UNIVERSITY OF HELSINKI

DEPARTMENT OF COMPUTER SCIENCE

Context transfer for Quality of Service in radio access networks

Student

Leggio Simone

Supervisor:

Prof. Kimmo Raatikainen

CONTEXT TRANSFER FOR QUALITY OF SERVICE IN RADIO ACCESS NETWORKS

Abstract

The telecommunication world is quickly changing nowadays, and new services, new devices, new technologies are growing, all promising better and ambitious quality of service to customers. The direction where this development seems to be going is without doubt the wireless, and the challenge is to provide these services on radio links. This choice of the market has as its consequence that these new applications must be suited for a wireless environment, with all the related problems.

Particularly, one of the most discussed topics is to achieve the possibility to utilize real time applications on these links as well: to be able to do this, the strict requirements that this kind of application imposes, mainly on the timeliness of the delivered data, must be met. The main issue to this purpose is represented on a wireless link, by the handover that mobile devices undergo when moving across different cells, causing the loss of the information that allows the data to be delivered timely.

One of the solutions proposed to face the problems provoked by handover is the context transfer. The mean purpose of context transfer for QoS is to reduce the time period when a multimedia flow does not receive the contracted QoS due to a handover situation, by transferring in advance to the access router where the mobile is going to attach to the information required to maintain the QoS for that flow.

After a handover, in fact, it takes some time at the new access router to recover the parameters describing the service requested by that mobile, especially to exchange signaling information. The time wasted in this signaling exchange can be harmful for those applications requiring a timely delivering of packets and that the level of service remains constant for all the duration of the transfer, the real time applications. This is why it is believed that transferring the information associated to a mobile (the context) before handover execution can be helpful in order to keep the service.

In this thesis we first present an introduction to the topics related to QoS, mobile environments and how to attain QoS in such environments; afterwards we describe in detail the context transfer problem, illustrating the advantages and the problems related to this approach. In the experimental part comparative performance tests are run to check out whether context transfer can be effective in the scenario described.

CONTEXT TRANSFER FOR QUALITY OF SERVICE IN RADIO ACCESS NETWORKS

Abstract

Il mondo delle telecomunicazioni sta cambiando velocemente oggigiorno e si stanno sviluppando nuovi servizi, nuovi dispositivi, nuove tecnologie, e tutte promettono una migliore e ambiziosa qualitá dei servizi forniti al consumatore. La direzione verso cui questo sviluppo sembra essere orientato é senza dubbio il wireless, e la sfida é quella di fornire questi servizi su un link radio. Questa scelta di mercato ha come conseguenza che queste nuove applicazioni devono essere adattate ad un ambiente wireless, con tutti i problemi correlati.

In particulare, uno degli argomenti piú discussi é su come avere la possibilitá di usare applicazioni real time anche su un link wireless: per ottenere questo, devono essere soddisfatti gli stringenti requisiti che questo genere di applicazioni impongono, principalmente sulla tempestivitá nella consegna dei dati. Il problema principale a questo proposito é rappresentato, su un link wireless, dall'handover che I dispositivi mobili subiscono quando si muovono attraverso differenti celle, causando la perdita dell'informazione che permette ai dati di essere consegnati tempestivamente.

Una delle soluzioni proposte per risolvere I problemi provocati dall'handover é il context transfer. L'obiettivo principale del context transfer per la QoS é di ridurre il periodo di tempo in cui un flusso multimediale non riceve la QoS precedentemente contrattata a causa di una situazione di handover, trasferendo in anticipo all'access router a cui il dispositivo mobile andrá a connettersi l'informazione richiesta per mantenere la QoS per quel flusso.

Dopo un handover, infatti, ci vuole un po' di tempo affinché il nuovo access router recuperi I parametric che descrivono il servizio richiesto dal dispositivo mobile, specialmente a causa degli scambi di informazione di segnalazione. Il tempo impiegato in questa operazione puó essere deleterio per quelle applicazioni che richiedono una tempestiva consegna dei pacchetti e che il livello del servizio rimanga costante per tutta la durata del trasferimento, le applicazioni real time appunto. Questa é la ragione per cui si crede che trasferire l'informazione associata ad un dispositivo mobile, il contesto, prima dell'esecuzione di un handover puó essere utile a mantenere il servizio.

In questa tesi prima presentiamo un'introduzione agli argomenti correlati alla QoS, ambienti mobili e come ottenere la QoS in questi ambienti; in seguito descriviamo in dettaglio il problema del context transfer, illustrando vantaggi e svantaggi derivanti da questo approccio. Nella parte sperimentale, eseguiamo test comparativi per verificare se il context transfer puó essere efficace nell'ambito dello scenario descritto.

Table of contents

1 Introduction

Issues related to QoS on fixed networks have been widely dealt with nowadays. Two structures were introduced to support the required service either for single flows or for aggregate of flows belonging to a specific class of services. In the former case, the Integrated Services [15], the service is guaranteed for every user who requires it in a per-flow fashion. In the latter case, the Differentiated Services [21], this guarantee is ensured for a whole class of service but not for the single flow, which will be able to achieve the desired QoS only in a statistical fashion. A common approach to the two above-mentioned was proposed for realizing an IntServ architecture at the boundaries of the network over a DiffServ-centric core network [26].

The mainstream work on QoS and multimedia concentrates on fixed networks and does not address the additional complications imposed by mobility, which has been an area of growing research in the last few years. The main reason lies in the fact that the number of mobile service customers has risen significantly in the last few years, so that the need for new kinds of services, which can satisfy the requests of new users, has become evident. In order to provide a certain QoS for a service on an IP mobile access network, problems mainly related to user mobility and the capability of the network to maintain the agreed QoS regardless of the mobile node's movements arise.

In IP access networks that support host mobility, the routing paths between the host and the network may change frequently and rapidly. In some cases, the host may establish certain routing-related services on the subnets that are left when the host moves. Examples of such services are AAA, header compression and QoS. In order for the host to obtain those services on the new subnet, the host must explicitly re-establish the service. This causes long latencies and service disruption, a situation that real-time applications cannot in any way tolerate, and waste of radio channel bandwidth to re-establish the service.

In the process of establishing the new routing path, the nodes along the new path must be prepared to provide similar routing treatment to the IP packets as was provided along the old routing path. An alternative is to transfer information on the existing state associated with these services, or *context*, to the new subnet, a process called "*context transfer*". The transfer of context information associated with the mobile node's active micro flows can help maintaining the service level during handovers. The main reason in fact to perform context transfer is the will to sustain the service provided to a mobile node's traffic during handovers.

Context transfer is a mechanism for establishing sufficient conditions at one or more access routers to fully support the microflow(s) of a mobile node. After the completion of a context transfer, an access router will be capable of forwarding the IP packets to and from the mobile node without disruption of the established service; the price we have to pay for seamless information transfers is the need for a new protocol to handle context transfer. The context information to be transferred comprises a number of feature contexts like header compression, security, buffering and indeed QoS. However, for each feature to which context transfer is applicable, one must identify the data that must be transferred, and any unique requirements that are relevant.

The Quality of Service offered to an access router is included in the context of the support provided to the IP traffic. The ability of a new access router to support the same Quality of Service after handover is determined by the availability of the necessary router resources, by the availability of unused bandwidth on the links the traffic must traverse to and from the router, and by the timely availability of the service support context at the router.

This context is initially established when the service is set up between the mobile node and the network, and changes over time as the components supporting the service features change state. In order for this context to be available at a new access router after handover, it must be replicated from the access router currently supporting the mobile node’s traffic. To not interrupt or degrade the service, the replicated context must represent the most recent support state.

In order to better understand the concept of QoS, Chapter 2 is a review of QoS in general and of network architectures supporting QoS; for each architecture is specified whether it meets the QoS requirements for an IP network. Chapter 3 presents an introduction to the problem of mobility, together with a description of the two macro-mobility protocols of the Internet world, Mobile IPv4 and Mobile IPv6. Chapter 4 describes some mobile networks: an introduction to GSM and its evolution towards a packet-switched service, GPRS, is given. There is also a brief description of the UMTS, the next generation mobile network. The final part of the chapter deals with IP QoS accomplishment issues within these architectures.

Chapter 5 depicts the problems related to mobility in the achievement of IP QoS and examples of architecture designed to face this challenge; the chapter ends providing introductory concepts about context transfer and showing the issues related to this approach. Chapter 6 brings ideas about context transfer together in order to discuss topics related to the problem more deeply; the requirement of a context transfer enabled framework, the possible mobility scenarios and the way to discover a suitable access router to which a mobile node can attach after a handover are described. Chapter 7 builds on these introductory problems to examine the challenge of transferring the QoS context; the parameters of QoS context transfer are described, together with a proposed architecture that meets context transfer requirements for real time applications.

Chapter 8 presents the environment utilised for test execution, together with a description of the way in which the experiments have been run and the metrics adopted. Chapter 9 finally, shows the results obtained from the execution of the tests.

2 An overview of Quality of Service and QoS enabled networks

This chapter presents a brief introduction to QoS, with the explanation of some basic concepts, and to some network architectures capable of supporting QoS. A comparison of the features of these networks with the requirements that a QoS enabled network must retain is provided to check out if the examined architecture could be employed in the future to provide QoS.

2.1 Quality of Service in brief

The Internet was historically designed to be a reliable and simple network, where the intelligence was situated at the end hosts while the central routers were somewhat "stupid entities" with their main (or even only) task being to forward packets. The price paid for this simplicity is that an IP network is not able to offer users a wide assortment of services; the IP protocol [1], provides addressing, packets forwarding, fragmentation and reassembly but no services other than best effort.

The reliability of the network is assured by upper level protocols, such as TCP [2] that keeps track of lost and duplicate packets, ensuring an ordered delivery of segments to the upper levels. However, TCP does not assure timely delivery of segments, as the signalling exchanges typical of TCP, required for warranting the reliability of service and establishing connections, take a considerable amount of time to be performed. Traditional applications, such as file transfer, e-mail and in general all those requiring only data transfers do not suffer from this delayed delivery, as their primary concern is that all of their data should arrive uncorrupted to the recipient.

The rise and the growth of a new kind of applications, like multimedia, video-conference and so forth, which need strict time requirements in terms of timely delivery and delay constraints, has made clear that the Internet as it was first designed, cannot be sufficient for its modern tasks. QoS, and the way to provide the requested QoS to the expecting applications, has become a key topic and a wide field of study worldwide [3].

2.2 Basic Quality of Service parameters

However, what exactly is QoS? Huston, in [4], tries to answer to this question. QoS is a broad area and so it is difficult to give a definition comprising the whole concept. In the context of network engineering, quality can describe the transferring of data in such a way that some particulars features are addressed (see timeliness, delay etc.) and in this sense it is an objective parameter. From user point of view, quality can involve parameters such audio cleanliness, video resolution and so on, which are subjective, so that it is difficult to give a general classification. Cost is also a very important topic to deal with when considering QoS provision.

Chalmers and Morris [5] divide these features into two groups, technology-based and user-based, according to the above discussion.

Category	Parameter	Description/Example
Timeliness	Delay	Time taken for a message to be transmitted.
	Response time	Round-trip time from request transmission to reply receipt.
	Jitter	Variation in delay or response time.
Bandwidth	System level data rate	Bandwidth required or available, in bits or bytes per second.
	Application level data rate	Bandwidth required or available, in application specific unit per second. I.e., video frame rate.
	Transaction rate	Number of operation requested or processed per second.
Reliability	Mean time to failure (MTTF)	Normal operation time between failures
	Mean time to repair (MTTR)	Down time from failure to restarting normal operation.
	Percentage of time available	MTTF/(MTTF+MTTR)
	Loss or corruption rate	Proportion of total data that does not arrive or arrive different as sent.

Table 1: Technology-based QoS characteristics [5]

Table 1 shows technology-based parameters, and examples of the situations where they can be utilised; obviously, this table (and also the next one) deals with only part of these characteristics and cannot be exhaustive. The mathematical relations among these parameters are beyond the scope of this thesis and are not discussed. Further detail can be found in [6,7].

Table 2 below shows the main user-based parameters, and utilisation examples. It can be noticed how these parameters are more subjective than the technology-based ones:

Category	Parameter	Description/Example
Perceived QoS	Picture detail	Pixel resolution
	Picture colour accuracy	Maps to colour information per pixel.
	Video rate	Maps to frame rate.
	Video smoothness	Maps to frame rate jitter
	Audio quality	Audio sampling rate and number of bits
	Video/audio synchronization	Video and audio stream synchronization, i.e. speech/lips.
Cost	Per-use cost	Cost to establish a connection.
	Per-unit cost	Cost per unit time or per unit data.
Security	Confidentiality	Preventing unauthorized access to information
	Integrity	Proof that data sent was not modified in transit.
	Authentication	Identification of use or servic e provider to prevent masquerading.

Table 2: User based QoS characteristics [5]

2.3 Quality of Service Management

Blair and Stefani [8] define QoS management as the necessary supervision and control to ensure that the desired QoS properties are attained and sustained. To achieve these tasks, both static and dynamic management functions can be used. The former deal with features that remain unchanged during the session, the latter refer to features changing within the QoS environment in answer to network events, user behaviour and so forth. The static QoS management functions are applied at the beginning of the session and refer to operations able to settle the service before its initiation. The dynamic QoS management functions are deployed to actually provide the agreed level of service.

The static QoS management functions are summarized by [5] in the following manner:

- *Specification:* The specification function is the definition of QoS requirements or network capabilities.

- *Negotiation:* The negotiation function deals with the reaching of an agreed specification between the user, who requires a certain QoS and the network provider who offers another level of QoS according to network conditions and resource availability.
- *Admission control*: The admission control function executes the comparison of required QoS and capability to meet requirements; it is a very important phase because it is strictly related to congestion control and QoS is achieved also avoiding congestion.
- *Resource reservation:* Resource reservation represents the actual allocation of resources, in order to guarantee QoS and can be exploited by various means; an example is RSVP [18].

The dynamic QoS management functions are recapitulated by [5] in the following manner:

- *Monitoring:* Monitoring can be seen as the measuring of QoS actually provided in order to ensure the agreed level to the user.
- *Policing:* The policing functions are very important in order to maintain the level of traffic on the network under congestion threshold; they are deployed to ensure that all parties adhere to QoS contract: the most classical example is shaping the user transmission rate to fulfil the agreed value.
- *Maintenance*: Maintenance is concerned with the modification of system parameters to maintain QoS, so is strictly related to policing.
- *Renegotiation:* This function is the dynamic renegotiation of a contract and can be caused by a lot of different situations, such as sudden change in network conditions.
- *Adaptation*: When dynamic modifications are required, it is necessary to adapt of applications to QoS changes in the system.

A more detailed explanation of these functions can be found in [9].

2.4 Quality of Service enabled architectures

The main task of these QoS-enabled architectures is to supply and guarantee QoS according to user Service Level Agreements (SLA); to obtain this, the network must satisfy some basic requirements, such as differentiating different levels of service in order to carry out an efficient scheduling of resources. To achieve this, the network provider must take care that the load on the network does not exceed congestion limits and the user adheres to the agreed SLA. Fundamental keys for the success of networks supporting QoS are also an efficient routing mechanism and an optimisation of the use of the resources by which considerable savings in cost and proficiency of service can be attained.

The paper [10] draws the issues explained above presenting the basic assignments a QoS based network should retain: as it can be seen, they are strongly related to the QoS management functions.

- *Distinguish different service levels*: This is accomplished by means of traffic classification, traffic marking either at the edge routers or at the core routers depending on the QoS network architecture.
- *Differentiate network service behaviour to suit every SLA:* This task reflects itself mainly in the scheduling of network resources according to the contract subscribed by the user.
- *Control user traffic:* These actions must be deployed to maintain user traffic under a specific threshold. Examples are traffic metering and traffic conditioning.
- *Control of network load:* QoS can be offered to users only under light traffic conditions, so admission control and congestion avoidance are necessary.
- *Ensuring routing efficiency:* To attain this goal, it is obvious that the size of routing tables and routing table scale time must be kept low.
- *Optimise the use of resources:* The issues related to resource optimisation suggest the employment of traffic engineering within our network architecture.
- *Ensure architecture success:* What makes the architecture really good? Improved traffic control and network management can be suitable solutions.

2.5 Overprovisioning

The following sections give a description of the main architectures proposed to sustain QoS, analysing whether they accomplish the requirements, outlined above, of an IP network capable to provide QoS. We begin with an overview of overprovisioning.

Overprovisioning is a technique used to provide QoS at the beginning of the nineties and consists simply of provisioning an amount of bandwidth much larger than the actual user requirements; no other facilities in fact, are provided. Clearly, this approach is not suitable for many new applications because it cannot ensure timing requirements. Overprovisioning tries to avoid congestion by reserving more resources than necessary, but does not address a timely delivery of the information.

The unsuitability of this approach with the above described network challenges is clearly shown by the following considerations [10]: before all, this architecture does not distinguish between different levels of service, offering the same best effort service to everybody since it is essentially based on the IP protocol. Routing efficiency, moreover, is very low since a lot of time is wasted on accessing and scanning large routing tables. Finally, reserving more resources than necessary does not meet the task of resource optimisation; in other words, this is not absolutely a suitable approach in order to achieve QoS.

2.6 Asynchronous Transfer Mode

Asynchronous Transfer Mode (ATM), such as MPLS, can be considered a low layer architecture, thus able to provide QoS operating on sub-IP layers; since ATM and MPLS are neither link layer nor network layer architectures (IP in fact can lie over ATM for example), they can be deemed as "2.5 layer architectures".

Asynchronous transfer mode (ATM) [11] is a high-performance, cell-oriented switching and multiplexing technology that utilizes fixed-length packets (cells) to carry different types of traffic. ATM is a connection-oriented technology that will enable networks to provide QOS through multiple ATM classes of services. The length of the cells is set to 53 bytes divided into 48 for the payload and by 5 for the header; this choice represents a good trade-off between the flexibility given by small cells and a large header overhead.

ATM provides its services by instituting virtual channels from source to sink during all the time of the connection, reserving resources that will be guaranteed to the user after an initial phase of set-up where the virtual circuit is established and resources reserved. A virtual channel connection (or virtual circuit) is the basic unit, which carries a single stream of cells, in order, from end user to end user.

In order to simplify routing, more virtual channels can be aggregated in a virtual path: in this case, the ATM network does not route cells belonging to a particular virtual circuit. All cells belonging to a particular virtual path are routed the same way through the ATM network, thus resulting in faster recovery if major failures happen. Figure 1 shows the relation between these entities:

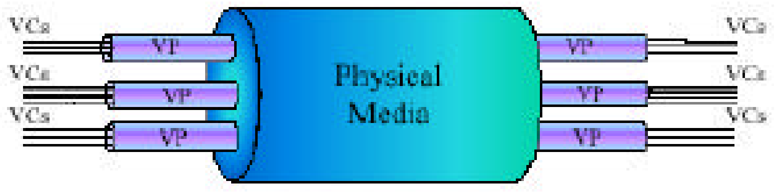


Figure 1: Relations between virtual path, virtual circuit and physical link

There are four classes of services in the ATM standard, each characterised by its own requirements on three typologies of parameters, as summarized in the table below:

PARAMETERS	CLASS A	CLASS B	CLASS C	CLASS D
Time relation	Required (real time)		Not required (not real time)	
Bit rate	Constant (CBR)	Variable (VBR)		
Connection mode	CO			CL

Table 3: ATM classes of service

CBR (class A) is used for emulating circuit switching; the cell rate is constant with time. CBR applications are quite sensitive to cell-delay variation and require a connection oriented environment. Examples of applications that can use CBR are telephone traffic and PCM voice. VBR-rt (class B) is similar to VBR–NRT but is designed for applications that are sensitive to cell-delay variation in a connection-oriented environment. Examples of real-time VBR are voice with speech activity detection (SAD) and interactive compressed video.

