosts can use the end-to-end ATM connection.

for one IP flow identified by a pair of sockets (eg. a TCP connection or a UDP stream).

The QoS is requested for each connection: the application specifies through the Arequipa API the needed QoS, which will be negotiated into the network via ATM signaling [4]. This API consists of calls for setting up and tearing down connections and for modifying the traffic parameters of connections. By this means the applications can negotiate and renegotiate the traffic parameters of a direct switched virtual channel connection (SVC).

Arequipa does not require any modifications in the networks (routers, switches etc.) but some changes need to be made to the TCP/IP stacks at the end systems (discussed in detail in [15]).

2.2 Arequipa demonstration

Arequipa has been implemented in the Linux operating system and is part of the ATM on Linux distribution [16], version 0.32, of ICA. For the establishment and the release of ATM connections the signaling parts of classical IP over ATM have been reused. A new virtual network device for Arequipa has been created and a few modifications have been made to the socket layer. The implementation is detailed in [15].

Arequipa is a relatively simple way for IP applications to use the QoS of ATM. In the framework of the ACTS project EXPERT we demonstrated this in the case of video conferencing (Vic - the Mbone tool).

The demonstration consisted of video conferencing traffic transported on ATM by means of Arequipa protocol, as illustrated in Figure 2.3. For this demo Vic was enabled to use Arequipa. End systems (PC running Linux over ATM [16]) and one of the switch (ATMLight Ring by ASCOM) were upgraded with (partial) UNI4.0 signaling [4] and the Q.2963.1 connection modification capability [13], and even Arequipa was extended to include the connection modification capability. Connection modification capability relates to modifying the peak cell

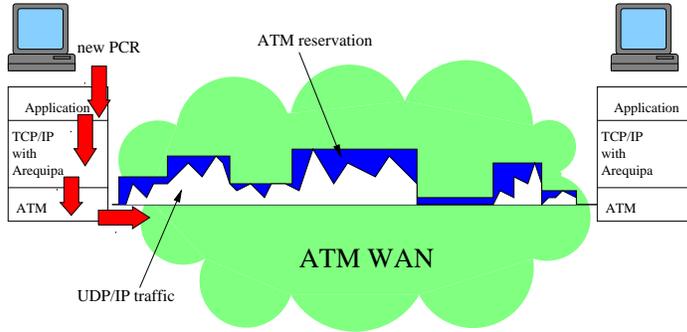


Figure 2: The demonstrations network scenario: the reservation profile (PCR) is modified by the users in order to adequate the ATM traffic reservation to the UDP/IP traffic generated by Vic application

rate (PCR) of an already active CBR connection. With these an application can now modify the QoS parameters at run time, after connection setup. To our knowledge this was the first time any application had the ability to tune ATM traffic parameters at run time. The application provides to the user the ability to tune only one traffic parameter (the PCR). We note that the choice was forced by the connection modification capability definition given by Q.2963.1 standard.

2.3 Discussion

However, even if the Arequipa solution was appropriate and satisfactory in this scenario, it was evident that the complexity to support each connection separately is too high to make attractive its usage for multimedia networks.

With Arequipa made the conscious choice to let the user, or the application, explicitly control the ATM connection. In our implementation, we support unspecified bit rate (UBR) and constant bit rate (CBR) connections. In the latter case, the user or application has to specify the requested peak cell rate, and can modify on demand at any time after connection setup. The additional traffic or QoS parameters required by the ATM signalling procedure are set transparently by our implementation. We believe that it is reasonable to limit the information requested from the user or application to just the choice mentioned above, namely : UBR with no rate information, or CBR with a specified peak cell rate. Our choice to make the traffic specification visible to the application is an essential part of Arequipa; we believe that it will become more common in the future for a large variety of applications. It is based on the concept that QoS comes with a price, and therefore we expect a dialogue between application and user to take place before a guaranteed QoS is requested.

However, this does come with a drawback : existing application code has to be modified. As mentioned above, modified code exists for the web and the Mbone. An alternative to our approach is to let the operating system choose the traffic parameters in lieu of the application. We do not follow this approach with Arequipa because we explicitly want to make quality of service visible to the end-user.