VBR-nrt (class C) allows users to send traffic at a rate that varies with time depending on the availability of user information in a connection-oriented environment. Statistical multiplexing is provided to make optimum use of network resources. Multimedia e-mail is an example of VBR–NRT.

Class D pertains to those VBR applications that do not require a real-time service and work in a connection-less environment. Practically, they are data applications, such as file transfer.

Two other service classes, introduced after the definition of the previous four, must be mentioned: these classes are not shown in the table since they comprise concepts common to the others. The Unspecified bit rate class (UBR) is entirely best effort: there are no guarantees of the delivery of cells and timing constraints are not met; the only attraction of UBR is the low cost. The Available bit rate class (ABR) is a middle way between UBR and CBR/VBR: in low traffic conditions, the network behaves as best effort, while in the imminence of congestion advices applications to reduce their cell rate. ABR does not guarantee all performance requirements, but only a minimum cell transmission rate. A good overview of ATM is provided in [13] and [14].

The IP over ATM Working Group of the Internet Engineering Task Force (IETF) is tasked with developing standards for routing and forwarding IP packets over ATM sub-networks [12]. This technique was introduced in the middle of the nineties, when the throughputs of ATM switches and the routing facility given by virtual paths attracted many IP networks operators. They wished to improve network performance by using the virtual connections. Nowadays, anyway, there exist routers even faster than ATM switches and moreover, the different managing of IP routing and ATM routing makes this approach no longer attractive.

With regard to the above considerations, it is possible to check if IP over ATM networks fulfil QoS requirements [10]. As we have seen, the architecture makes distinctions between various service levels (this function is obviously referred to ATM) by allowing the presence of different service classes tied to the requirements of every kind of applications, both real time and not. The employment of virtual circuits helps to effect resource scheduling and mostly to improve routing scalability: however, the overall routing efficiency of IP over ATM is very low since IP and ATM routing are on separate plans and it is difficult to smooth the differences in these two different approaches. Furthermore, the best effort service offered by IP implies that the other network functionalities, as described in section 2.4, are referred to ATM; finally, ATM overlayering makes it difficult to ensure the commercial success of this approach, since industries seem more attracted by all-IP networks.

2.7 Multiprotocol label switching

Multiprotocol label switching (MPLS) [27] is an IETF standard intended to enhance the speed, scalability and service provisioning capabilities on the Internet. MPLS uses the technique of packet forwarding based on labels and decouples packet forwarding from routing, facilitating the provision of varied routing services independent of the packet forwarding concept.

In MPLS, traffic aggregates are associated with Label Switched Paths (LSP: an ordered, unidirectional list of MPLS nodes traversed by packets all belonging to the same FEC) through an IP network. When a packet goes into a MPLS network, it is assigned to a Forward Equivalence Class (FEC); all packets that belong to a particular FEC and travel from a given source node will follow the same path. An FEC is a group of layer-3 packets that are forwarded in the same manner. All packets in this group follow the same network path and have the same prioritization; for example packets that belong to the same FEC join the same path and require the same QoS.

In traditional IP forwarding, every router in the path followed by the packet examines the header IP in order to redirect the packet; in MPLS, the assignment of a particular packet to a particular FEC (and so the examination of the header IP) is done just once, as the packet enters the MPLS-enabled network. The FEC to which the packet is assigned is encoded as a short 20 bit fixed length value known as *label*. A label is a short, fixed length, locally significant identifier, which is used to identify a FEC. When a packet is forwarded to its next hop, the label is sent along with it; all of the packets marked with the same label are forwarded in the same way.

The use of a label is meaningful only locally between a pair of Label Switching Routers (LSR: an MPLS node capable of forwarding layer 3 packets); this fact assures MPLS a large scalability also into large networks, since labels are replaced hop-by-hop and are not used along the entire path. At subsequent hops, there is no further analysis of the packet's network layer header. The old label is replaced with the new label, and the packet is forwarded to its next hop. Figure 2 shows the packet headers stack in MPLS architecture.

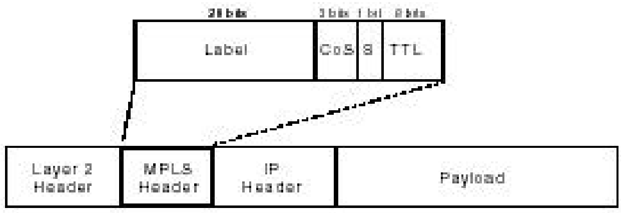


Figure 2: Headers encapsulation in MPLS architecture [29]

Label swapping is the act of forwarding a packet by using labels. To forward a labelled packet, an LSR checks the label, forwards the packet to the next hop (using the below described NHLFE) and attaches a new label to the packet. To forward an unlabelled packet, an LSR simply examines the IP header (or in general the network layer header) to assign the packet to its own FEC, and then maps this to an NHLFE, performing the other required operations. Using the information in the NHLFE, the LSR determines where to forward the packet. At the egress of the MPLS network, the LSR takes off the label from the packet that can be routed in the traditional way.

To actually execute the forwarding, a MPLS node uses a next hop label forwarding entry (NHLFE): there is at least one NHLFE for each FEC flowing through the node.

A NHFLE contains the following information:

1) The packet's next hop.
2) The operations to perform on the label stack (for example the hop-by-hop label substitution).
3) The data link encapsulation to use when transmitting the packet.
4) The way to encode the label stack when transmitting the packet.

2.7.1 Traffic engineering

Traffic engineering [28] is the process of selecting network paths so that the resulting traffic patterns achieve a balanced utilization of resources. Traffic engineering is most important in networks where multiple parallel or alternate paths are available. Routing based on conventional IP algorithms (IGP) in fact may select network paths that result in unbalanced resource utilization; in these environments, some network resources are overutilized, while others are underutilized.

In order to accomplish this effort, MPLS specifies that packets can follow a particular path, maybe one in which resources are underutilized. The cost of this approach is the constant monitoring of resource utilizations in the paths and of packet data streams, sometimes resulting in intensive processing at routers.

2.7.2 Label distribution protocol

The label distribution protocol (LDP) is defined in [30] as a set of procedures by which an LSR informs another one of the meaning of labels used to forward traffic between and through them. It is the set of procedures and messages by which LSRs establish LSPs through a network by mapping network layer routing information directly to link layer switched paths.

Anyway, the MPLS architecture does not require that the protocol used is unique; in fact, the protocol used should depend on what requirements must be met by a particular network; there are a number of label distribution protocols under standardization either as development of already existing protocols, or just new ones. LDP associates a Forwarding Equivalence Class (FEC) with each LSP it creates. The FEC associated with an LSP specifies which packets are "mapped" to that LSP.

LDP alone cannot meet QoS requirements. For the purpose of supporting QoS, LDP should be capable of reserving resources along a selected LSP. To achieve this, it is possible either to take an existing protocol used for resource reservation (RSVP), and enhance it with label switching constraints or to get a protocol exploited for label switching (LDP) and extend it with resource reservation features.

More information about the applicability of LDP can be found in [31]; [32] in its turn, provides some examples of structures that aim to achieve QoS in MPLS networks.

2.7.3 Conclusions on Multiprotocol Label Switching

In the above sections, we have illustrated briefly the main characteristics of MPLS; there is a lot of ongoing related work about this topic since it is a relatively new field of study; however, discussing it is something that is outside the scopes of this thesis.

We now present a discussion about the fulfilment of MPLS to QoS network requirements [10]. The important function of distinguishing different service levels, fundamental in every

QoS based architecture, is achieved by introducing opportune FECs for every SLA. The network is also able to modify its behaviour to adapt the service to the agreements by setting appropriate LSPs; this makes it possible to control the load in the network if resources are properly reserved for LSPs.

Label swapping and switching are the major advantages of this architecture; the use of labels, and thus avoiding the need to examine the header IP hop by hop ensures great routing efficiency. The optimisation of use of resources is carried out with the utilization of traffic engineering. Finally, we can affirm that this, relatively new, architecture can be predicted as successful since it introduces many revolutionary concepts (that at the moment seem to meet the QoS requirements very well) and it can be profitably deployed as an "island" in the whole network through which packets can be tunnelled to provide end-to-end services.

MPLS can be used with upper level QoS mechanisms with the goal to obtain an overall architecture that holds both the advantages of MPLS and end-to-end architectures. For example, [35] proposes a network structure that delivers end-to-end QoS for DiffServ and Intserv networks, with an underlying technology based on MPLS.

2.8 Integrated services

Integrated Services (IS) [15] is a model developed by the IntServ IETF working group to optimize network and resource utilization for new applications, such as real-time multimedia, which require QoS guarantees. This aim is achieved by considering two kinds of traffic on the Internet, the best effort one and the one requiring a guaranteed QoS.

The main concept behind Integrated Services is that network resources must be properly managed in order to ensure QoS to applications that require it: admission control and resource reservation are necessary and can be considered the building blocks of this architecture.

2.8.1 Traffic control

In order to support the IS model, an Internet router must be able to identify single flows and assign them the required QoS; this router function is known as *traffic control* and is comprised of three functional elements, which can be intended as part of the same process. When a packet enters an Integrated Services domain it is handled by these elements:

Packet classifier

The packet classifier identifies IP flows as belonging to a particular class of service. All of the packets that are classified in the same class get the same treatment from the packet scheduler. The selection of the class is achieved by examining the packet header and/or some additional classification numbers added to each packet. Each class can include a large category of flows or even only one. Note that the choice of the class is not a univocal matter, but is something that is demanded to each router. Different routers can thus classify the same flow as belonging to different classes, depending on their position in the network (edge routers or core routers) or on other issues.

Packet scheduler

The packet scheduler, which is implemented obviously at the points where packets are queued, manages the forwarding of different packet streams using a set of queues, assigning to each flow a particular queue based on its class of service, by using opportune scheduling mechanisms, according to the classification given by the packet classifier. Another feature of the packet scheduler is the policing and shaping of traffic flows not corresponding to agreements between user and network administrator.

Admission control

Admission control implements the decision algorithm that a router or host uses to determine whether a new flow can be granted the requested QoS without affecting earlier guarantees provided to other flows. If this is not possible and there are not enough resources available, the request must be rejected. One must not get confused with policy control, which is the way used to avoid users overcoming agreed traffic characteristics.

Since the Integrated Service model provides per-flow reservations, each flow is assigned a flow descriptor. The flow descriptor defines the traffic parameters and QoS characteristics for a specific flow of data packets. In the integrated service specifications, the flow descriptor consists of a filter specification (filterspec) and a flow specification (flowspec), as illustrated in

Figure 3.

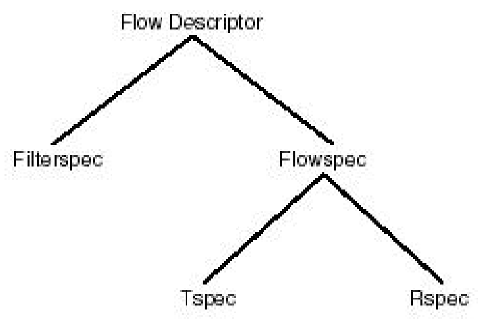


Figure 3: Flow descriptor

The filterspec is used to identify the packets that belong to a specific flow using header information. The flowspec contains a set of parameters called the *invocation information*. It is possible to sort the invocation information into two groups: traffic specification (Tspec) and service request specification (Rspec). The Tspec describes the traffic characteristics of the requested service. The Rspec specifies the QoS the application wants to request for a specific flow.

2.8.2 Integrated Services classes of service

The Integrated Services model uses different classes of service that are defined by the Integrated Services IETF working group. The main difference between these two classes lies in the constraints they provide to the flows they contain.

Controlled load service (CLS)

The controlled load service class [16] is intended to support a broad class of applications, which have been developed for use in today's Internet, but are highly sensitive to overloaded conditions, such as "adaptive real-time applications". The service provided to applications is minimal in order to approximate best effort; an application requiring a CLS thus expects to find a best effort network with light traffic condition as forwarding scenario. This means very few losses of packets and that the transit delay induced by the network is close to the minimum transit delay of successfully transmitted traffic.

Guaranteed service (GS)

The Guaranteed Service model [17] provides functions that assure that datagrams will arrive within a guaranteed delivery time. Every packet satisfying traffic specifications will arrive within a maximum delay specified in the flow descriptor. The delay consists of the fixed delay and the queuing delay; the fixed delay is a path property and is determined by the setup mechanism. Guaranteed Service is used for applications that need a guarantee that a datagram will arrive at the receiver not later than a certain time after it was transmitted by its source, such as real-time multimedia applications, like video and audio broadcasting. The GS service controls only the maximum delay; thus it does not control the minimum delay or control or minimize the jitter.

2.8.3 Resource Reservation Protocol

Integrated Services use the Resource Reservation Protocol (RSVP) [18] for the signaling of the reservation messages. The Integrated Services instances communicate via RSVP to create and maintain flow-specific states in the endpoint hosts and in routers along the path of a flow. Host requests for specific qualities of service from the network for particular application data streams or flows use RSVP. Routers also use RSVP to deliver quality-of-service (QoS) requests to all nodes along the path(s) of the flows and to establish and maintain the state to provide the requested service.

The main RSVP features are:

- RSVP makes reservations for unidirectional data flows.
- RSVP is receiver-oriented, i.e., the receiver of a data flow specifies desired QoS when receiving a flow.
- RSVP maintains a "soft" state in routers and hosts.
- RSVP supports both IPv4 and IPv6.

In brief, an RSVP operation can be explained thus: a source sends a message (PATH) containing its flow specifics to the receivers. When a receiver gets this message, it sends a message enclosing the desired QoS flow specifics (RESV) along the path sink-source. Routers must keep track of the previous router that forwarded them the path message because the Resv message must follow the same reverse path (hop-by-hop) of the path message in order to ensure QoS.

To obtain this, it is necessary that routers hold information about all of the flows to which they are reserving resources; in other words, they have to maintain a soft state, information that can be used to make resource allocation decisions about the packets belonging to the flow. The main difference between soft state and hard state is that the first does not need to be created and removed by signalling. Soft state is middle way between a merely connection-less approach in which there is no maintenance of state at the routers and a connection-oriented approach where routers maintain hard state [19].

The soft state and receiver oriented nature of RSVP leads to many facilities: one is that it is very easy to modify the level of allocated resources provided to a receiver, since each receiver periodically sends refresh messages to keep the soft state on place and eventually to change it. In the event of a host crash, resources allocated by that host to a flow will time out and be released. If the crash is of a router or of a link, routing protocols create a new path from sender to receiver and the receiver's next RESV message will follow the new path.

The main issue with IntServ/Resv [20] approach is that Integrated Services is based on flow-state and flow processing. If flow processing rises dramatically, it could become a scalability concern for large networks.

2.8.4 Discussion on Integrated Services/Resource Reservation Protocol

The IntServ/RSVP technology introduces many advantages with respect to previous technologies, since it provides the applications with a strong guarantee that the service level will be accomplished; however, it presents important drawbacks that constitute an obstacle to its deployment on the Internet [10]. The multi-field traffic classification permits the differentiation of services, and resource reservation is achieved on a per-flow basis; on a per-flow basis are also resource scheduling, metering and conditioning of user traffic.

This per-flow nature of IntServ/RSVP, on one hand makes sure the applications receive the contracted QoS, since for each flow resources are analysed and managed, on the other hand creates a very serious problem of scalability. In a scenario where large networks make use of the IntServ/RSVP model, it would become difficult to handle resources when several requests arrive at an IntServ router; moreover, to be sure that resources are actually reserved for a flow, it is likely that it could not be possible to share unused reserved resources. This is not a good optimisation of the use of resources. These reasons make this approach unsuitable for providing QoS, so that a Differentiated way to handle resources, or at least a combination of these two latter, discussed below, seem more attractive at the moment.

2.9 Differentiated Services

The Differentiated Services (Diffserv) architecture [21] was introduced to avoid the scalability and complexity problems of Intserv. Scalability is achieved by offering services on aggregate basis rather than per-flow and by forcing the per-flow states to the edges of the network as much as possible. The service differentiation is achieved by means of the Differentiated Service (DS) field in the IP header and the Per-Hop Behavior (PHB) as main building blocks. At each node packets are handled according to the PHB invoked by the DS byte in the packet header.

A central component of differentiated services is the Service Level Agreement (SLA). The SLA is a service contract between a customer and a service provider that specifies the forwarding service a customer should receive. The service provider must take care to supply the contracted QoS in accordance with the agreed SLA.

In order to differentiate packets from different users the architecture makes use of the TOS field in the IPv4 header and of the traffic class octet in IPv6 [22].

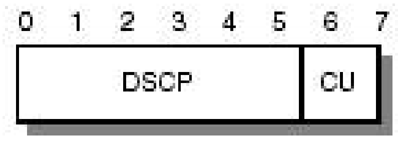


Figure 4: The DS field

In Figure 4 above is shown the DS-field: six bits of the field are used for the Differentiated Services Codepoint (DSCP), used by each node in the network to select the PHB. The last two bits are currently unused. The DSCP bits of the IP packet header are set at the network boundaries; the marking of these bits indicates to the internal nodes (core routers) the treatment type in the packet forwarding path; in other words, the DSCP field specifies the class of service in which a given flow belongs.

2.9.1 Per-hop behaviors

The PHB can be defined as a set of parameters inside a router that can be used to control how packets are scheduled onto an output queue. Two PHB groups have been defined, the assured forwarding AF PHB [23] and the expedited forwarding EF PHB [24].

Assured forwarding PHB group

The Assured Forwarding (AF) PHB group is used by a Diffserv domain to provide the assured service to the customers that require reliable services even when congestion occurs in the network. The AF-PHB group provides the means to offer different levels of forwarding insurances to packets according to four defined AF classes with three levels of drop precedence within each class. The level of assurance provided to a packet depends on the level of resources assigned to its class and inside the class on the drop precedence level of the packet. The AF PHB group could be used to implement, for example, the so-called Olympic Service, which consists of three service classes: bronze, silver, and gold. Packets are assigned to these three classes so that those in the gold class experience lighter loads (and thus have greater probability of timely forwarding) than packets assigned to lower classes; packets are also assigned a drop precedence within each class.

Expedited forwarding PHB-group

The Expedited Forwarding PHB provides tools to build a low loss, low latency, low jitter, assured bandwidth, end-to-end service through Diffserv domains. [25] defines this service "Premium", provisioned according to peak capacity profiles that are strictly not oversubscribed and that is given its own high-priority queue in routers. The SLA specifies a peak bit rate that is the upper transmission limit for the customer: if the user transmission rate does not exceed the specified peak bit rate, he will then receive the agreed bandwidth. Boundary routers set the premium bit of the user's packet requesting the service; the routers along the path must implement different mechanisms of scheduling priority, such as simple priority queuing or weighted round robin scheduling.

2.9.2 Traffic conditioners

A packet enters a DS domain by way of a node placed on the boundaries, the boundary node, which has the function of conditioning of the traffic forwarded through it. The traffic conditioning is done inside of a boundary node by a *traffic conditioner*, which consists of the following components:

Classifier

A classifier selects and forwards packets based on their packet header. The DS model specifies two types of packet classifiers: Multi-field (MF) classifiers, which can classify based on the value of a combination of one or more header fields, for example the IP address and the port number, and Behavior Aggregate (BA) classifiers, which only classify based on the bits in the DS field.

Meter

Traffic meters measure whether the forwarding of the packets selected by the classifier corresponds to the traffic profile that describes the QoS for the SLA between customer and service provider. When a packet does not fulfill this profile, the meter triggers the use of other conditioning functions by passing them state information.

Marker

DS markers set the DS field of the incoming IP packets to a particular bit pattern, adding the marked packet to a particular DS behaviour aggregate. One of its functions is to mark the first six bits of the DS field in order to define a PHB, according to the SLA between service provider and customer.

Shaper/Dropper

Shapers delay some or all of the packets in a traffic stream in order to bring the stream into compliance with a traffic profile. A shaper usually has a finite-size buffer, and packets may be discarded if there is not sufficient buffer space to hold the delayed packets. Droppers discard some or all of the packets if they are overcoming SLAs to make them adapt to a correct rate according to configured traffic policies.

2.9.3 Discussion on Differentiated Services

DiffServ seems without doubt a better network architecture able to provide QoS for demanding applications; it overcomes the IntServ scalability drawback thanks to the use of the aggregate behaviours, and guarantees QoS to a class of service, mapping each flow in this class. This makes DiffServ scalable compared to IntServ, which, guaranteeing QoS on a per-flow basis is difficult to handle in large networks. The major issue in the DiffServ approach is that a flow is not assured the requested QoS, because, as noted above, the reservations are made based on aggregate forwarding behaviours [10].

The use of the DS field (in particular of the Differentiated Service CodePoint) permits the distinguishing of different service levels and the assigning a flow to its proper service class. The problems of scalability are solved by utilising a per-aggregate policy (in contrast with the IntServ one, which is per-flow), which reflects in achieving a per-aggregate scheduling of resources, metering and conditioning of traffic. The control of network load is performed using a per-aggregate queue management.

The routing algorithm employed, just like IntServ, is based on classical IP and so presents the same advantages and disadvantages; it is to say, instead, that the optimisation of resources can only be accomplished at node level, since only routers can execute proper scheduling operations. The scalability provided by this solution makes it the most attractive one for an incoming network, both fixed and mobile, this feature overcomes even the fact that a QoS guarantee is ensured only in a statistical fashion, within a class of service, and not in a per-flow manner.

2.10 Integrated Services over Differentiated Services

The two approaches described above can be combined to form an architecture holding QoS warranties typical of IntServ and scalability facilities characteristic of DiffServ. The RFC 2998 [26] illustrates a framework for exploiting IntServ over DiffServ. In the framework presented, end-to-end, quantitative QoS is provided by applying the Intserv model end-to-end across a network containing one or more Diffserv regions, as shown in Figure 5.

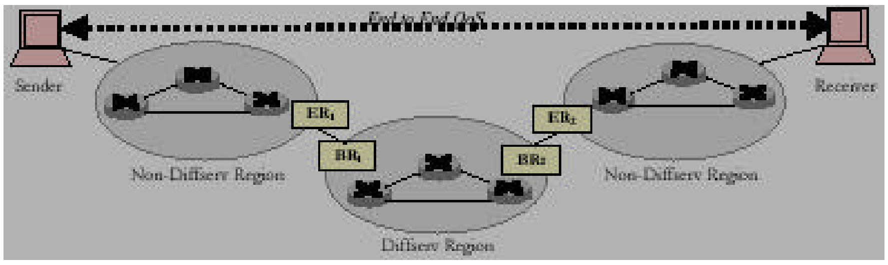


Figure 5: Reference network for IntServ over DiffServ framework

The DiffServ region can be considered, from the IntServ point of view, as virtual links connecting IntServ routers. The framework consists of the following elements.

The sender and the receiver can initiate an RSVP process based on the requirements of QoS aware applications running at the hosts; in fact, the use of IntServ at the boundaries allows the sender to use RSVP to make explicit resource reservations, while maintaining the scalability since the core is DiffServ. Edge and border routers are the edge devices whose functionality depends on the specific realization of the framework. Non-DiffServ region is an IntServ region, containing nodes capable of supporting IntServ functionalities. A DiffServ region can or cannot be RSVP aware depending on whether it contains RSVP routers or not; in the latter case, the DiffServ region is seen as a link by the IntServ region.

To ensure the correct functionality of the framework, a service mapping from IntServ services to DiffServ services must be attained depending on an appropriate selection of PHB, admission control and policy control on the Intserv, request based on the available resources and policies in the Diffserv. There is a known default mapping defined as is shown in Table 4:

IntServ	Qualifier	DiffServ PHB
Guaranteed Service	--	EF
Controlled Load	L	AF (higher priority)
	H	AF (lower priority)

Table 4: DiffServ-IntServ service mapping

The routers that support both IntServ and DiffServ carry out this mapping and they should have to be situated in DiffServ region boundaries. The network structure portrayed here is a very new field of interest and the subject of increasing research in the forthcoming years; the present state of the art authorizes us to say that it is a promising architecture able to provide, end-to-end QoS regardless of the size of the served network,. The only doubt could be whether the boundary IntServ regions can support the increasing number of Internet customers; in other words, the challenge is to overcome the (possible, not certain) scalability problems that could rise in the IntServ region.

2.11 Real time Transport Protocol

The Real time Transport Protocol (RTP) [33] is a transport layer protocol designed for those applications characterised by hard timing requirements, the so-called real time applications (i.e. audio, video, multimedia). Although RTP was designed to support multiparty real time applications (for example multimedia conferences), it can be suitably employed for the transport of data exchanged between point-to-point real time applications. RTP consists of a data and a control part. The latter is called RTCP (Real Time transport Control Protocol).

The name “transport protocol” could be misleading; RTP in fact, uses UDP (a transport layer protocol) [34] to achieve its tasks. The name emphasizes, however, that RTP is an end-to-end protocol. RTP has no notion of connection; it may operate over either connection-oriented or connection-less lower-layer protocols. The reason for which RTP lies on UDP and not on TCP is simple to understand; real time applications do not need the full reliability of the information carried by TCP, rather they require that this carrying is performed according to appropriate timing requirements. TCP, spending a lot of time in signalling to establish connection and recover errors, cannot be the right solution, so UDP was chosen to support RTP.

Why is there the need of RTP, then? To support real time traffic, the basic service provided by UDP could not be sufficient, the new protocol should have to offer means to handle these issues. RTP supplies:

- Identification of payload type, in other words of the kind of media identified by the information in the payload (Jpeg video, GSM audio for example).
- Timestamp: this function allows monitoring delays in data transferring so that upper level applications can perform, if needed, media synchronization.
- Sequence numbering: functionality offered to upper level applications so that they can eventually carry out loss recovery and packet re-ordering.

RTP offers a control protocol called RTCP that supports the protocol functionality and monitors the QoS offered to applications sustained by RTP. An RTCP message consists of a number of packets, each with its own type code and length indication. Their format is quite similar to data packets. RTCP is based on the periodic retransmission of control packets to those applications involved in a RTP session. RTCP offers the following functionalities:

- QoS monitoring and congestion control. RTCP packets contain the necessary information for QoS monitoring.
- Inter-media synchronization: the RTCP sender reports contain a timing indication and a corresponding RTP timestamp. These two values allow the synchronization of different media.
- Session size estimation and scaling: each session member sends RTCP packets periodically. The desire for up-to-date control information has to be balanced against the need to limit control traffic to a small percentage of data traffic, mainly with sessions consisting of several hundred members.

To achieve these functionalities, RTCP can operate in two modes:

- Loosely controlled session: this is the most utilized modality because it allows to provide basic features from which control information to manage multimedia application can be caught, at the expense of a minimum exchange of packets.
- Strong controlled session: this modality is used for those applications, characterised by particular requirements, which need a major number of parameters compared to the loosely controlled session.

3 An introduction to mobility

The concept of mobility is very wide, and it is difficult to summarize all the problems concerned in only one chapter; we have chosen to focus our attention on the differentiation of the various defined classes of mobility and on a description of handovers that occur in a mobile network. The last part of the chapter presents some basic notions about the two most important mobility protocols on the Internet, Mobile IPv4 and Mobile IPv6.

There is often confusion between “wireless networks” and “mobile networks”, and the two terms are used to indicate the same concept. A wireless network is simply a non-cabled network where the various entities communicate with each other by means of aerial interfaces: the employment of such a network can be useful for example when there is the need to install a LAN in an existing building. It could be too expensive to cable the entire building, so that a wireless network is a solution more suitable than, say, an Ethernet.

However, a fixed network ensures higher bandwidth than a wireless one; for this reason, a trade-off must be considered between the easiness of installation of a wireless network and the high bandwidth ensured by fixed networks. Oppositely to wireless networks, where terminals are fixed, in mobile networks they can move on keeping their session active: mobile networks support mobility. A mobile network is always a wireless network; the opposite is not true.

We provide now definitions of mobility-related terms, which will be widely used in the remainder of the thesis:

- *Mobile Node (MN):* It is an IP node capable of changing its point of access to the mobile network. Usually an end user host performs this process, but any forwarding node can also carry it.
- *Forwarding node:* It is a node that supports IP forwarding within the network.
- *Access Point (AP):* It is a layer two device connected to one or more access routers offering wireless connection to the MN.
- *Access Router (AR):* It is an IP router connected to one or more access points.

3.1 Classes of mobility

Three different classes of mobility can be distinguished:

1. *Terminal mobility:* Terminal mobility is characterised by the capacity of a terminal to access a telecommunication network while it is moving through it and to be identified and localised at every moment.
2. *Service mobility:* This kind of mobility allows users to access services pertaining to their profile utilising a network different from the one by which they subscribed to the contract. In the near future in fact, if users will subscribe to a service provider, specifying all of the services they are interested in (e.g. e-mail, videoconference, etc.) and afterwards they move into a network managed by another service provider, they must be able to receive the same subscribed services. In this way, the subscription to a service provider is transparent to the customer.
3. *Personal mobility:* Personal mobility deals with the ability of a user to access subscribed telecommunication services by means of every terminal (PDAs, laptops, desktops) and with the capacity of the network to identify and localise the user in order to provide the subscribed services. Personal mobility can be configured as the integration of Terminal mobility with Service mobility.

With regard to the technology utilised, mobile networks can be distinguished thus:

1. *I generation*: Analogical transmission systems, in general non-standardised by official institutes and thus not compatible each other.
2. *II generation*: Digital transmission systems standardised and interworking on an European scale, the Global System for Mobile communication is the most prominent example.
3. *III generation*: Multimedia global systems, integrating several services (including those provided by previous generations) accessible by a unique user terminal with the purpose of achieving Personal mobility. The Universal Mobile Telecommunication System is commonly addressed as "the" third generation mobile network.

3.2 The concept of handover

It is our intention to give some concepts about handover in this sub-section, since it is a topic of primary importance concerned with mobile networks and mostly because the context transfer is a way to minimize the troublesome effects of handovers on the QoS provided to demanding applications. The main facility that mobile networks offer to a mobile terminal is the possibility to change its point of attachment without losing the connection; the change of the point of attachment to the network during a connection is called *handover*.

3.2.1 Depth of handovers

When an MN performs a handover, a certain amount of signaling is required to handle this situation, mainly related to the fact that after the handover, the MN has changed its point of attachment to the network and consequently several configuration parameters have changed, one among all the IP address of the router that offers connectivity to the MN. In [36], the authors operate a distinction of handovers based on their depth; if during the handover the MN changes only its point of access (a layer 2, intra AR handover), only radio resources need to be handled by the handover control mechanism since the forwarding path does not change at all.

If the mobile node changes access router (inter AR handover) within the same access network, both radio and access network resources (new admission control and resource reservation phases needed) must be checked to ensure the service. Instead, if the mobile node moves into a different access network (or remains in the same, but attaches to an AR served by a different gateway), there can be an interval of time during which the mobile node does not receive the contracted QoS and routing changes involving a large part of the end-to-end path occur. Recent techniques, such as context transfer, aim to minimize and possibly eliminate, this period of disruption of service. The worst case is when the mobile node changes administrative domain; in this case, in fact, the amount of signaling (and thus the time required to receive the expected QoS) grows considerably, since authentication and/or authorization information must also be exchanged. Figure 6 shows an exemplification of the problem:

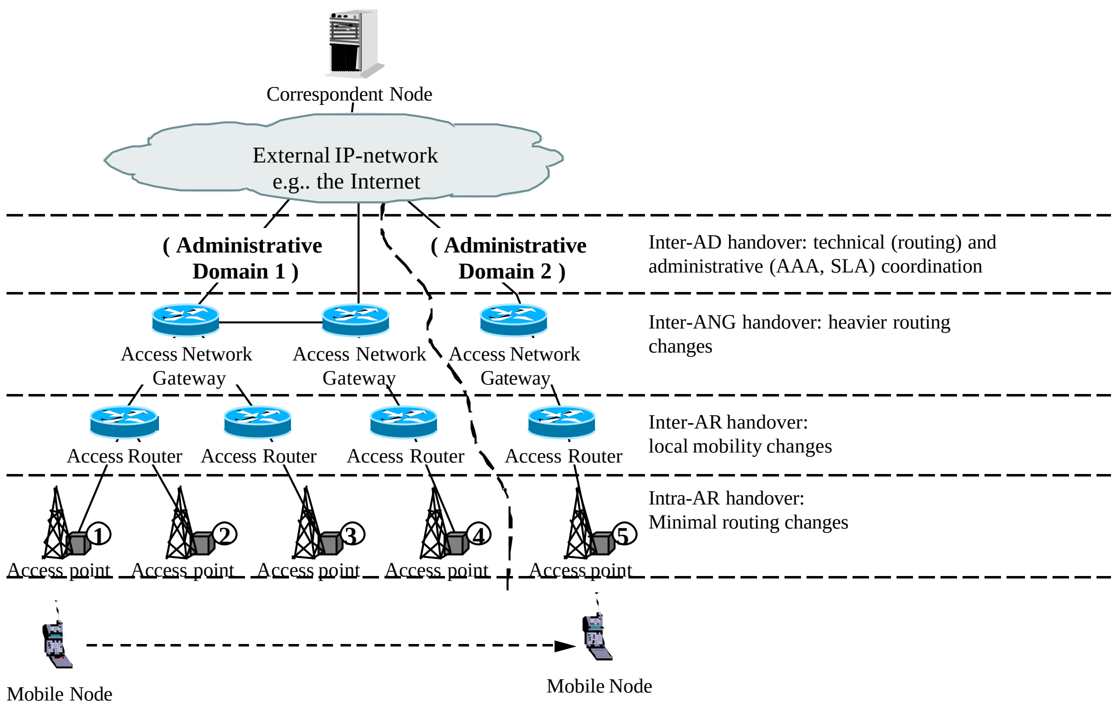

Correspondent Node
External IP-network
e.g.. the Internet
(Administrative Domain 1)
(Administrative Domain 2)
Inter-AD handover: technical (routing) and administrative (AAA, SLA) coordination
Access Network Gateway
Access Network Gateway
Access Network Gateway
Inter-ANG handover: heavier routing changes
Access Router
Access Router
Access Router
Access Router
Inter-AR handover: local mobility changes
Intra-AR handover: Minimal routing changes
1
2
3
4
5
Access point
Access point
Access point
Access point
Access point
Mobile Node
Mobile Node

Figure 6: Possible handover scenarios [36]

As defined in [37], handovers can be either "horizontal" or "vertical": in the former case, the layer 2 network interface seen by the IP layer does not change. This situation is typical of a handover in which the MN changes only its AP (and not the AR), however, it can also be an inter AR handover if the network interface of the new AR is the same as the old one. In the "vertical" case, the new AR uses a wireless interface different from the old one. This is typically the case of an inter AR handover, but if the MN changes just the AP of the same AR and the new AP uses a different interface from the one of the previous AP, then a vertical handover can also be performed.

3.2.2 Phase of handovers

We can identify the following phases in a handover [38]:

Initiation phase: The objective of this phase is to recognize the need for a handover and subsequently to initiate it. The request of a handover can be exploited either by the MN or by the network, generally when a degradation of the wireless link between the MN and its AR is perceived. Matters concerning load balancing in the network could also force the network to request a handover and move an MN towards a less loaded AR.

Decision phase: During this phase, entities involved in handover collect information about neighbour capabilities and localise the most suitable target AR, according to its resource availability.

Execution phase: In the execution phase the actual transfer of the MN from the old AR to the new one is carried out. A soft handover is deemed as one in which the MN maintains connections with both the old and the new AR, simultaneously; if this does not happen, the handover is called hard.

The kind of handover that holds the major interest in this thesis is the seamless handover, intended as a handover where minimum packet losses and delays are experienced. Seamless handovers are a way to provide the desired level of QoS even during the phase of changing the point of attachment to the network; context transfer is based on this fundamental concept.

3.3 Mobile IPv4

In the following sections, we will provide a survey of the two most important mobility protocols, designed to support host mobility since the classical versions of IP either do not support it at all (v4) or need a protocol for better operating (v6). We first begin with a description of mobile IPv4 [39].

There is a growing need for mobility; computer users wish to connect to the Internet and maintain this connection even if they move from place to place. However, IPv4 was designed bearing in mind that hosts are always connected to a network at a physical fixed location. An IPv4 user who wishes to connect from a point different from the one usually used, would have to execute a slow process of parameters configuration to connect, before all retrieving a new IP address.

Mobile IP, a standard proposed by IETF, was designed to solve this problem by allowing the MN to utilize two IP addresses: a fixed home address and a care-of address that changes at each new point of attachment. This protocol assumes that a network that allows users mobility must create a home agent, and every network that allows the presence of “visitors” must create a foreign agent.

When an MN attaches to a foreign network, it has to register at the foreign agent. If the registration was successful and the MN authenticated, the latter receives a *Care of address*, which is the MN’s new IP address in the foreign network. The home agent must receive the MN’s new IP address to be able to forward packets destined to the MN correctly. To perform these operations, mobile IP works in three phases:

Agent discovery

Agent Discovery is the method by which an MN determines whether it is currently connected to its home network or to a foreign network, and by which it can detect when it has moved from one network to another [39]. Either foreign or home agents generally perform this task by periodically broadcasting agent advertisement messages. If an MN receives such a message, it recognizes whether it is in its home network or not. However, if the MN wants to get a care of address, it may broadcast a solicitation message in order to force the foreign agent to send the care of address.

Registration

The registration function permits to create a mapping between the care of address and the home address. There exist two kinds of registration messages, request and reply. The mobile sends the first to inform its home agent of the actual care of address; the second contains the answer of the home agent to the MN's message. In this process, the foreign agent is only a passive entity, it may or more commonly may not participate in it.

Tunnelling

The tunnelling function allows the home agent to redirect packets addressed to the MN to its new care of address, by using a "tunnel" where original IP packets are encapsulated within another IP packet. For this reason, this encapsulation process (Figure 7) is called "IP within IP".

.

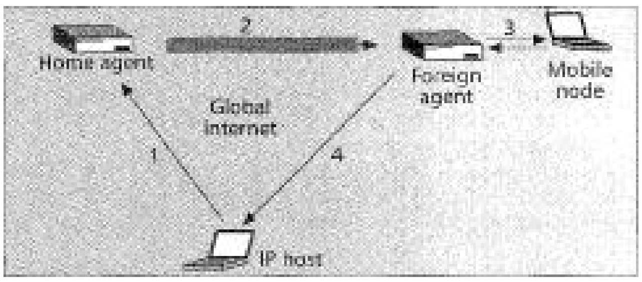


Figure 7: The encapsulation process in mobile IPv4

The packet flow is the following: an IP host sends one or more packets to the MN's home address (1); the home agent takes care of these packets and tunnels them, encapsulated within another IP packet, to the MN's care of address (2). The foreign agent collects the packets and delivers them to the MN (3); it also takes care of routing MN's answers (4). To optimise the routing, it is also possible that the foreign agent communicates the MN's care of address to the

IP host so that afterwards the latter can send packets directly to the MN's care of address, avoiding this triangulation. However, this feature is optional in Mobile IPv4 and may not be supported. Other solutions for route optimisation are presented in [41].