There is a number of lessons we learned from the implementation and deployment of Arequipa, but in this paper we would like to focus on one major lesson. It has to do with the observation that, contrary to our expectation when we started the project in 1994, the penetration of end-to-end ATM remains minuscule. One obvious reason is the fact that ATM requires specific communication adapters, and cannot run today on the existing hosts, which, for the vast majority of them, use Ethernet.

However, our work on Arequipa may give us some additional clues about the reasons for this state of affairs. Certainly, the lack of ATM penetration is *not* due to the difficulty of making QoS available and visible to the user. Indeed, with Arequipa, we have a solution readily available for the Unix environment, and we conjecture that porting that solution to the market dominating operating system would not be a major effort. We also do not believe that the minor changes required to web clients or servers are a major drawback, since the lifetime of this type of software is usually shorter than a year. If there would be a massive push to obtain QoS in hosts, then the ATM penetration would be higher. Therefore, we are lead to think that making QoS visible to the user is an idea that simply did not meet its market. Many users would like to have it, but hardly any organisation is willing to invest into the network technology required to support it. This also leads us to conjecture that approaches based on RSVP that would attempt at making QoS visible to the end user will equally suffer from the same lack of penetration, because introducing RSVP into the Internet is also a major investment.

If we follow this line of thoughts, we conclude that it may not be a good idea to let the QoS be visible to the application, except maybe for niche application settings where the investment is justified. Such settings are, for example, remote lecture rooms used in medical teaching applications, or video on demand over ATM. In contrast, in a TCP/IP setting, it may be that what users need is “a better service”, instead of, for example, “a guaranteed 250 kb/s flow”. This led us to an implicit reservation model.

3 The Implicit Reservation Model: SRP

The implicit reservation model tries to obviate the difficulty of specifying the traffic parameters by aggregating flows into the network. SRP ¹ provides a light-weight reservation mechanism for adaptive multimedia applications [18]. Our main focus is twofold:

1. support good scalability to very large numbers of individual flows
2. avoid explicit reservations

With SRP, end systems (i.e. senders and destinations) actively participate in maintaining reservations, but routers can still control their conformance. Routers aggregate flows and monitor the aggregate to estimate the resources needed to support present and new reservations. There is no explicit signaling of flow parameters.

¹A detailed description of SRP can be found in [17]. The Web page of SRP is at <http://lrcwww.epfl.ch/srp/>

Although similar in its goals, SRP goes beyond the capabilities expected from the Differentiated Services architecture [19] in that it already provides a consistent end-to-end mechanism for establishing reservations.

3.1 End-to-end service

Many adaptive multimedia applications require a well-defined fraction of their traffic to reach the destination and to do so in a timely way. We call this fraction the *minimum rate* these applications need in order to operate properly. SRP aims to allow such applications to make a dependable reservation of their minimum rate.

One reflection that led us to design SRP is the following. When performing a reservation, the guarantee that a multimedia application requires is simply that a certain amount of traffic can be accepted by the network from now on, for as long as the application requires. We call this the “sticky network service”. In contrast, a real reservation would consist in booking in advance some resources, with the guarantee that they would be available at the agreed time in the future.

SRP is a simple solution for providing the sticky network service. The sender can expect that, as long as it obeys the rules SRP, no *reserved* packets will be lost due to congestion. Furthermore, forwarding of *reserved* packets will have priority over best-effort traffic. The service provided by SRP can be thought of as being similar to the INTSERV Controlled-Load service [20].

3.2 Reservation mechanism

A source that wishes to make a reservation starts by sending data packets marked as *request* packets to the destination. (Figure 3.)

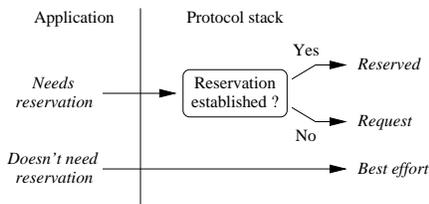


Figure 3: Initial packet type assignment by sender.