The encapsulation described above is attained at the home agent by adding to the original packet a new IP header that contains the care of address; the foreign agent will remove the outer header and forward the original packet to the MN. The following figure shows the mechanism of header encapsulation:

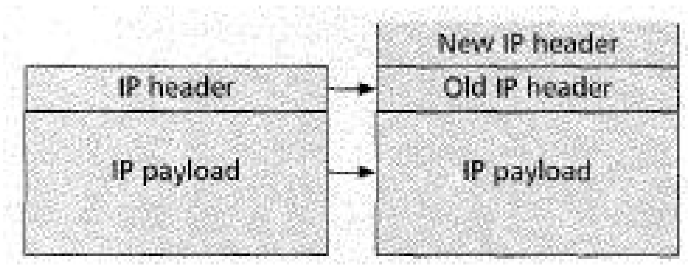


Figure 8: IP within IP encapsulation

Note that other types of encapsulation are possible too, but the default algorithm that must always be supported is IP within IP (described in [40]).

3.4 Mobile IPv6

Mobile IPv6 [42] is the protocol that enables IPv6 users to mobility. Although it makes use of the basic idea presented by mobile IPv4, and thus that an MN retrieves a care of address when it moves into a foreign network, while its home agent takes care to forward packets addressed to the MN, mobile IPv6 benefits of the improvements carried out by IPv6 [43].

These improvements deal mostly with the mobility support offered by the IPv6 extension header by using the destination options. Only the recipient processes these options so that intermediate routers do not worsen their performance. The problem of triangle routing, which in mobile IPv4 could only be addressed as an option, here is a full functionality of the protocol that provides the creation of a binding cache for this purpose, where all the associations between home address and care of address are mapped.

The authentication header of IPv6 is a good help in traffic authentication, a primary security concern in this mobility scenario since a malicious node could hijack the traffic directed towards a mobile node to itself; this functionality is not present in the IPv4 header.

3.4.1 Operation of a mobile node in IPv6

When a mobile node realizes it is in a foreign network, it tries to retrieve a care of address, which can be considered its IP address when attached to the foreign network. The way by which the mobile node can obtain the care of address can be either stateless (by means of stateless address autoconfiguration [44]) or stateful; these techniques ensure the mobile node a higher capability (compared to mobile IPv4) to obtain its care of address, thus reducing the need for foreign agents. This operation does not forbid the mobile node to hold previously acquired care of addresses, in order to continue to receive packets sent to these old addresses.

The mapping between the new acquired care of address (primary) and the home address is called *binding;* after the registration, the mobile node sends a binding update destination option to its home agent and to all of its correspondent nodes, in order to update their binding cache and optimize routing. To perform this task, the mobile node must keep a binding update list, in which are maintained the addresses of all of the nodes to which the binding update must be sent, comprising the temporal validity of the entry.

Adding a node to this list is easy: since the route optimization is always performed in order to avoid the problem of triangle routing, if the MN receives an encapsulated tunneled packet from the home agent, this means that the correspondent node does not know its care of address and so it must be added to the binding update list.

To make sure that both home agents and correspondent node receive the binding update, the MN must receive a binding acknowledgment. To force them to send the ack, the MN must set the acknowledgment bit in the binding update message. Unless it does not receive the binding acknowledgement, the mobile node keeps on periodically retransmitting the binding updates; this task is compulsory only for binding updates sent to a home agent, optional for those transmitted to correspondent nodes, in fact an MN can realize that correspondent nodes have not received the binding update simply if it gets tunneled packets from them. Binding updates or binding acknowledgements, being Ipv6 options, can be sent either included within an IPv6 packet carrying a payload such as TCP, or as a separate IPv6 packet containing only options and no upper layers payload.

Figure 9 below explains this situation. Note that when the binding options are not piggybacked within a payload, the next header field must indicate “no next header”; the authentication header, present in both cases, shows the improved security level provided by IPv6, that is one of the advantages that make mobile IPv6 superior to its v4 counterpart.

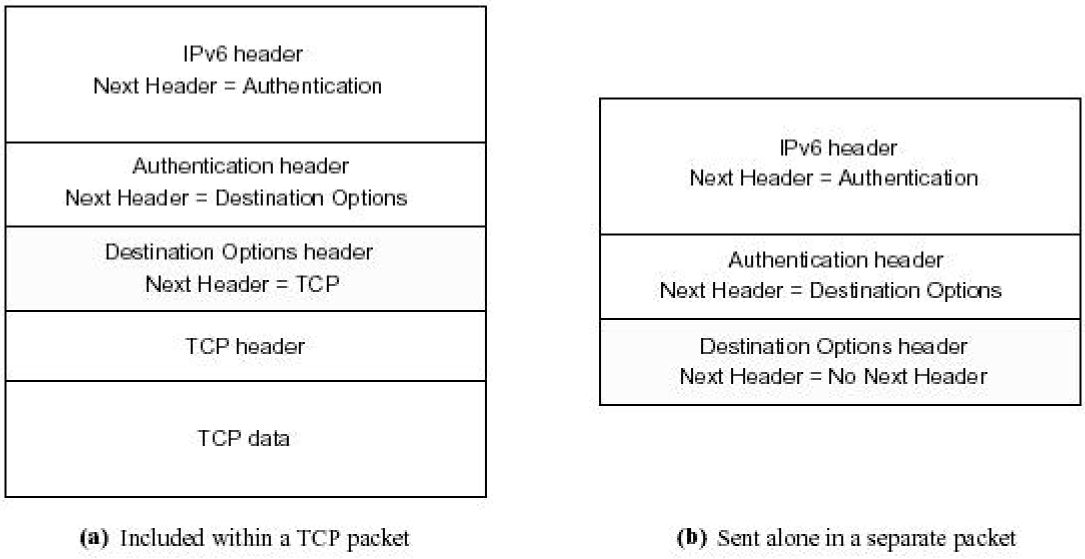

IPv6 header
Next Header = Authentication
Authentication header
Next Header = Destination Options
Destination Options header
Next Header = TCP
TCP header
TCP data
(a) Included within a TCP packet
IPv6 header
Next Header = Authentication
Authentication header
Next Header = Destination Options
Destination Options header
Next Header = No Next Header
(b) Sent alone in a separate packet

Figure 9: Binding updates and acknowledgements as IPv6 destination options

Correspondent node can send the MN a binding request in order to receive a binding update and refresh their binding cache before an entry expires.

In case an MN does not know the IP address of its home agent, mobile IPv6 provides a mechanism that permits a MN to dynamically retrieve it, choosing it from a list of suitable home agents candidates. To perform this task, the MN must send a binding update to the home agent anycast address [45] (for its home subnet); this message will be processed by one of the possibly many routers acting as home agent in the MN's home network. The anycast addressed home agent will reject the registration request but will add to the message a list of home agents (called home agent list) sorted in such a way that home agents are placed in order of decreasing preference value.

The home agents, through the periodic exchange of multicast unsolicited router advertisements, update this list and keep it coherent with the real network conditions. When the MN receives this list, it sends a binding update to one of the home agents in the list. If the registration request is rejected, the MN has to choose another entry in the list. Note that the choosing of a home agent should have to be performed respecting the order in the list, since routers are sorted in a descending manner of suitability.

3.4.2 Advantages of mobile IPv6

Now that the main features of mobile IPv6 have been described, we want to present a brief list of the advantages brought by this version of protocol, in relation to Mobile IPv4 drawbacks:

- Mobile IP (both versions) assigns different IP addresses to an MN each time it changes its point of attachment to the network; since the lack of addresses is becoming evident in IPv4, introducing IPv6, many more addresses will be available.
- The use of anycast addresses is enabled in Ipv6, and allows the deployment of a mechanism to dynamically discover the home agent's address, returning a single reply to the MN. IPv4 does not provide this mechanism, but broadcasts a separate reply to the MN from each home agent in the MN's home link. The Mobile IPv6 mechanism is more efficient and more reliable, since only one packet need be sent back to the MN.
- Support for Route Optimisation is now built in as a fundamental part of the protocol, rather than being added on as an optional set of extensions that may not be supported by all nodes as in Mobile IPv4. This functionality allows, as noted above, to avoid the grave problem of triangle routing.
- The use of IPv6 destination options allows all Mobile IPv6 control traffic to be piggybacked on any existing IPv6 packets, whereas in Mobile IPv4 separate UDP packets were required for each control message.
- Mobile IPv6 eliminates the need for deploying foreign agents, as done in mobile IPv4, since it makes use of IPv6 features such as Neighbour Discovery [46] (the protocol that replaces ARP [47], used in IPv4) and Stateless Address Autoconfiguration [44] and allows an MN to operate in a foreign network without any special support supplied by a local router.
- The fact that mobile IPv6 almost eliminates the need for encapsulation decreases header overheads, thus improving the throughput of the system.
- Mobile Ipv6 uses all security improvements introduced by IPv6.

4 Current and future mobile networks

This chapter presents an overview of GSM, the current second generation cellular mobile network and of its architecture enabled to provide data services, the GPRS. It concludes with a description of the third generation network for mobile communications, the UMTS. The last sections explain how IP QoS can be attained in the described architectures.

4.1 Overview of the Global System for Mobile communications

During the early 1980s, analog cellular telephone systems experienced a very rapid growth in many countries in Europe. Each nation developed its own system, which was incompatible with everyone else's in equipment and operation, both in hardware and in software. The mobile terminal was restricted to operate within national boundaries. This was an undesirable situation since the market for mobile equipments was limited to the single nation and economies of a larger scale (with the subsequent money savings) could not be realized.

In order to overcome these problems, the Conference of European Posts and Telecommunications (CEPT) formed, in 1982, the Groupe Spécial Mobile (GSM) [48] to develop a pan-European mobile cellular radio system (the GSM acronym later became Global System for Mobile communications). The standardized system had to meet certain criteria:

- Good subjective speech quality
- Low terminal and service cost
- Support for international roaming
- Ability to support handheld terminals
- Support for range of new services and facilities
- Spectral efficiency
- ISDN compatibility

The GSM system was developed using a digital technology, since it seemed the one that offered better guarantees in order to meet the above requirements.

4.1.1 Network architecture

The GSM network is composed of several functional entities, which can be grouped in three main parts [49], as Figure 10 shows:

- The Mobile Station (MS).
- The Base Station Subsystem (BSS).
- The Network and Switching Subsystem (NSS).

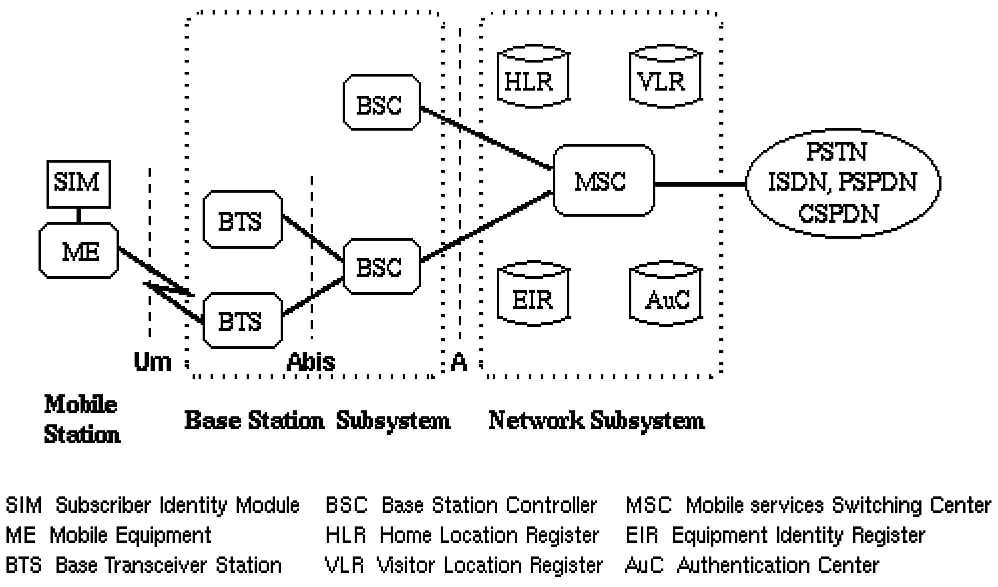

SIM
ME
BTS
BTS
BSC
BSC
HLR
VLR
MSC
EIR
AuC
PSTN ISDN, PSPDN CSPDN
Um
Abis
A
Mobile Station
Base Station Subsystem
Network Subsystem
SIM Subscriber Identity Module
ME Mobile Equipment
BTS Base Transceiver Station
BSC Base Station Controller
HLR Home Location Register
VLR Visitor Location Register
MSC Mobile services Switching Center
EIR Equipment Identity Register
AuC Authentication Center

Figure 10: Architecture of the GSM network

The mobile station (MS) represents the device with which a user can receive the services offered by GSM. It consists of the mobile equipment (the terminal) and a smart card called the Subscriber Identity Module (SIM). The SIM allows the personal mobility, so that a user can receive the subscribed service regardless the specific mobile equipment used. The mobile equipment is uniquely identified by the International Mobile Equipment Identity (IMEI), whereas the SIM is identified by the International Mobile Subscriber Identity (IMSI). The IMEI and the IMSI are independent, thereby ensuring personal mobility. The SIM card is protected by a four-digit Personal Identific ation Number (PIN) against unauthorized use.

4.1.2 The Base Station subsystem

The BSS subsystem takes care of managing the radio part of the system, and thus comprises the functional entities that provide radio coverage of an area constituted by one or more cells. The base station is constituted by two units: a Base Transceiver Station (BTS) and a Base Station Controller (BSC); the communication interface between these two entities, A-bis, is standardised in order to not be bounded to proprietary solutions and utilize components produced by different suppliers.

The BTS corresponds to the transceivers and antennas used in each cell of the network and is usually placed in the centre of a cell. Its transmitting power defines the size of a cell. Each BTS has between one and sixteen transceivers depending on the density of users in the cell. The BTS handles the radio communication with mobile equipments performing several tasks, among others:

- Frequency hopping
- Discontinuous transmission
- Dynamic power control
- Antenna diversity
- Radio connection monitoring

The Base Station Controller manages the radio resources for one or more BTSs, handling channel set-up, frequency hopping, internal handover, control of the radio frequency power levels of the BTSs and so forth. In a wide urban area there are a lot of BTS controlled by few BSC.

4.1.3 The network subsystem

The main role of the network subsystem is to manage the communications between the mobile users and other users; it also includes the data-bases needed to store information about the subscribers and to manage their mobility. The central component of the Network Subsystem is the Mobile services Switching Center (MSC). An MSC controls all the BSC of its pertaining zone, and must serve the mobile equipments passing through its control area, it sets-up, releases and bills calls to and from mobile equipments in the geographical area served. Moreover, it manages mobility and forwards calls in cooperation with the other entities of NSS. It provides connection with fixed networks *Public State Telephone Network (PSTN), Integrated Services*

Digital Network (ISDN), Packet Switched Public Data Network (PSPDN), and *Circuit Switched Public Data Network (CSPDN).*

In order to manage user mobility, a MSC must exchange information with a database, the Visitor Location Register (VLR) in which are memorised, temporarily, the data concerned with mobile equipments that are in that area (i.e IMEI, authentication parameters). Each provider holds a central database (Home Location Register) where data related to the users is permanently memorised, both the static (such as subscription details) and the dynamic data (it can vary because of actions of the users).

The HLR is simply a database, so it does not generate the data it is holding. The task to calculate these parameters is committed to a functional entity called Authentication Center (AuC). The Equipment Identity Register (EIR) is a register containing information about the mobile equipments. More particularly, it contains a list of all valid terminals, forbidding to effect calls from stolen or unauthorized terminals.

Another subsystem to mention is the Operation and Support Subsystem (OSS). It is connected to the different components of the NSS and to the BSC, in order to control and monitor the GSM system. It is also in charge of controlling the traffic load of the BSS. However, the increasing number of base stations, due to the development of cellular radio networks, has provoked that some of the maintenance tasks are transferred to the BTSs. This transfer decreases considerably the maintenance costs of the system.

4.2 The General Packet Radio Service

The growth of GSM, in the last few years, has been more and more significant, and the number of users raised exponentially. However, the main field of utilisation of GSM has been mobile telephony, and only a small part of the overall traffic was constituted by data. The most known data service in GSM is without doubt the Short Message Service; other applications seem to have little success, one of the reasons might be the fact that the data transmission rate in GSM is relatively low, 9.6 kbit/s.

The main reason for the development of GPRS is to achieve data services with greater bandwidth efficiency. A key concept of GSM architecture has been changed. Thus, GSM provides a circuit switched approach, not suitable with data services. For this reason, GPRS is designed as a packet switched architecture, which best addresses the bursty nature of data traffic. This philosophy adds another advantage from the user point of view: it is more cost efficient than a switched circuit one, since the user must not pay the time spent in the connection, as happens in GSM, but only the exact amount of data exchanged.

From the technical point of view, the drawback results from the fact that in circuit switched radio transmission, at the air interface, a complete traffic channel is allocated for a single user for the entire call period. In case of bursty traffic (e.g., Internet traffic), this results in highly inefficient resources utilization. It is obvious that for bursty traffic, packet switched bearer services result in a much better utilization of the traffic channels. This is because a channel will only be allocated when needed and will be released immediately after the transmission of the packets.

With this principle, multiple users can share one physical channel (statistical multiplexing). This billing method allows the user to maintain a permanent connection to the network, without suffering any charge, and to carry out a conversation while downloading data. GSM does not permit this, since you can only have either a data connection or a voice connection at a time.

4.2.1 System architecture

The GPRS architecture is quite similar to its GSM counterpart, but integrates some entities able to support the additional functionalities introduced by GPRS. A new class of network nodes, called GPRS support nodes (GSN), is present in the GPRS architecture [49]: the task of GSNs is to route data packets between mobile stations and external Packet Data Networks (PDN). Figure 11 below illustrates the system architecture:

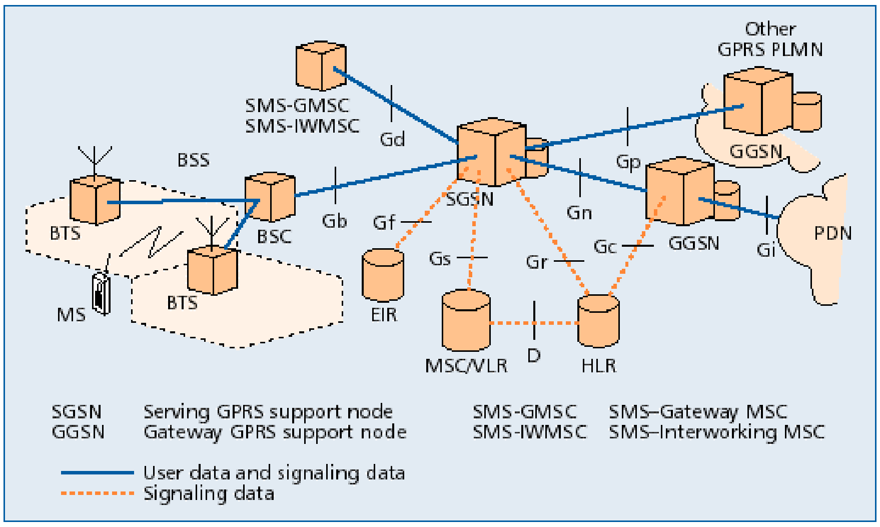


Figure 11: The GPRS system architecture

A serving GPRS support node (SGSN) is responsible for the delivery of data packets from and to the mobile stations within its service area. It also has to solve topics related to mobility management, such as attach/detach of a mobile and location management, authentication and charging functions. The location register of the SGSN stores location information and user of all GPRS users registered with this SGSN.

A gateway GPRS support node (GGSN) can be considered an interface between the GPRS backbone network and the external PDNs. It converts the format of packets outcoming the GPRS network into the appropriate packet data protocol (PDP) for the PDN (an example of PDP could be IP) and in the other direction converts incoming PDP packets into GSM packets and sends them to the responsible SGSN. To perform this task, the GGSN keeps in memory the current SGSN address and profile of the user in its location register. Note that a GGSN can serve more SGSNs, whereas a SGSN can forward its packets towards different GGSNs in order to reach different PDNs. In Figure 11 are also shown the interfaces between the new network nodes and the GSM network as defined in [49].