Packets marked as *request* are subject to packet admission control by routers, based on the following principle. Routers monitor the aggregate flows of *reserved* packets and maintain a running estimate of what level of resources is required to serve them with a good quality of service. The resources are bandwidth and buffer on outgoing links, plus any internal resources as required by the router architecture. Quality of service is loss ratio and delay, and is defined statically.

A router (figure 4), when receiving a *request* packet, determines whether hypothetically adding this packet to the flow of *reserved* packets would yield an acceptable value of the estimator. If so, the *request* packet is accepted and forwarded towards the destination, while still keeping the status of a *request* packet; the router must also update the estimator as if the packet had been received as *reserved*. In the opposite case, the *request* packet is degraded and

forwarded towards the destination, and the estimator is not updated. Degrading a *request* packet means assigning it a lower traffic class, such as best effort. A packet sent as *request* will reach the destination as *request* only if all routers along the path have accepted the packet as *request*. Note that the choice of an estimation method is local to a router and actual estimators may differ in their principle of operation.

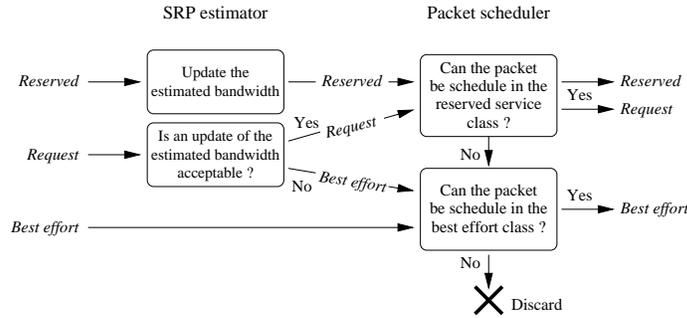


Figure 4: Packet processing by routers.

The destination periodically sends feedback to the source indicating the rate at which *request* and *reserved* packets have been received. This feedback does not receive any special treatment in the network (except possibly for policing, see section 3.5). Upon reception of the feedback, the source can send packets marked as *reserved* according to a profile derived from the rate indicated in the feedback. If necessary, the source may continue to send more *request* packets in an attempt to increase the rate that will be indicated in subsequent feedbacks.

Thus, in essence, a router accepting to forward a *request* packet as *request* allows the source to send more *reserved* packets in the future; it is thus a form of high scalability reservation.

3.3 Aggregation

Routers aggregate flows on output ports, and possibly on any contention point as required by their internal architecture. They use estimator algorithms for each aggregated flow to determine their current reservation levels and to predict the impact of accepting *request* packets. The exact definition of what constitutes an aggregated flow is local to a router.

Likewise, senders and sources treat all flows between each pair of them as a single aggregate and use estimator algorithms for characterising them. The estimator algorithms in routers and hosts do not need to be the same. In fact, we expect hosts to implement a fairly simple algorithm, while estimator algorithms in routers may evolve independently over time. Issues as estimator, multicast and evaluation of example host and router algorithms can be found in [17].

3.4 Source and destination behaviour

A source needs to limit the *reserved* traffic it emits to the rate corresponding to the reservation in the network. It does this by taking the minimum of the rate

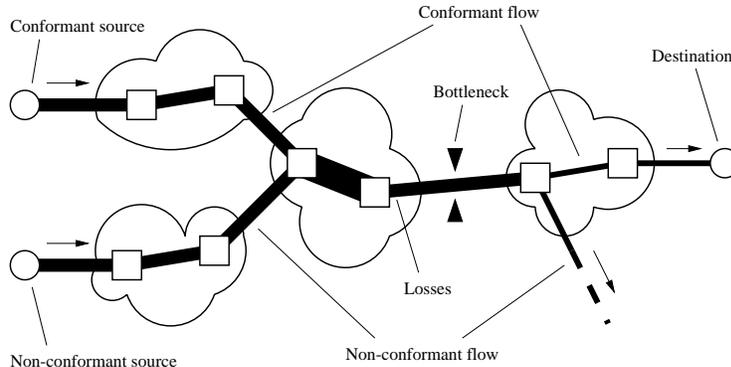


Figure 5: Policing: the bottleneck router can block non-conforming flows

computed using the feedback received from the source and a local estimate of the requested reservation. Small rate fluctuations due to jitter are eliminated from the feedback in order to prevent reservations from drifting.