The dialogue between a BSC and SGSN is warranted by the Gb interface; instead, Gn and Gp interfaces serve for signalling and data transmitting between GSNs. The former interface is used if SGSN and GGSN are located in the same PLMN (public land mobile network); the latter is utilised in the other case, if they are in different PLMNs. The Gi interface connects a GGSN to an external PDN (possibly Internet). The Gd interface is defined to exchange messages of the short message service (SMS) via GPRS. It interconnects the SMS gateway MSC (SMS-GMSC) with the SGSN. The functions of the HLR and VLR are essentially the same respecting GSM, with some additional feature added in order to address GPRS functionalities, for example the handling of switched packets.

4.2.2 Mobility management

Before a mobile station can use the services provided by GPRS, it must be registered; the registration procedure, called GPRS attach, is achieved by storing the user profile (retrieved from the HLR) into the SGSN and assigning to the user a packet temporary mobile subscriber identity (P-TMSI). The disconnection from the GPRS network is called GPRS detach. It can be initiated by the mobile station or by the network.

To communicate with external PDNs, an attached mobile must receive an address suitable for that PDN (for example an IP address): this address is the PDP address (Packet Data Protocol address) and describes the characteristics of the session initiated by the mobile. It contains the PDP type (e.g. IPv4), the value of the address, the QoS specifications and the address of the GGSN acting as access point to the PDN. This information is called context and it is not unique since a mobile can hold more than one context for each session.

The allocation of the PDP address can be static or dynamic. In the former case, the network provider permanently assigns the address to the user. In the latter case, it is assigned contextually to the activation of a PDP context: this is the case of the request for an IPv4 address. Due to the lack of IPv4 addresses, it is obvious that it must be assigned dynamically (for example via DHCP).

Another problem that must be addressed is that of location management. It refers to individuate the user's position in order to deliver the packets addressed to the mobile terminal correctly at any instant. The mobile station sends update messages to its current SGSN at regular intervals of time: for this reason, the problem of setting the time of updating is very important. If it is too short, i.e. the mobile often sends update messages to its SGSN, this means that the mobile position is well known by the network, and no delay is added in delivering packets correctly, but the high number of messages exchanged could waste precious wireless resources and cause consumption of power, shortening battery lives. On the other hand, if the update messages are sent seldom, the location of the mobile is not known exactly by the network and paging is necessary for each downlink packet causing a significant delivery delay. Thus, a trade-off should be realized in order to optimise location management.

For this reason, a state model, shown in Figure 12, has been defined for handling location management in GPRS [50]:

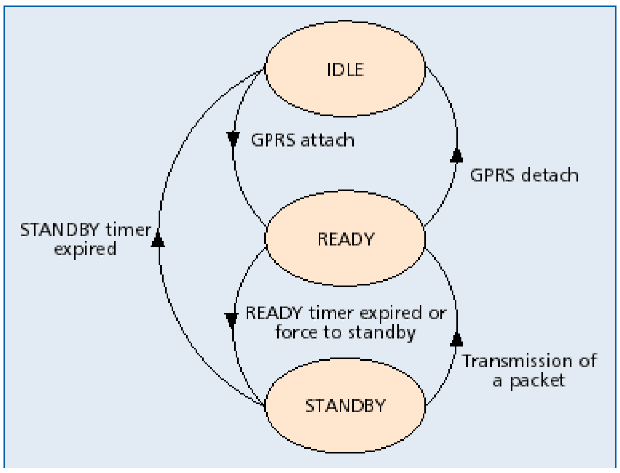


Figure 12: State model of a GPRS mobile station

A MS can be in one of three states depending on its current traffic amount; the location update frequency is dependent on the state of the MS. In the IDLE state, the MS is not reachable and no location updating is performed, i.e., the current location of the MS is unknown to the network. After performing a GPRS attach, the MS gets into the READY state. After a detach operation, the MS comes back to the IDLE state and PDP contexts are deleted. The STANDBY state will be reached when an MS does not send any packets for a long period (the time of expiration of the READY timer). The timer is refreshed when the mobile sends a packet, causing the return to the READY state. An MS in READY state informs its SGSN of every movement to a new cell. If an MS in the STANDBY state keeps on not transmitting packets, the STANDBY timer will also expire and the MS will return to the IDLE state.

4.3 The Universal Mobile Telecommunication System

The Universal Mobile Telecommunications System (UMTS) [51] is the new third generation standard being developed in order to improve GSM performance. UMTS is called third generation since it offers a wide set of services to users regardless of their location in the world; these services can be digitised voice, video and multimedia at data rates up to 2 megabits per second. The dream of video phones could be finally realised with UMTS! A UMTS user will be able to access to the service wherever he is, whenever he wants, since “universal” coverage is ensured by the deployment of satellite links.

UMTS is universal because it will operate in all environments and all countries; users will receive specific services even when they are in a nation different from the one where they have subscribed to the service; in this case, a different network operator will have to provide all the specific services the user has subscribed. Personal, terminal and service mobility will be achieved everywhere; circuit and packet switched services will be supported.

The fundamental way to achieve the promised broadband is a new mechanism of accessing to the radio medium. Whereas GSM uses a TDMA/FDMA technique where the frequency spectrum is subdivided into slots (either temporal or frequency slots), UMTS utilizes Code Division Multiple Access (CDMA). CDMA is a modulation and multiple-access scheme based on spread-spectrum communication. In this scheme, multiple users share the same frequency band at the same time, by spreading the spectrum of their transmitted signals, so that each user's signal is pseudo-orthogonal to the signals of the other users.

4.3.1 Cell structure

The cell structure in UMTS is planned as being a hierarchical one, using pico, micro and macro cells. The macro cells provide wide area coverage and are used for high-speed mobiles. The micro cells are used for outdoor coverage to supply extra capacity where macro cells could not be sufficient, and should placed every 200-300 meters. The pico cells would be deployed mainly indoors, in areas where there is need for high speed data services such as videoconferencing and may be of the order of 75 meters in distance.

When the UMTS is affirmed to have a 2 megabit/second bit rate, it is intended that is the maximum bit rate available with such a technology, achievable only in pico cells (those able to provide the major bandwidth) for a user almost standing or moving very slowly. Anyway, for fast users (such as those moving in a car) a bit rate not lower than 144 kbit/second should be ensured. Figure 13 shows the cell structure in UMTS; the world cell represents the fact that thanks to satellite coverage, a user can make use of UMTS service even in wide areas, such as oceans or deserts. However, this has not been implemented in practice yet.

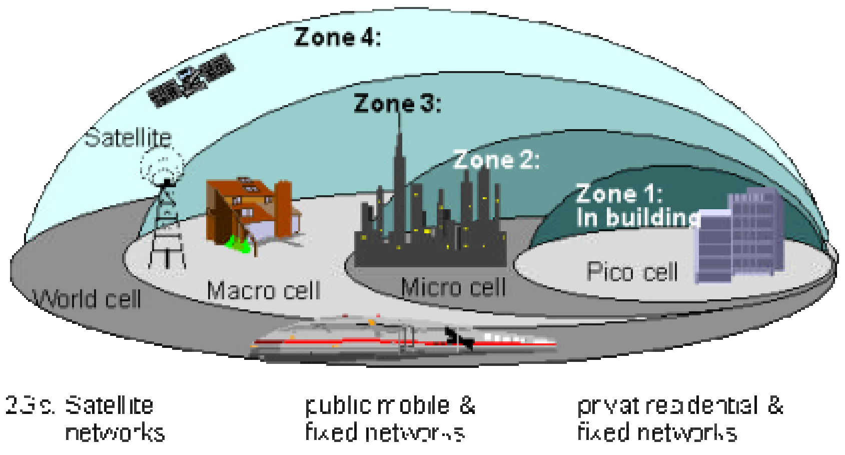


Figure 13: Cell structure in UMTS

4.3.2 Network architecture

The basic UMTS network architecture [52] is divided into three parts:

- User equipment
- UMTS Terrestrial Radio Access Network (UTRAN)
- Core network

Figure 14 shows this simplified UMTS architecture with its basic domains and external reference points and interfaces to the UTRAN:

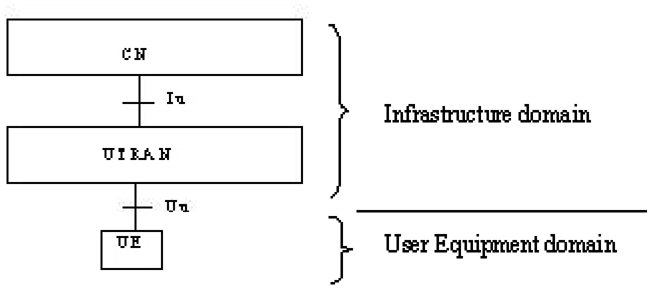


Figure 14: Simplified UMTS network architecture

The user equipment is the terminal used to access UMTS services (both data services and voice services); its interface to the UTRAN is called Uu. The terminal is referred to as belonging to the User Equipment Domain, intending to express with this that the user resides in a lower hierarchical level than the network, since it simply “utilizes” services provided by UMTS. The Core network and the UTRAN form the infrastructure domain of the UMTS, since they are the entities designed to supply the service. All the authorised users within the coverage area of this infrastructure share the resources provided. The UTRAN is connected to the Core Network through the Iu interface.

The main idea behind the design of the UMTS terminal is a backward compatibility with the GSM ones (this idea, however, is applied to large parts of the UMTS system), since for long time GSM and UMTS will co-exist, since it is impossible to suddenly change from second to third generation. For example, UMTS provides solutions for performing handovers from a GSM to an UMTS zone and vice versa. According to this idea, the terminal is divided in two functional entities, just like GSM, the mobile equipment and the U-SIM (UMTS SIM). The mobile equipment is the device that actually performs the connection to the network and the access to the services. The U-SIM allows personal mobility and comprises many features that enhance those provided by its GSM counterpart (i.e. greater memory capacity, better performances, contact less operations).

The UTRAN [53] is formed by more Radio Network Subsystems (RNS), which are the access part to the UMTS network (it can be considered equivalent to the BSS of GSM); a RNS performs the allocation of radio resources when a connection between user equipment and the UTRAN is being established and its release when the connection itself is torn down. Each RNS then, consists of a Radio Network Controller (RNC, analogue to GSM’s BSC) and a set of the so-called Node B (BTS in GSM). A Node B, which may be connected only to one RNC, has the tasks of radio transmission and reception in one or more cells to or from the user equipment; the RNC has the overall control of the logical resources of its Node Bs and the responsibility for the handover decisions that require signalling to the user equipment. In Figure 15 the UTRAN architecture and the interfaces between the entities is drawn:

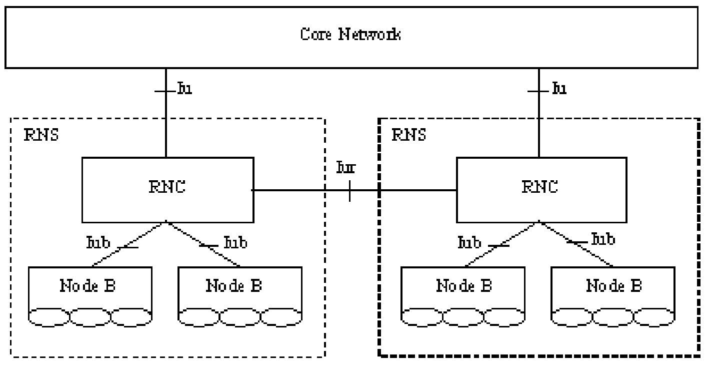


Figure 15: UTRAN architecture

The RNS employed in a connection between User Equipment and the UTRAN, is deemed as the serving RNS, since it must manage the radio connection and perform all the related tasks. If the resources handled by the serving RNS seem not to be sufficient, the so-called drift RNSs can be added in order to provide more resources to the connection.

The Core Network is formed by the physical entities that provide support for network services, such as management of user location information (the same as HLR,VLR in GSM) and signalling mechanisms. The Core Network is further sub-divided into the Serving Network Domain, Home Network Domain and Transit Network Domain, as shown in Figure 16:

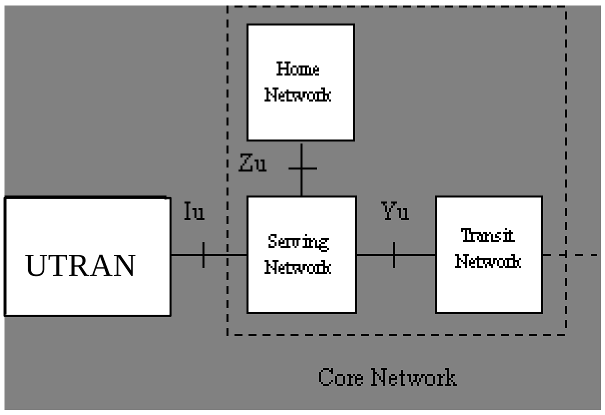


Figure 16: UMTS Core Network

The serving network domain is directly connected with the UTRAN (via the Iu interface). It carries out functions that are local to the user and thus subjected to variation due to user movement, i.e. routing functions. The home network domain performs its tasks regardless of the user position. The U-SIM is related to this domain, so that the home network domain is responsible for managing information relative to the user (identity, subscribed services, etc.). The transit network domain lies between the serving network and the remote network, not necessarily a UMTS network.

4.4 IP Quality of Service in GSM

This and the following sections address the problem of providing IP QoS in the mobile cellular networks described above. We first begin with the GSM network, continue with GPRS and finish our discussion dealing with IP QoS aspects in UMTS.

It is quite difficult to define the concept of IP QoS in a GSM network, since it is essentially a circuit-switched network and thus does not support IP traffic very well, IP traffic being packet-switched oriented by definition. The global GSM architecture is thought to provide circuit switched services; the clearest example is in the billing policy, customers pay for the time they are connected, even if they are doing nothing during connection time. Such an approach is clearly not feasible in order to provide IP services, since the resources are actually used only when the packets are transmitted; we will present the approach that [54] has proposed to achieve an integration of GSM and mobile IP.

In this proposed version of GSM and Mobile IP (called GSM.IP), the role of the home agent and of the foreign agent is performed by the MSC, by means of GSM's HLR and VLR. The combination of these entities is called MSC-HA and MSC-FA. The care of address is always held by the MSC-FA (instead of being assigned dynamically). The resulting addressing structure is very similar to its IP counterpart and allows an optimisation of routing schemes. The MSC has its own IP primary class address, the addresses of BSCs and BSs pertaining to that MSC are extensions of the MSC address; the care of address for the MN, finally, is an extension of the BS address. Figure 17 clarifies the addressing mechanism:

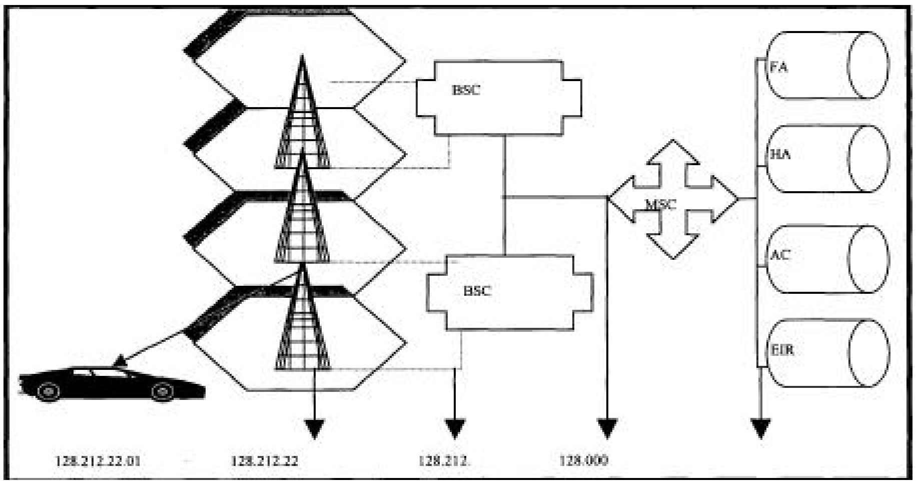


Figure 17: The GSM.IP architecture [54]

The operations performed in the GSM.IP architecture recall their mobile IP counterparts. The MSC-HA takes care of forwarding packets addressed to the home MN address; when the mobile is in a foreign network, it receives packets via the MSN-FA. Packets travelling between MSC-HA and MSC-FA are encapsulated as well as in mobile IP; MSC-FA sends packets to the BSC actually serving the MN after removing the outer header.

This approach seems attractive, but needs enhancements in two directions. First, it must implement mobile IPv6 operations rather than IPv4, since it is expected that Mobile IPv6 will receive a major deployment in the near future compared to Mobile IPv4; second, GSM should interwork with TCP as well in order to realize a full GSM.TCP/IP architecture, which can provide IP services better than a simple GSM.IP one.

4.5 IP Quality of Service in GPRS

The packet switched approach presented by GPRS makes this technology more feasible than GSM to provide IP QoS in a cellular system. The major drawback to GPRS, is that packet switched networks do not support real time applications very well, since there is no guarantee that timing constraints are met. From this point of view, the circuit switched GSM network seems more adaptable to handle real-time applications, because of the set-up of virtual circuits before initiating the connection and the consequent warranty that resources are actually reserved for the demanding user.

Anyway, even if we consider the improvements that new circuit switched techniques, such as HSCSD [55] have carried out, the bit rate available with GSM is too low to sustain real time applications. The GPRS provides data services at a rate considerably higher than GSM, but has two main limitations:

- It differentiates QoS only on the basis of the IP address of a mobile station, but not on the basis of IP flows, i.e. only one PDP context can be supported for each IP address.
- The GPRS core network uses IP tunnels, which makes the applicability of IP schemes difficult.

To overcome these problems, [56] proposes two solutions based on the IntServ approach and on the DiffServ one. The first solution obviously requires the establishment of reservations across the GPRS core; here the problems related to the tunnelling arise. The communication between SGSN and GGSN is exploited with tunnelled packets, and the first router that examines the IP header is the GGSN. This means that when RSVP reservations travel through the GPRS backbone network towards a GGSN or vice versa, no resource reservation is accomplished.

The same problem is also present when the DiffServ approach is deployed. In this case, in fact, the service is set by means of the DS field, which cannot be examined being part of the IP header that is encapsulated in the tunnelled packet. To address this problem, the proposed solution is to let original the RSVP messages pass transparently through the GPRS core network, but at the same time to generate additional RSVP signalling messages to create and maintain RSVP reservation at the core network routers. The GGSN is the coordinating entity, which performs all these reservations, both for the traffic originated by the mobile station and for the traffic to the mobile station. By means of simulations, [56] shows that this approach does not suffer of scalability problems typical of IntServ approaches; moreover, the only GPRS functional entity that needs a modification is the GGSN, the process is in fact transparent to the other entities.

Another structure that allows providing IP QoS across a GPRS network makes use of DiffServ concepts, and thus indicates which classes of services must be employed. A specific QoS treatment will be guaranteed within each class in the, say, GPRS access network (MS-SGSN-GGSN) while the GGSN must take care of assigning the correct PHB to each class before that the traffic is forwarded from the backbone network. To support this DiffServ scheme, the mobile station is required to be multihomed, capable of maintaining more IP addresses at the same time; each IP address, in fact, is mapped to a particular PHB, according to the mapping present in GGSN memory. For uplink traffic, the mobile station must be able to forward packets using the appropriate IP address (depending on the QoS required, i.e. on the class of service). The SGSN must assign traffic to the specified PHB setting a value in the outer header of the tunnelled packet.

It is simple to note that such an architecture requires that all of the entities involved in a GPRS network must be changed in order to support DiffServ and that this approach produces a higher resource consumption in network nodes than the IntServ one.

4.6 IP Quality of Service in UMTS

As stated above, second generation cellular systems essentially offer voice services and a small set of best effort data services, like SMS. The introduction of GPRS permits to overcome the problem of providing data services, but the nature of the offered service remains best effort yet. Third generation cellular systems must supply more than simple best effort services, since they must support real time applications, and in general those requiring strong timing constraints. These considerations make the imposition of QoS requirements in a 3G cellular network necessary.

There are several reasons to deploy IP transport in the radio access network; one among them is without doubt the fact that IP is independent from lower layers protocols, so that it is possible for network providers to choose different lower layer technologies under an IP network. Moreover, researchers are orientated to make IP the basis for the transmission of packetised voice, data and signalling and management information in the Internet world. Since UMTS must guarantee Internet access, everywhere, at all times, it becomes clear that IP based solutions are the most feasible for 3G networks among the other proposals.

However, the employment of IP transport presents the problems typical of IP, which must be improved to provide QoS support addressing delay, jitter and loss requirements. Furthermore, the large packet overhead due to real time applications (including RTP fields as well as TCP/IP ones) may constitute a factor that worsens the throughput significantly. A way to address this problem is performing header compression, which in conjunction with the proposed QoS architectures can help in providing the service to requiring applications.

The IP transport is a reasonable alternative transport mechanism for the last mile; however, we must also consider the rest of the UTRAN, paying particular attention to the IP backbone interconnecting mobile network elements. The paper [57] illustrates the mechanisms needed to supply QoS in an IP-backbone for the UTRAN. Figure 18 shows a scheme of principle of an IP-based UTRAN:

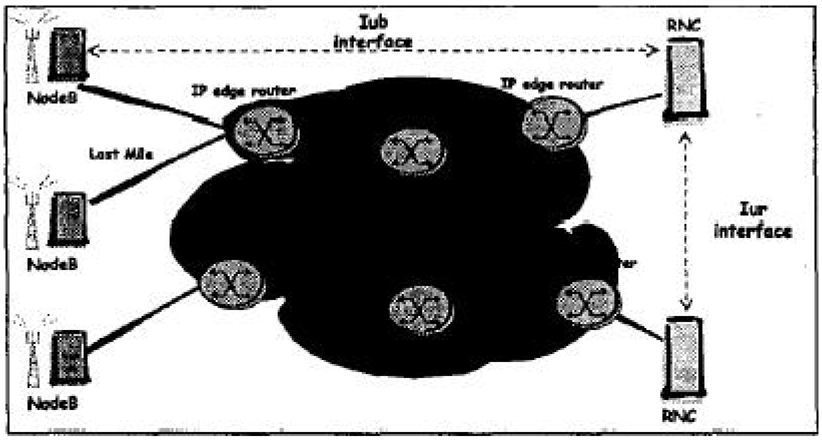


Figure 18: IP-based UTRAN [57]

In the architecture we are discussing, IP-routes interconnect RNCs and NodeBs; the introduction of such an architecture impacts the following elements:

- RNCs and NodeBs themselves, since they have to implement IP in their protocol stacks
- The last mile topology, the one that, as Figure 18 shows, interconnects a NodeB and an IP edge router
- The IP-access backbone, necessary to link mobile network elements

The main issues that rise when we talk about the last mile essentially consist of the scarcity of bandwidth of the radio link; reducing IP overhead is crucial in order to address properly the problem. The authors of the paper think to achieve this purpose by reducing the relevant IP overhead using header compression techniques or at least to reduce overhead by multiplexing several payloads into the same IP-packet. Anyway, the last method presents the drawbacks common to large size packets: large transmission time and waste of bandwidth in case of retransmission increased delay. There must be a trade-off between low throughputs granted by small packets and low efficiency presented by bigger ones. To meet these constraints, [57] proposes to treat both voice and data packets as real time traffic, ensuring that RNCs and NodeBs comply to the most stringent delay requirements (i.e. voice), thus foreseeing a single real time traffic flow.

Transmission of the mobile traffic over the IP backbone, dispels its bandwidth resources; to ensure a sufficient grade of service to requiring flows, different techniques can be deployed, such as overprovisioning, an IntServ IP-backbone or a DiffServ UTRAN. Among these proposals, the most suitable one seems to be a DiffServ capable IP-backbone; this solution holds all the advantages of a DiffServ approach, reducing the need of resource overallocation and well exploiting the effect of statistical multiplexing.

The model envisages to aggregate all of the traffic channels between an RNC and a Node B in a single, statically provided, stream, provoking that all packets are treated as real-time packets; this also means that this stream will be mapped onto an Expedited Forwarding (EF) traffic class. In their model, the authors state that the reservation is needed in upstream direction, since the path from the Node B area to the RNC through the IP backbone comprises a number of intermediate IP routers. In the opposite direction (downstream), it is task only of the edge router on the last mile to exploit these functions.

The traffic is divided into two categories: real time and best effort traffic, i.e. high priority and low priority traffic, respectively. This traffic aggregation makes a scheduling policy necessary, according to the DiffServ model, at the edge router and at intermediate IP routers; traffic streams of the same class are aggregated and served in agreement with the requirements of the class. Since the distinguished traffic classes are two, it is sufficient to have a simple scheduler with two queues, for real time traffic and for best effort; the two queues are served with a weighted fair queuing scheduling policy, where the real time traffic holds a higher weight, obviously, and best effort traffic can utilize the residual bandwidth.

5 Mechanisms to provide IP Quality of Service in radio access networks

The telecommunication world is moving towards an all-IP network, capable of supporting real time applications; therefore, it is not sufficient to study issues related to QoS, but efforts must be undertaken to conjugate IP and QoS in the same architecture. This chapter deals with the problems that mobility brings about the provision of QoS to demanding applications, explaining briefly some solutions proposed to overcome limits imposed by mobility. The final part of the chapter introduces the concept of the context transfer, which is the primary concern of this thesis.

5.1 Issues related to mobility

This section depicts the challenges that arise in designing a mobile network, with particular attention paid to obtaining IP QoS in such a network. Obtaining IP QoS presents the double problem of meeting general QoS requirements and adapting the IP protocol to a QoS context, since IP was not designed bearing in mind QoS constraints. In a fixed network, the main obstacle to QoS managing consists of the simultaneous request of resources performed by competing services, resulting in a situation of congestion. Mobility introduces other issues, such as the fact that the quality of the link is variable, up to complete loss and sudden recoveries. In any case, everyone must expect that bandwidth availability in a radio link is at least one order of magnitude lower than in fixed networks and the use of high rate applications is more problematic in mobile networks than in fixed ones.

A mobile terminal, moreover, is expected to be small and light; battery consumption is a problem that applications running on a fixed host must not deal with. Large screens, full size keyboards and user-friendly interfaces are replaced in mobile terminals by small, low resolution screens, sometimes even monochrome, often more suited to text than graphic. These limitations impact the QoS that users perceive.

As [58] indicates, issues in QoS assurances in mobile environments cover five layers of complexity:

1. Physical layer problems deal with the basic connection capabilities: obviously, if any of the transmission links limit QoS, it is clear that end to end QoS will suffer from this situation. Upper layers can solve only part of these problems.

2. Cell problems are produced by handovers. This is a major concern and we will discuss it later.
3. Network problems are due to changes in the IP network point of attachment (they happen because of a handover). The challenge is handling these issues in a real time manner.
4. Management problems are related to network management and configuration in order to ensure required QoS levels.
5. Administration problems consist of accounting and charging users for resource usage. This may seem not to be related with QoS, but handovers to different network provider cells can be refused by users, even if the requested QoS is assured, if the cost of the service is high. Sometimes it could be useful to reduce the QoS provided to keep the cost of the connection low and maintaining the connection with the customer.

Handovers can affect mobility in various manners and generate different amounts of signaling information. If the mobile terminal changes simply wireless cells, ideally this should not lead to AAAC (Authentication, Authorization, Accounting and Charging) traffic. If the mobile terminal changes autonomous domains (inside the network of the same wireless provider) this will require only a small amount of AAAC traffic; when the terminal moves into the network of a different provider, the problems discussed above, besides the exchange of AAAC information, dealing with the cost of services in the new network arise.

When QoS issues are considered in this discussion, the handover problem becomes much more complex; when a mobile terminal with a specified SLA moves into a new domain, it is likely that the new network cannot ensure the level provided by the previous one. Performing a handover in this case means seeking a network capable to sustain the SLA, or at least, renegotiating QoS parameters to modify the SLA.

Layer three handovers are problematic, since they require more time to be performed than layer two ones, since the redefinition of routing parameters and the checking of the capabilities of the new AR is necessary. It could be possible for example, either that the mobile node changes its IP address when attached to a new AR, or that it maintains the same address. In the former case, the exchange of information is necessary to provide a new routing treatment to the MN; in the latter case, it is necessary to modify routing tables since the same IP address now pertains to a different physical location.

Moreover, a layer three handover may only be processed after the conclusion of a layer two handover. This means that the overall delay will be the layer two plus the layer three handover delay. To minimize the overall delay, and thus minimize the drop or QoS degradation probability, [58] suggests to decouple a layer two handover from a layer three handover, by making every layer three router serve more cells and every cell served by more layer three routers.

5.2 Solutions to mobility related problems

The problems due to mobility in the management of QoS are several and different each other; for such a reason, the perfect solution does not exist. Proposals developed till now address this or that aspect of the problem, and trade-offs are often necessary to build an architecture that could be successful in the years to come. For example, protocols for supporting mobility and QoS have traveled until now on different planes separately. The RSVP protocol, for example, is not totally suitable in a mobile environment since it makes reservations along a fixed path.

If a user changes the point of attachment to the network, much signaling information is required to reestablish the service level along the new path; these operations take a lot of time to be performed, so that RSVP is not a suitable approach for real time applications in mobile environments. On the other hand, the two versions of Mobile IP have been designed with particular regard to mobility management, paying attention to the problem of forwarding packets to the new mobile node’s care of address, and little has been done to deal with the provision of QoS.

5.2.1 QoS enabled link layer architectures

Many researchers have directed their efforts towards exploiting QoS in link layer architectures and in wireless LANs. The paper [67] proposes a method to extend the MAC protocol for the IEEE 802.11 standard by distinguishing real time traffic from elastic traffic thanks to a priority scheduling approach; furthermore, traffic differentiation has been exploited also for elastic traffic. This way, high priority traffic receives more bandwidth than low priority traffic. Aad and Castelluccia [68] address the problem introducing three differentiation mechanisms to provide different levels of QoS in IEEE 802.11; these schemes are based on scaling the contention window according to the priority of each flow or user, on assigning different inter-frame spacing to different users and on using different maximum frame lengths for different users.

HiperLan 2 [69] is a short range mobile system under development by ETSI BRAN (Broadband Radio Access Networks). It is a new generation wireless LAN technology, supporting many features: it allows high-speed data transmission, up to a tenth of Mbps, ensuring support for QoS. Since HiperLan 2 is a wireless ATM system, it supports connection oriented services, suited for service differentiation. The main advantage of HiperLan 2 is thus that it can offer better (low latency) and differentiated QoS (guarantee of bandwidth) at the expense however of a big header overhead introduced by the architecture. A way to provide IP QoS in the HiperLan 2 architecture is presented in [71]. The authors propose functions that enhance the HiperLan 2 radio interface to support IP QoS, based on the system's ability to schedule the various connections dedicated for IP traffic according to their QoS requirements.

Bluetooth [70] is a global standard that eliminates wires and cables between stationary and mobile devices, facilitating both data and voice communication. The Bluetooth specification defines a short (around 10 m) or optionally a medium range (around 100 m) radio link capable of voice or data transmission up to a maximum capacity of 720 Kb/s per channel. Up to three simultaneous synchronous voice channels are used, or a channel that simultaneously supports asynchronous data and synchronous voice. Each voice channel supports a 64 kb/s synchronous (voice) channel in each direction. The asynchronous data channel can support maximal 723.2 kb/s. The requirements for QoS enhancements to the Bluetooth 1.0 specification and a Quality of Service Framework are presented in [79], which additionally gives explanations about the deficiency of the system to provide QoS.

5.2.2 Related work on mobility

Dealing with mobility, another challenging issue is a trade-off between radio link bandwidth availability and frequency of handovers: a high bandwidth radio link, in fact, can be realized, with the current technologies, only within a wireless LAN (WLAN). Anyway, the small distances involved in such a network, can create frequent handovers due to fast moving users. Mobile IP is not a suitable way to address frequent handovers, since the high amount of signaling required in the occurrence of a handover, and mostly the fact that this information is propagated up to the correspondent node, which can be far away from the MN, could introduce intolerable delays in data propagation.

A technique to achieve fast handover and improve the mechanism of providing an MN with a new care of address has been recently proposed [59]: this Internet Draft specifies protocol enhancements to MIPv6 that enable MNs to quickly change their point of attachment to the Internet, i.e., move from one AR to another. These protocol enhancements minimize the time during which a mobile node is unable to send or receive IPv6 packets and the agreed QoS could not be ensured to end-users. The way to accomplish this task is initiating the handover before the MN has moved to the new AR, to minimize the time required to transfer the information, by assigning to the MN a new care of address while still attached to its old AR.

The BRAIN project [60] analyses an all IP network, where users can access to Internet services in a seamless fashion; the system must manage real time multimedia mobile traffic regardless of the layer two technology used to connect the user to Internet. This implies that the system must guarantee support for IP mobility and end-to-end QoS transport, allowing roaming with the new cellular networks such as GPRS or UMTS. As stated in [60], one of the main goals of the BRAIN project is to facilitate the development of seamless access to existing and emerging IP-based broadband applications and services for mobile users in global markets and to propose an open architecture for wireless broadband Internet access, which will cooperate with the fixed Internet, emerging/mobile Internet specifications and UMTS/GSM.

The MIND (Mobile IP based Network Developments) project [72] broadens the concepts introduced by BRAIN to research the extension of IP-based radio access networks to include ad-hoc and wireless elements both within and attached to the fixed infrastructure. Starting from an IP core, which can be accessed from a variety of lower layer architectures, it will develop this vision with business models and user considerations, also researching applications for HiperLan 2 and IP based access networks. As stated in [72], the project will research the use of WLANs and an IP-based access network as complement to UMTS for high bandwidth provision in hot spot areas. In addition, it will facilitate the rapid creation of broadband multimedia services and applications that are fully supported and customised when accessed by future mobile users from a wide range of wireless access technologies.

In the following sections we present a short overview of two architectures designed for providing different levels of services to IP flows, ITSUMO and INSIGNIA and of two enhancements to the RSVP protocol that make RSVP work in a wireless mobile network too. We conclude the chapter introducing the context transfer solution to mobility issues, before analysing this new proposal in detail in the next chapters.

5.3 ITSUMO

The ITSUMO architecture [61] provides QoS following a bandwidth broker scheme; this approach is based on DiffServ, as the traffic is aggregated and forwarded in a backbone network according to PHBs. The architecture presents a QoS global server (QGS) and more QoS local nodes (QLN), which are the ingress nodes of the DiffServ domain; the MN communicates its QoS requirements directly to the QGS and if the latter accepts MN's requests, the terminal will be able to move within the domain and to receive the desired QoS.

The ITSUMO approach defines classes of service based mainly in the combination of latency and loss requirements; different applications, in fact, tolerate different values of latency and loss. Data applications (e.g. FTP), for example, need very low data loss, regardless of the timeliness of the transport; vice versa, real time applications suffer from delayed transport but accept higher loss of data. A classification of application in the different classes of service, according to latency and loss requirements, is shown in Figure 19:

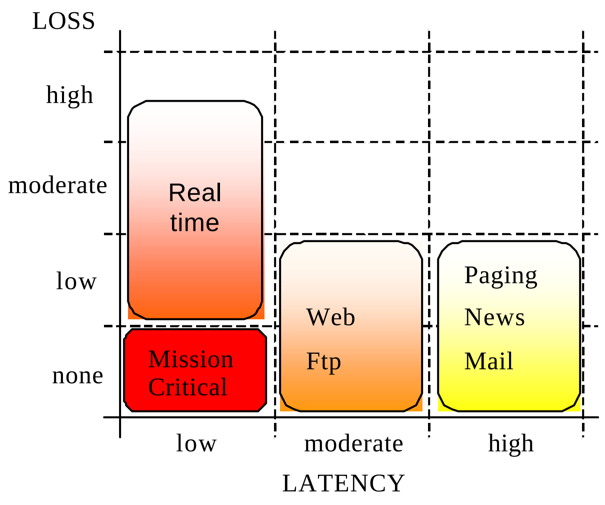


Figure 19: ITSUMO classes of service

The ITSUMO approach follows the philosophy of advanced reservations, where the mobile node has to explicitly request a reservation and specify a mobility profile; as anticipated above, the reservation is actually performed by the QGS, based on the local information and the mobility pattern negotiated in the service level specification (SLS) with the mobile node. The QGS reserves an amount of bandwidth in each QLN belonging to the specified mobility pattern and periodically updates the QLN likely to be visited by the mobile node.

One drawback of this approach is that the QGS must have global knowledge of the domain it pertains, making difficult the use of this scheme in places where the network topology changes frequently, such as hot spots and possibly presenting problems of scalability.

Figure 20 below shows the mechanism of explicit advanced reservations of ITSUMO.

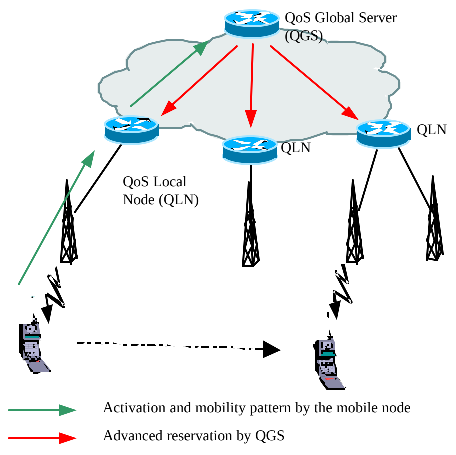

QoS Global Server (QGS)
QLN
QLN
QoS Local Node (QLN)
Activation and mobility pattern by the mobile node
Advanced reservation by QGS

Figure 20: ITSUMO advanced reservations

5.4 Mobile RSVP

RSVP is not adequate to make guaranteed reservations in a mobile environment, since they are requested and granted just when resources are actually needed. To obtain reservations independent from mobility, a mobile must perform the so called *advance resource reservations:* they can be classified into two groups: *admission control priority* and *explicit advanced reservation signalling.*

The first group bases its philosophy on the generally accepted fact that a wireless network must give higher priority to a handover connection request than to a new connection request. The basic idea is to reserve resources in each cell to handle handover requests, in such a way that new connection requests are not too penalised. There are different ways to calculate the amount of bandwidth reserved in a cell for handover requests, and [36, 38] depict them, underlining advantages and disadvantages.

MRSVP (and ITSUMO too) uses the second technique, because the former strategies are not feasible for both applications that tolerate variations in the QoS provided and those that do not tolerate any variation. While in ITSUMO advanced signalling is performed by a centralised entity, the QGS, in MRSVP [62] the mobile node explicitly signals reservations to locations it is likely to visit during its connection. These locations are retrieved from a set, called MSPEC; the main challenge in this approach is the determination of the MSPEC; in many case the mobile node specifies its own mobility profile. Two types of reservation are possible in MRSVP: active and passive (

Figure 21):

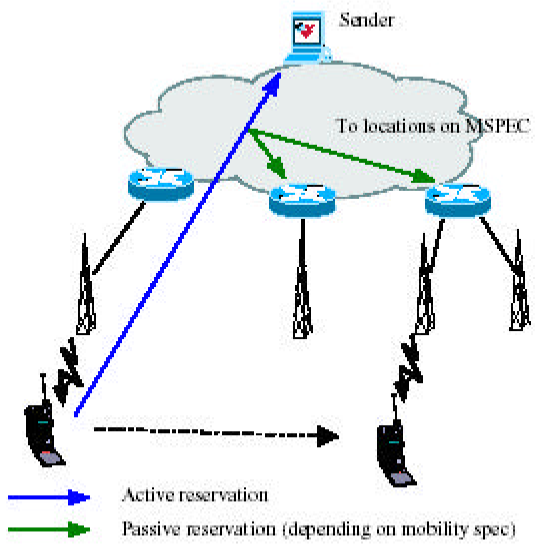


Figure 21: MRSVP advanced reservations

A mobile sender performs an active reservation from its current location, and a passive reservation from other locations in the MSPEC. To optimise the use of bandwidth, passive reserved resources can be used by other flows requiring a lower QoS; however, if a mobile node moves into a location where it had performed a passive reservation, this becomes active and low priority flows may be affected.