Destinations send feedback frequently when the reservation changes (e.g. when the network has accepted an increase, but also if the rate drops due to network failures), and they slow down if there are no changes. If a destination remains silent for too long, the source must assume that it is no longer reachable and stop sending.

As with any packet network protocol, it is necessary to implement congestion control mechanisms. Congestion control for *reserved* packets is implemented by the network nodes when they take packet admission decisions. Therefore, sources may send *reserved* packets at the rate computed by the feedback, without any further restriction (this is the very definition of the sticky network service).

In contrast, a source might wish to send a large number of *request* packets in order to establish a reservation. However, depending on the implementation, *request* packets compete for bit rate and buffer with best effort packets. Thus the emission of *request* packets should obey some form of congestion control, namely, sending *request* packets should be “TCP-friendly”.

3.5 Policing

Reserved traffic may exceed the actual reservation if end systems or routers violate the semantics of SRP. This can happen due to defects or also because a system is set up to intentionally generate non-conforming traffic. If a flow exceeds its reservation, it may inflict delay, congestion, and eventually losses on conforming flows. Therefore, non-conforming traffic needs to be detected and treated such that it does not cause damage.

Unfortunately, aggregation complicates policing. Figure 5 shows a typical scenario where a conforming flow is harmed by a non-conforming flow. Assuming that upstream routers have granted the non-conforming flow a reservation exceeding what it has obtained at the bottleneck link, only the router before the bottleneck is able to detect the non-conformance. Since the non-conforming flow has already been merged with conforming flows before, arbitrarily dropping packets from the aggregate also affects the conforming flows. Note that upstream routers cannot simply examine feedback packets in order to determine

the actual end-to-end reservation, because routes may be asymmetric.

We are currently studying a scalable approach to policing, where flows that differ sufficiently from the established reservation for the aggregate are detected by statistical means. This way, the bottleneck router can block non-conforming flows even if upstream routers are unable to detect the non-conformance themselves. In addition to that, a router detecting non-conformance may also notify its upstream neighbour in order to squelch the non-conforming flow as early as possible and to limit the number of blocked flows a router has to keep track of.

4 Conclusion

We have presented two methods for providing reservations.

The former is Arequipa, a method for providing the quality of service of end-to-end ATM connections to TCP/IP applications. It makes it possible to use ATM natively, in those cases where end-to-end ATM connectivity exists, while preserving the TCP/IP environment, and with only minimal changes to the application code. We have implemented Arequipa in Linux, tested it extensively, made it a part of the ATM-Linux distribution, and published an Internet RFC documenting it. Finally, we have presented the results of a test of Arequipa in a European WAN setup. With Arequipa, we have made the proof that supporting explicit quality of service in end systems is indeed simple. An alternative solution to Arequipa, which would use a different network technology, would be based on RSVP.

With Arequipa, we were led to think that making QoS visible to the user is an idea that simply did not meet its market. Many users would like to have it, but hardly any organisation is willing to invest into the network technology required to support it. This also led us to conjecture that approaches based on RSVP that would attempt at making QoS visible to the end user will equally suffer from the same lack of penetration.

We thus designed a new method for providing reservations called SRP. It starts from the idea that a reservation is in reality nothing more than a form of the “sticky network service”. With SRP, sources send data as candidate packets (data packets marked as *request* packets), and then learn from destinations how much was really accepted by the various network nodes. When a network node accepts a flow of request packets, it commits to have enough resource in the future in order to support the same flow again, but now under the form of *reserved* packets. Thus a source learns from the packet flow how much can be guaranteed. There is no implicit reservation for the source, and aggregation of flow comes for free in the network.

Arequipa is fully implemented (ACTS project Expert) and is publicly available in the Linux ATM distribution. We are currently implementing SRP on hosts and routers (ACTS project Diana).