5.5 Localised RSVP

The localised RSVP (LRSVP) [64] is a proposal intended to enhance RSVP functionalities in radio access networks and can be exploited when it is not possible to guarantee an end-to-end QoS. One of the possible reasons for the lack of end-to-end QoS, can be the fact that the access link to the network is wireless; a wireless link has some characteristics that can prevent applications from receiving the desired QoS, among others a scarce availability of bandwidth. Moreover, the mobility of users introduces other problems related to the redirection of resource reservations along the new end-to-end path between the mobile node and the correspondent node. LRSVP, via small modifications to the standard RSVP protocol, allows achieving reservations locally within the access network. The proposal makes use of the Localised RSVP proxy server, an enhanced version of the RSVP proxy server described in [65], as the entity in charge of exchanging signalling messages with the mobile node to set-up reservations.

The solution addresses two of the problems typical of an RSVP proxy. It foresees a trigger mechanism in order to make the proxy act as sender RSVP proxy and by setting an opportune bit in the eight flag bits of RSVP session object, permits the proxy to recognize if reservations are made for the access network or must be intercepted by the correspondent node. The authors call the flag bit *Local Indication* (LI) and the enhanced RSVP proxy server, able to process packets with the LI bit set, *Localised RSVP proxy server* (LRSVP proxy). When an MN wants its reservations to pertain only to the access network, it sets the LI bit; if the RSVP proxy receives a packet with the LI bit set, it will no longer forward the packet towards the correspondent node but it will respond according to protocol steps.

Setting upstream reservations is simple and is achieved following the normal RSVP operations, with the difference that the LI bit is set if it is required that the validity of the reservation is local. The mobile node sends a PATH message with the LI bit set and the RSVP proxy responds with a RESV message (with LI=1) and when the mobile node has received the message, resources for the session defined in the PATH message have been allocated. The situation is a bit different for downstream reservations since, as noticed above, the RSVP proxy needs a trigger to initiate the RSVP reservation. Based on the behalf of the sender, the MN sends a new defined message, the Path Request message, with the LI flag set (to bound the message in the access network), identical to a standard PATH message besides the message type field. The proxy server uses the information contained in the received message to fill the proper field in a standard PATH message (LI=1), to be sent to the mobile node. The MN receives the PATH message and sends a RESV message (LI=1) to the proxy in order to actually reserve resources for downstream flows.

The issues with this approach are those common to the proxy server approach; the MN does not know the address of the right proxy server. The mobile can thus use the IP address of the correspondent node, so that a PATH message is routed through the normal IP mechanism, until a RSVP proxy gets the message. The problem with this approach is that, in case of downstream reservations, the data flow from the correspondent node can happen via a different RSVP proxy (when the latter is on the edge of the access network and also acts as gateway) due to the asymmetry of IP routing. An alternative is to multicast the address for all LRSVP proxies; each LRSVP proxy in the access network will respond to the mobile and resources will be reserved on several paths. To remove unnecessary reservations, a timeout mechanism can be used. If through a particular path there is no data flow, no refresh messages will be sent along that path and soft state removed by routers. The drawback to this approach is clearly that multicasting creates many signalling messages in the access network; moreover, resources will be reserved unnecessarily.

5.6 INSIGNIA

INSIGNIA [66] is an IP based QoS framework mainly designed to support adaptive services in mobile ad hoc networks. The primary features of the framework are in-band signalling and soft state resource management, which well address the problem of achieving QoS in a mobile environment, where the network topology and thus routing paths change frequently. INSIGNIA holds up resources fast reservation and restoration, end-to-end adaptation to QoS changes due to the robustness and scalability of IP networks.

The vantages that in-band signalling and soft state carry out are evident: approaches that rely on virtual circuits require explicit connection management and the establishment of hard state in the network before the connection, via an out-band signalling mechanism. Due to the intrinsic mobility of nodes in a mobile ad hoc network, this approach is not suitable, since the amount of data exchanged to set up a virtual circuit, a necessary operation when a node moves on and the path must be re-arranged, is too high for a wireless network. With an in-band signalling scheme, the control information is transferred along with payload data.

Adaptive services provide applications with minimum bandwidth when there is no availability of resources, allowing enhanced levels of service when they get free. Adaptation is an application-specific process. Data applications, such as FTP, would prefer to follow any change in available bandwidth at any time; adaptive media applications (e.g. audio and video) rather prefer some fixed levels of service delivery than a network that responds to every instantaneous change in bandwidth availability.

Protocol commands in INSIGNIA use the IP option field, as Figure 22 depicts, so that they can be carried along with IP data and the need for encapsulation is avoided.

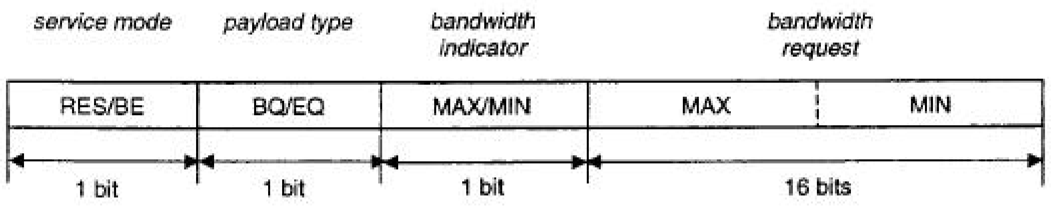


Figure 22: INSIGNIA IP option

The functionalities of the fields of the INSIGNIA IP option are well explained in [66], we will now describe the operation that the protocol executes.

Fast reservation

A source node performs a reservation by setting the RES bit. Reservation packets are processed hop-by-hop at the intermediate nodes to check whether there is resource availability until they reach the destination. If every node has enough resource availability, the bandwidth indicator field is set as MAX, meaning that the application can receive the desired level of service. If at least one node does not have resource availability, at the following hops, resources are allocated for the minimum and the bandwidth indicator will be set as MIN. In this case, all the reservation packets will receive a degraded service and the bottleneck will put the service level down from RES to BE (best effort).

QoS reporting

QoS reporting is used by the destination to inform the source about the level of QoS that the network can supply. When the source receives this message, then the reservation phase is complete. These messages are also deployed to handle end-to-end adaptations, to communicate to the source that an adaptation to new network conditions is required.

Soft state resource management

In order to manage reservations, intermediate nodes keep a soft state. The major advantage of soft state is that resources allocated during flow establishment are automatically removed when the path changes. In this case, a node does not receive any more packets of a given flow thus, when the timer associated to the soft state expires, resources are immediately released. When an intermediate router instead receives packets belonging to a flow, the timer for that flow is initialised, at every packet arrival.

Restoration

The goal of restoration is to re-establish reservation in a fast and efficient manner, in case for example of change of the routing path. Obviously, the INSIGNIA signalling system takes into account the various possible cases of a rerouting; flows could in fact be directed through a path where there is no availability of resources or in a lucky case through a light loaded path where the requested QoS is supported. These various scenarios, and the different kinds of restoration they bring to, are described in [66].

Adaptation

When the network could not support the level of service (or can support it if before it could not) due to resource unavailability or changes in the routing path, the destination sends QoS reports to the source to force it to adapt to network conditions. The kind of adaptation that the application running at the destination node requires at the source is completely up to it, based on the kind of application. The destination could send to the source a scale down command, forcing the latter to send its enhanced QoS packets as best effort. A drop command requests a source node to drop its enhanced QoS packets, by ceasing to transmit these packets. On the other hand, a scale up command signals that resources are again free in the network and thus the source can perform a new reservation phase with enhanced QoS.

5.7 Context transfer

We have seen that the major problem of mobile networks is the redirection of the host forwarding path in order to deliver packets to the new point of access correctly. The success of a time sensitive service, thus requiring a certain QoS, depends heavily upon the minimization of the impact of this traffic redirection and how seamless a handover is performed. This and the following sections present some introductory concepts to the context transfer problem, before it is discussed in more detail in the following chapters.

The treatment of IP packets within a network entails the collection of information about the packets at the network routers; this collection may require a certain amount of signalling time for the protocol exchanges. If the host is required to re-establish these services when it performs a handover in the same manner as it did when it initially established them, delay-sensitive real time traffic could be seriously impacted. If the mobile host must perform these signalling exchanges when it attaches to the new AP, real time applications might suffer from this delayed service reestablishment; instead, if the previously mentioned information is transferred to the new subnet in order to re-establish the service quickly, this could lead to a considerable improvement.

The *"context"* can be deemed as the information on the current state of a service required for re-establishing the service on a new subnet without having to perform an entire protocol exchange with the mobile host from the beginning; it is the information required to produce the necessary forwarding treatment at a node. The *"context transfer"* thus is the movement of context from one router or other network entity to another as a means of re-establishing context transfer candidate services on a new subnet.

With context transfer candidate services, we mean those concerned with the routing treatment of packets to and from the mobile host or the admission of a mobile host to network services. Examples of context transfer candidate services are Quality of Service, AAA and header compression. The context transfer may be helpful to minimize the impact of mobility on delay sensitive features like header compression and QoS, or simply to transfer state information to the new router in advance in order to improve the performance of stateful protocols by activating the state and related services with less delays and signalling overhead.

However, context transfer may not always be the best solution for re-establishing the context transfer candidate services on a new subnet. There are certain limitations to when context transfer may be useful. Context transfer between two routers is possible only if the receiving router supports the same services as the sending router. This does not mean that the two nodes are identical in their implementation, nor even that they must have identical capabilities. It is obvious, however, that transferring the context to a low resource node might cause degrading of the level of service or even its disruption; so an essential requirement in selecting the router to which to perform the handover is that it can support the transferred context. If a forwarding node cannot support the received context, it should refuse the transfer, resulting thus in a situation not very different from handover without context transfer.

The primary motivation for developing context transfer assumes that service sustaining during handover is attractive. Nevertheless, there may be situations in which either the user or the network administrator would prefer to release the connection and re-establish in the usual fashion the service level. It is the case of when the mobile node attempts to attach to the forwarding node of a different administrative domain.

The context transfer may apply, for example, to two of the proposed architectures for a QoS-enabled network: Integrated Services and Differentiated Services; they both require that routers keep some information in memory. In the former case, the router keeps information and service requirements for each flow, in the latter case, the router performs a scheduling of flows in aggregate classes and makes reservations for the whole class and not for the single flow, thus needing classification information.

In a handover situation, all the candidate ARs will need the QoS context used by the old AR to support MN’s microflows. Once the information is available, thanks to context transfer, at the potential new AR, it can be processed to make any decision on whether the whole context can be supported or it can be supported only in a part, with regard to the available capabilities of the AR.

Capabilities may include support for the QoS mechanism (for example the router may not have DiffServ support), size of buffers, availability of resources and so forth. In absence of context transfer, significant QoS protocol exchanges are required to re-establish QoS state information at the new AR and time sensitive services might suffer from this situation. However, context transfer alone is not sufficient to provide end-to-end QoS since all the routers in the path between the mobile node and the correspondent node must support the QoS required.

A new IETF working group, Seamoby [63], is studying the context transfer problem, with the aim to provide seamless mobility through wireless networks. The work of this group will hopefully lead to better mobility support, especially for multimedia applications; context transfer issues are still under development, and much work must be done in order to present the context transfer as a valid solution to the mobility problem. At the time of writing, the debate about the requirements of the context transfer solution, the operations that a context transfer protocol must perform and the functionalities that a context transfer framework must support is ongoing.

In the following sections, we will follow the steps performed by the Seamoby working group in the realization of a framework for context transfer, comprising network entities able to support this technique and the protocol needed to manage interactions between these entities. The attention of the working group, at the time of writing, is intensely focused on the definition of the problem statement and of the requirements that the context transfer architecture must meet. The problem statement draft [73] explains why this architecture is being designed, and which advantages it will, or should carry out; the requirement draft [74] defines the functional requirements of the context transfer solution.

5.8 Useful definitions

In this section, we will present some definitions of terms, which will be used widely in the remainder of the thesis.

- *Microflow:* [21] defines it as a single instance of an application to application flow of packets destined to or originated from an MN, which is identified by source address, source port, destination address, destination port and protocol id. It is the smallest component of traffic sent to and from a given MN that may have a distinct context. IP microflows can be aggregated, i.e. Differentiated services are provided to aggregates of IP microflows

- *Feature:* It is an aspect of the IP forwarding treatment instantiated for each MN. Examples of features are header compression, security and QoS. Each feature is supported by a set of specific protocols; thus, an instance of a feature reflects the state associated with protocols and the corresponding transactions.
- *Feature state:* At a given instant in time, the state of a feature instance is represented by the values of the data elements associated with the feature instance. Some of these elements are time invariant and represent the configuration of feature instance, while other elements, the state variables, change value over time in support of various protocol operations related to that feature.
- *Feature context:* It is the condition or state of an instance of a particular feature (or better the information associated with a particular feature) and represents the current evolution of the behaviour of that feature instance. A feature context is the smallest indivisible unit of context that can be transferred between forwarding nodes.
- *Microflow context:* A microflow context is the totality of various feature contexts associated with a single IP microflow. It is the smallest unit of context that must be transferred in order to re-establish support for the microflow at a forwarding node.
- *Seamless:* Seamless is a handover in which there is no change in service capability, security or quality. A seamless handover would mean thus that mobility will not give the user of IP based services any reduction in the QoS received. At the very least, the user should not perceive any degradation in QoS during a handover.

Figure 23 shows the reference architecture for the context transfer:

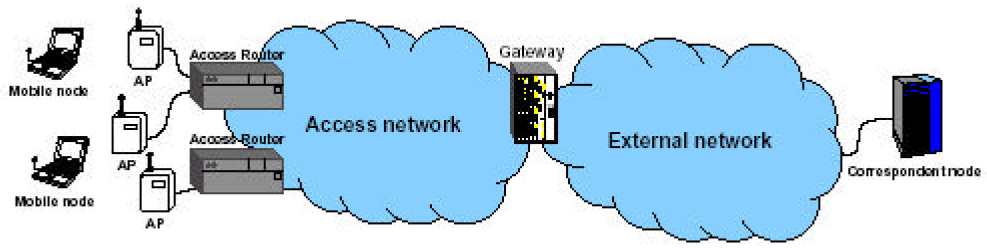


Figure 23: The reference architecture for the context transfer

The MN attains access to the network via an AP; the AP is connected to an AR, which has more APs connected to itself. Considerations about context transfer must be applied only when the MN performs a handover between ARs, i.e. a layer three handover and they deal with the radio access network. How to reach the correspondent node is a routing issue and it is not concern of context transfer.

5.9 Context transfer issues

In this section, we will describe some issues related to context transfer and we will try to outline some solutions, if they exist.

5.9.1 Selection of a generic architecture for context transfer

The Seamoby working group debated on two architectures for context transfer: a centralised one and a distributed one. The first architecture delegates all the functions pertaining to context transfer to a single central entity in the network or in a defined domain, which is in charge of keeping updated context information and distributing it to the potential receivers. This approach presents the drawbacks typical of all the centralised architectures, and is not suitable to large networks since the central entity should maintain a large amount of data to process and transfer in real-time with minimum delays; in other words, it is not scalable. Furthermore, in case of failure of the central entity, it would be impossible to carry out context transfer in the network.

The second architecture distributes the role of context maintenance and distribution to the ARs theirselves, which act as peer entities. This approach seems to be more appropriate as the ARs hold most of the context information (QoS, header compression etc) and all of them are actively involved in context transfer when there is need to transfer MN’s context information of the microflows. This means that this architecture is scalable and less subjected to failure since if a candidate AR tears down, it could be possible to replace it with a suitable one.

5.9.2 Definition of sender and receiver

A mobile may be connected to more than a single AP. The case where it moves between two APs connected to the same AR, no context transfer is required, since the MN has performed just a layer two handover, but when the potential APs are connected to different ARs, they must initiate the context transfer to receive the context information. A context group can be defined as the group of potential ARs in the network that have, or need to have, the context information related to a MN. There are a few issues to address in the dynamic formation of a context group:

- The addition and deletion of the members of a context group.
- The way in which a new member of the context group can identify the other members.
- Transferring the context information within the group

The first issue could be triggered by certain network events or events generated by the mobile's movement prediction in order to find out if the mobile is joining or leaving the group. For the second issue is required an identifier for each context group and that the information of the identifier should be propagated to any potential member of the context group. When an AR signs up a context group, the problem of recognizing the sender of the context arises, in order to properly retrieve the current context. There is also the need to transfer the context information in a timely manner: two possible approaches are "reactive" and "proactive", which will be discussed later. Finally, one must also pay attention to the transfer of updates in the context, in such a way that all the members can receive the update and understand it.

5.9.3 Partially failed context transfer

The "context information" may not be completely supported by one or more of the receivers of the context. The reason could be a different set of capabilities available at the receiver of the context data, for example a router not designed for DiffServ could not properly support the transfer of the context DiffServ QoS. Another simple example of such a scenario is failure of admission control due to unavailability of resources required by a "sub-context". The impact could lead to service degradation or even its disruption for one or more sessions of the mobile.

The puzzle is that the capabilities of a receiver cannot be known without actual transfer of context. The receiver may then decide whether a complete support of the requirements specified in the context is available or not. In the latter case, there is a possibility of service degradation for one or more of the mobile's sessions.

Two possible scenarios can be considered:

1. It is avoided to handover context traffic to receivers that cannot support entirely the transferred context.
2. The mobile's traffic is transferred to multiple targets, in such a way that different receivers may support different sub-contexts of the mobile. This scenario requires the source AR to decide which sessions are to be moved to which target based on any feedback from the target ARs. Nonetheless, this is a complicated situation to handle.

When the context transfer cannot be successfully completed, this could lead to a situation of handover without context transfer, but this should not be considered a failure situation, since context transfer is a solution to improve performances after a layer three handover and not an absolute requirement in a mobile network.

6 A context transfer framework

In this chapter, we will present a framework for the context transfer problem. We will focus our attention on the different mobility scenarios that can be addressed in a context transfer environment and on the functional entities involved in transferring the context. We will also examine the question of discovering the candidate AR in a situation of context transfer. Many authors have presented to the Seamoby working group individual submissions, which illustrate the characteristics that a framework and a protocol enabled to context must hold [75,76,77]. In this chapter, we will summarize these drafts, in order to better understand the point where the Seamoby group has arrived.

6.1 Terminology used in the following sections

This section defines a few additional terms related to the context transfer framework proposed in [76].

- *Context Group (CG)*: is a logical group of ARs. The context group is comprised of the AR currently supporting a given MN's microflow, and those ARs maintaining an updated replica of the context associated with the MN's active microflows. These latter members may never be required to forward the MN's microflows because only the router actually supporting MN’s traffic can perform the transfer.
- *Context Transfer Source (CTS):* is an AR that is actively supporting one or more active microflows of an MN. It transfers the context of the microflows it is supporting to one or more context group candidates.
- *Context Group Candidate (CGC):* is an AR that receives a notification of a MN arriving in its communication area. The AR thus becomes a candidate for transfer of the context associated with the MN's active microflows.
- *Configuration Context*: represents the part of the overall context information (for each of the sub-context associated with the MN's active sessions) that remains unchanged throughout the life of the session. Examples of such information could be the bandwidth requirements for each session between the MN and the access network and user authentication information that is determined for the MN at the time of its attachment to the network.

- *State Context*: it represents the part of the overall context (for each active microflow) that is updated on each packet arrival. Examples of such information include the metering and header compression states.