References

- [1] S. Giordano, R. Schmid, R. Beeler, H. Flinck, J.-Y. Le Boudec, “IP, ATM - current evolution for integrated services ,” in *Interworkin98 Proceedings, Ottawa, 1998*.

- [2] NIG-G3 Chain-Group, *NIG-G3: Internet, ATM coexistence Guideline*, 1998. <http://gina.ihe.ac.be/nig-g3>.
- [3] J.-Y. Le Boudec, "The Asynchronous Transfer Mode : A Tutorial," *Computer Networks, ISDN Systems*, vol. 24 (4), pp. 279–309, May 1992.
- [4] The ATM Forum, *ATM User-Network Interface (UNI) Signalling Specification, Version 4.0*, 1996. <ftp://ftp.atmforum.com/pub/approved-specs/af-sig-0061.000.ps>.
- [5] D. Clark, S. Shenker, L. Zhang., "Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanisms," *Proceedings of ACM SIGCOMM '92*, 1992.
- [6] R. Braden, D. Clark, S. Shenker, *RFC1633: Integrated Services in the Internet Architecture: an Overview*. IETF, June 1994.
- [7] R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin, *RFC2205: Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification*. IETF, September 1997.
- [8] W. Almesberger, J-Y. Le Boudec, Ph. Oechslin, *RFC2170: Application REQuested IP over ATM (AREQUIPA)*. IETF, July 1997.
- [9] W. Almesberger, J-Y. Le Boudec, Ph. Oechslin, "Arequipa: TCP/IP over ATM with QoS ... for the impatient," Technical Report 97/225, DI-EPFL, CH-1015 Lausanne, Switzerland, January 1997. <ftp://lrcftp.epfl.ch/pub/arequipa/impatient.ps.gz>.
- [10] W. Almesberger, L. Chandran, S. Giordano, J.-Y. Le Boudec, R. Schmid, "Quality of Service Renegotiations," *to appear in: SPIE Int. Symp. on Voice, Video and Data Communications proceedings, Boston*, November 1998.
- [11] W. Almesberger, L. Chandran, S. Giordano, J.-Y. Le Boudec, R. Schmid, "Using Quality of Service can be simple: Arequipa with Renegotiable ATM connections," *to appear in: Computer Networks and ISDN Systems*, October 1998.
- [12] DIANA Consortium, *DIANA AC319 home page*, 1998. <http://www.telcom.ch/Diana>.
- [13] ITU Telecommunication Standardization Sector - Study group 13, *ITU Recommendation Q.2931, Broadband Integrated Services Digital Network (B-ISDN) - Digital subscriber signalling system no. 2 (DSS 2) - User-network interface (UNI) - Layer 3 specification for basic call/connection control*, 1995.
- [14] M. Laubach, *RFC1577: Classical IP, ARP over ATM*. IETF, 1994.
- [15] W. Almesberger, "Arequipa: Design, Implementation," Technical Report 96/213, DI-EPFL, CH-1015 Lausanne, Switzerland, November 1996.
- [16] W. Almesberger, *ATM on Linux*. EPFL, 1996. ftp://lrcftp.epfl.ch/pub/linux/atm/papers/atm_on_linux.ps.gz.

- [17] W. Almesberger, T. Ferrari, J.-Y. Le Boudec, "SRP: a Scalable Reservation Protocol for the Internet," Technical Report 98/009, DI-EPFL, CH-1015 Lausanne, Switzerland, March 1998. <ftp://lrcftp.epfl.ch/pub/people/almesber/srp/srpMar98.ps.gz>.
- [18] C. Diot, C. Huitema, T. Turlatti, "Multimedia Applications should be Adaptive," *HPCS'95 Workshop*, 1998. <ftp://www.inria.fr/rodeo/diot/nca-hpcs.ps.gz>.
- [19] IETF, *Differentiated Services (diffserv) working group*, 1998. <http://www.ietf.org/html.charters/diffserv-charter.html>.
- [20] J. Wroclawski, *RFC2211: Specification of Controlled-Load Network Element Service*. IETF, September 1997.