6.2 Requirements for a context transfer framework

The efforts of the Seamoby working group have also been directed towards the definitions of the requirements that an efficient context transfer solution must retain. The Internet Draft [74] specifies these characteristics, even if it does not provide a practical implementation; in this sense, many individual submissions have been produced within the working group. In the following sections, we will furnish a list of the main requirements, while the whole roll is obviously in [74].

6.2.1 General requirements

This section addresses the facilities and services required in the access network to properly support context transfer, since it will have to assume certain characteristics in order to ensure a successful performance.

- The context transfer solution must define the characteristics of the IP level trigger mechanisms that initiate the transfer of context.

Handover at the IP level is a consequence of a change in the path used to communicate between the MN and the access network. The mechanisms utilized to change the communications path at layer two are specific to the physical characteristics of the medium. To maintain IP sessions during a change of a layer two path, the IP level must somehow receive an indication of this path change. One example of an indicator would be the trigger event that initia tes context transfer.

An MN can access a network through more than one physical path, via different ARs. This situation implies that two or more ARs are candidates to receive the MN’s context after a handover, and so require the appropriate IP context when forwarding commences. Discovering a suitable AR is matter of discussion, considering that before the handover is actually initiated, the target AR is unknown to the network; anyway, the process of identification of the AR helps to accelerate the handover process.

- The context transfer solution must support a distributed approach in which ARs act as peers during the context transfer.

The reason for such a requirement have been explained earlier; moreover, it is believed that a distributed approach minimizes the number of protocol exchanges, and thus signalling messages on the air, giving a considerable performance improvement.

- The entities transferring the context must support a process for mutual authentication before initiating the transfer.

Performing the authentication process during the handover process would add computation overhead for both the sender and the receiver, causing an unnecessary and even dangerous (for those time sensitive applications) latency to the transfer process. Therefore, context transfer peers must be authenticated before the initiation of any context transfer.

- Context information may be transferred in phases.

Foreseeing such a mechanism for exploiting context transfer enables the prioritisation of the transferred information. Proactive and reactive context transfer can be considered two phases of the same context transfer process.

- The context information to be transferred must be available at the AR performing the transfer, before the initiation of a given phase of the context transfer.

This is an obvious requirement, since the reason to perform context transfer is achieving seamless handovers, the context information must be ready to be transferred at the involved AR to carry out a rapid transfer of the information. If the context transfer is comprised of just one phase, then the entire context must be available before the initiation of the context transfer.

- The context transfer solution must be scalable.

Another clear requirement. However, there has been much discussion within the Seamoby working group, to specify how much the solution must be scalable. It is true that to ensure the success of the architecture, scalability is a major goal. The best way to achieve it seems to exploit a distributed approach to context transfer, the one where ARs act as peer entities.

6.2.2 Protocol requirements

This section captures the general requirements the context transfer protocol must meet in order to ensure an efficient communication between context transfer entities and allow a fast and reliable transfer of the information. Some of the requirements are presented in [75]: in this Internet Draft, Syed and Kenward provide the requirements that a context transfer framework should meet.

- The context transfer protocol must be capable of transferring all of the different types of feature context necessary to support the MN's traffic at a receiving AR.
- The context transfer protocol must operate autonomously from the content of the context information being transferred.
- The context transfer protocol design must define a standard method for labelling each feature context being transferred.

The first two requirements make the protocol efficient. If the protocol was dependent by a particular feature context, it would lose generality and its application to the context transfer solution would become a difficult challenge. To achieve this generality, the information that concerns a given feature context, needs to be encapsulated and labelled, in a standard way, to keep separated each other different feature contexts. Based on the label associated to the encapsulated packet, the receiving AR should be able to process the transferred context appropriately.

- The context transfer protocol must be capable of updating context information when it changes.
- A context update must preserve the integrity, and thus the meaning, of the context at each receiving AR.

Since the context at the AR actually supporting MN's traffic can change with time (it is the case of state context), any change of context at the supporting AR must be communicated at those ARs that have already received the context for that MN in order to preserve the consistency of the information present at the ARs.

- The context transfer protocol may provide a mechanism for supporting partial handovers.

In a situation where a single AR does not have the capabilities to support all the microflows instantiated by a MN, the context transfer protocol should allow different IP microflows to be handed over to different ARs. To achieve this, the protocol must foresee a mechanism for dividing the context so that an AR receives only the appropriate context.

- The context transfer protocol must deliver the context without duplication or re-ordering of the information.
- The context transfer protocol must transfer the context fast enough for the information to be meaningful at the receiving AR.
- The context transfer protocol must provide a method for synchronizing context information when it changes.
- The synchronization of the context must preserve the integrity, and thus the meaning, of the context at each AR that has received the context.

The context at the AR actually supporting traffic from the MN changes over time. Besides this, an MN could create new microflows or could delete existing ones, with a timing given approximately by the distance of two consecutive packet arrivals in the MN's traffic flow. Regardless of the changes that happen at the AR that actually retains the context information, the protocol must ensure that this information is present, after a handover, at the new AR, taking care of retransmission of corrupted packets and avoiding reordering of packets too.

6.2.3 Security requirements

- The security of the context information exchanged between ARs must be ensured.

The transfer of the information must be safe. This issue is origin of an extensive discussion within the Seamoby working group, both for the kind of security features the solution must retain and for the way in which security can be accomplished. Security provisioning includes protecting the integrity, confidentiality and authenticity of the transfer.

- The security provisioning for context transfer must not require the creation of application layer security.
- The protocol must provide for the security provisioning to be disabled.

The creation of application layer security is a useless complication for the framework supporting context transfer; an efficient design of the architecture should therefore avoid introducing new entities beside those strictly necessary allowing the exploitation of security functions without resorting to other methods. In some environments, the security provisioning provided for by the context transfer protocol may not be necessary, or it may be preferred to minimize the context transfer protocol overhead to avoid delays and latency which could make obsolete the context information being carried.

6.2.4 Timing requirements

This section captures the timing requirements for the context transfer protocol, a concern of primary importance, especially when the context transfer is applied to real time applications. Since on the radio link there is scarcity of bandwidth, the signalling exchanges required by the protocol could affect those applications that require a timely delivery of their data. The discussion is ongoing.

- A context transfer must complete with a minimum number of protocol exchanges between the source AR and the rest of the ARs.

Context transfer is designed to be a peer-to-peer interaction. If performing such an interaction requires a high number of protocol exchanges, the unreliability, resource consumption and completion time of that interaction rise considerably. Keeping the number of these exchanges to a minimum, the reliability, resource utilization and completion delay of the transfer should improve. Too many protocol exchanges, moreover, could also worsen latency and delays of the transfer.

- The design of the context transfer protocol must minimize the amount of processing required at the sending and receiving ARs.

Another cause of remarkable delay in the context transfer process could be the processing of context information at the routers involved in context transfer. If the context transfer protocol is not optimised in order to guarantee minimum processing of information at ARs, delays that affect the reliability of the context may occur. In other words, the context may become obsolete before it can be reconstructed at the receiving ARs.

- The context transfer protocol must meet the timing constraints required by all the feature contexts.

A given feature context may have timing constraints imposed by the nature of the service being supported. The delivered context must always comply with the requirements of the service to be useful. This topic is very important with regard to the context transfer for applications needing real-time services.

6.2.5 Management mechanism of the context group requirements

- The management mechanism of the context group must define the events for inclusion and removal of the ARs from the context group.
- The management mechanism of the context group must support collection and distribution of the ARs group membership information.
- The management mechanism of the context group must define how to identify the context source AR for the newly joined AR.

Management of the CG is a dynamic process, in which the number of members can vary continuously over time due to MN's mobility; the most well defined event causing the adding or the removal of an AR from a CG must be strictly related to a relative movement of the MN with respect to the coverage area of AR. It is necessary that all the information pertaining to an MN must be exchanged between CG members in order to be readily available in case of handover of the MN to that AR, in such a way that each member of the CG maintains an updated replica of the context associated with the MN's traffic. It must be clear also which AR is in charge to process the MN's context in a given instant of time, mostly for a newly joined AR.

.

6.3 Transferring the context within the framework

This section provides the details of the way in which the framework replicates the context associated with MN's traffic flows to one or more receiving ARs. The framework identifies the functional entities participating in the context transfer and describes the role of each functional entity. We will now describe the major functional elements of the context transfer framework: the context transfer agent (CTA), and the membership collection and distribution function (MCDF).

Context transfer agent

The context transfer agent (CTA) is the functional entity that actively participates in the process of context transfer. It resides at the peers of the context transfer, and is responsible for collection of all the information required to support the active microflow associated with the MN to create the overall context that needs to be transferred. The CTA performs several tasks, such as storing the context information in order for it to be correctly transferred between ARs and cooperating with the MCDF in the managing of the CG. At the receiver side, at last, the CTA takes care to distribute the received context to the involved entities for processing; the first action to be undertaken is the admission control to check whether the AR has the capabilities to support the received context.

Membership collection and distribution function

The membership collection and distribution function (MCDF) is a functional entity that administers the formation of context groups. It creates and maintains one context group per active MN, collecting the identification of the ARs that are supporting the active microflows of the MN. Each of them can become a potential source of the context for the MN, i.e. the AR that actually handles MN's traffic. When a MN gets in the coverage area of an AR, the latter becomes a CGC and needs an updated replica of the context information associated to that MN and to know which is the CTS for the MN. This information is collected via the MCDF.

Syed and Kenward, in [76], provide some sample scenarios to explain the interworking of the various functional entities involved in context transfer, and the correspondent protocol used. The Internet Draft describes a typical case of managing a CG. The sequence of events described comprises an AR that joins the CG, the creation of a new CG and the way in which a new member can receive the context information.

However, defining the functional entities involved in context transfer is not sufficient to achieve a successful framework, since the data structure representation of context information and packet formats needed to transfer context information must be defined. The Internet Draft [77] discusses these topics, depicting also an explanation of the handover signalling required when a context transfer is performed, with the relative description of the messages exchanged in such an operation. However, for lack of space, we cannot provide this description (as well as the example of CG management); the interested reader can refer to [76,77].

6.4 Forwarding path handover scenarios

When an MN performs a handover, it can either maintain the connectivity with the access network during the requested time or lose it. Two possible scenarios can be described [75]:

Break before make

Break-before-make is a term used to describe a discontinuity in connectivity between an MN and the access network during a handover. With a break-before-make handover, the forwarding of MN's traffic along the current path is interrupted before the connection with the new AR is established. In break-before-make, there is a period of time where the MN's traffic cannot be forwarded, since the connectivity is lost during the handover. An example will clear out the situation; an MN moves from an AR to another (or better from the AP of an AR to the AP of another router). In a break-before-make scenario, the MN's traffic through the old AR, and old AP, is stopped before being redirected through the new AR / AP pair.

Make before break

Make-before-break is a term used to describe the continuity of connectivity between an MN and the access network during a handover. With a make-before-break handover, the forwarding of the MN's traffic along the new path is carried out before the old path is released. For example, an MN moves between the APs of two ARs. In a make-before-break scenario, the MN's traffic will be forwarded to the new AR, and the new AP, while the old AR/AP pair continues to forward traffic. Thus, in make-before-break, a period of time in which the MN’s traffic is present in both the old path and the new one exists.

6.5 Context transfer scenarios

This section aims to describe proactive and reactive context transfer and the way in which they may be achieved in the framework, while paying attention to the issues related with these two context transfer scenarios in order to describe the context transfer framework correctly. Proactive and reactive context transfer are associated with the actual performing of the context transfer and with the need of transferring the context quickly in order to preserve its meaning at the new AR.

6.5.1 Reactive context transfer

Reactive context transfer occurs when an MN actually attaches itself to the new AR and thus the context information required to completely support an IP micro-flow is replicated to this AR. With reactive context transfer, the context is made available to an AR when the need is imminent. This means that to trigger a reactive context transfer an event takes place (it can be either the initiation of handover or another event after the handover).

Reactive context transfer is not the best solution for context transfer because it may not complete as quickly as required (for several different reasons) resulting in loss of meaning of the transferred context or even its obsolescence. Instead, it must be coupled with a proactive context move to provide successful information relocation, since transferring the context just in a proactive manner could lead to an obsolete version of data at the receiving AR, while performing simply a reactive context transfer could be insufficient when timing constraints must be met.

6.5.2 Proactive context transfer

The proactive context transfer allows the transfer of the context for the MN to the context group members before an actual handover is required. The main reason to perform a proactive context transfer is to achieve the possibility to effect admission control at the new AR before the MN executes a handover, in order to be sure that the AR can support the received context and to minimise any disruption of the service provided to an MN. With proactive context transfer, the context is available to the new AR before it is needed, that is to say before the handover of the MN's traffic is complete. This implies that some network events, which could lead to a proactive context transfer, must occur to indicate the possibility of a handover, before the MN initiates to change its point of attachment to the network. Nevertheless, there can be no guarantee that this event will occur, and thus, a reactive context transfer is always required.

This approach presents the issue that during the time from when the context is transferred proactively and the actual handover of the MN's traffic to another AR occurs, the context at the context transfer source (CTS) may be changed. Therefore, transferring it to the new AR is conceptually wrong since the new AR would be provided with an inexact copy of the MN’s context.

6.5.3 Issues related to the two approaches

To address the problem of inconsistency of data provided to an AR after a proactive context transfer, it is to be considered that the context information for the MN is composed of both the configuration and state components (see section 7.1). The context information involved in admission control is the configuration one, since the state component, information updated on each packet arrival at the source, plays no role when performing this task. Thus, separating the context into configuration and state components, context transfer can be performed in a progressive way.

In particular, according to the above considerations and to the goal of proactive context transfer, the configuration context is transferred proactively, and the state information reactively, when a handover is really in action. The information transferred reactively (i.e. header compression state) is updated at every packet arrival. Since the configuration context changes very slowly over time, events that trigger the need for an update from the CTS to the other context group members might be simply the addit ion and deletion of microflows to the overall context associated with the MN's traffic at the CTS. In this scenario, proactive and reactive context transfer are really two possible phases of a single attempted context transfer.

The framework must support both proactive and reactive context transfer, since, as we have seen, the presence of only one type of context transfer mode in the framework is not sufficient to ensure the continuity of service. To summarise, only reactive context transfer is not feasible in the network since some information could not be transferred and made available timely at the new AR; only proactive context transfer is not suitable either, as the state context must be transferred in the imminence of a handover, being composed of information updated at every packet arrival.

To clear out ideas about context transfer scenarios, [76] provides examples of both proactive and reactive context transfer modes, focusing on the progressive transfer of configuration and state context and evidencing the protocols and the functional entities involved in the process.

6.5.4. Demand and reservation states in the proactive context transfer

The proactive transfer of configuration context and the subsequent admission control on this context determine the level of support for the MN's traffic if this is re-directed to the AR. Due to the fact that the transfer of the context to an AR happens before the actual handover of the traffic to the AR (because of the intrinsic nature of the proactive context transfer) the resource availability at the AR may not be the same as was reported after the admission control process on transferred context. To solve this puzzling matter, it is necessary to put the AR into one of the two states in terms of resources: the *demand* and the *reservation* states.

In the demand state, the AR is aware that it has the capability to support the MN's traffic, but it does not perform any attempt to reserve resources. Thus, when the MN finally executes a handover, the necessary resources may not be as available as they were when checked during the admission control phase, and the handover to the AR may fail partially or completely. To ensure an acceptably low level of failed handovers, the policy is to overprovision resources during admission control. This option could be used for the services that may not require immediate resource availability. Examples could be http sessions, FTP, e-mail.

In the reservation state, an AR not only is aware that it has the capability to support the MN's traffic, but also reserves the resources necessary to MN’s traffic. In this case, when the handover is finally attempted, it should succeed. This option minimizes handover failures, but allocates resources for potential handovers that may never occur (we remind that *proactive* context transfer could never come true). This could be useful for services that cannot tolerate disruptions like VoIP, multimedia, or video conference, since the reservation mode guarantees the availability of resources for such services immediately after the handover.

6.6 Discovering the candidate access router

IP mobility protocols enable MNs to change the ARs by which they obtain layer three connectivity to the Internet. Handovers in IP mobility protocols involve moving an MN's layer three routing reachability point from one AR to another, before or after the MN has established a layer two connection with the radio AP that is covered by the new AR. In addition, other context information about the MN's packet session may be transferred from the old AR to the new one, in order to minimize the service disruption during the handover process.

It is not sufficient to just transfer the context information enough before the actual performing of a handover so that it is available to the AR; it is also necessary that the new AR can support the received context. What is required is a protocol that would allow an AR or an MN to discover the neighbouring ARs whose capabilities meet the requirements of the MN, thus becoming potential target ARs for a handover. Such ARs are called candidate ARs. The issues in candidate AR discovery are addressed in [78], a draft produced by the Seamoby working group. In the following sections, we will point out the concepts expressed by the draft, underlining the most important questions.

6.6.1 Terminology used in the CARdiscovery problem

To better understand the problem of candidate AR discovery, it is useful to give some definitions, strictly related to the argument:

- *Geographically Adjacent AR (GAAR)*: it is an AR whose coverage area is such that a MN may move from the coverage area of the AR currently serving the MN into the coverage area of this AR. In other words, GAARs have APs whose coverage areas are geographically adjacent or overlap.
- *Capability of an AR:* it is a characteristic of the service offered by an AR that may be of interest to an MN when the AR is being considered as a handover candidate.

- *Candidate AR (CAR):* is an AR that is a candidate for MN's handover. CAR is necessarily a GAAR of the AR currently serving the MN, and has the capability set required serving the MN.
- *Target AR (TAR):* is an AR with which the procedures for the MN's IP level handover are initiated. TAR is usually selected from the set of CARs.
- *TAR Selection Algorithm:* is an algorithm that determines a unique TAR for an MN's handover from the set of CARs.

6.6.2 Motivation for candidate access router discovery

This section describes some scenarios where the procedure of discovery of a CAR could be useful in order to improve the performance of the system:

Load balancing

In a situation where an MN is attached to an AR, say AR1, heavily loaded, CAR discovery is useful to discover a CAR, say AR2, which has the capabilities to support the MN's traffic and is not heavily loaded. Under these assumptions, AR1 can initiate a handover of the MN to AR2 to alleviate some of its own load. In order to accomplish this, AR1 needs to have the knowledge of the capabilities of AR2 and the load on AR2, that is to say, of the neighbouring ARs. Such information sharing can be obtained with the help of CAR discovery protocol.

Resource intensive applications

In a situation where an MN is running a streaming video application, or in general an application requiring an intensive use of resources, handover of the MN to a new AR could lead to a situation where the new AR cannot support the high bandwidth and possibly other QoS supports needed by the MN. CAR discovery provides means to face this problem. The MN can then be informed about the possible resource unavailability at the new AR when it is still connected to the current AR. Clearly, to accomplish this, the current AR needs to have information about the capabilities of the neighbouring ARs, and this can be achieved using the CAR discovery protocol.

Adaptability to change in the coverage topology

In a situation in which some ARs are temporarily introduced in hot spots to satisfy the existing traffic demand, a static configuration of the neighbourhood information in ARs is not feasible. A protocol is therefore needed in such cases to allow an AR to identify automatically and dynamically any change in the coverage topology and exchange capability information with the neighbouring ARs. This can be facilitated by the CAR discovery protocol.

The CAR discovery applies very well to the context transfer problem, since the discovery of a suitable AR, capable to support the MN's traffic, is a major issue, probably one of the most challenging that the Seamoby working group is facing.

6.6.3 Candidate ARs discovery problem

There are two basic approaches to CAR discovery, namely, Anticipated CAR Discovery and Dynamic CAR Discovery.

Anticipated CAR discovery

In this approach, an AR currently serving the MN identifies all the CARs for the MN's handover before the handover is actually performed. This information then constitutes an input to the TAR Selection Algorithm. Another input to the TAR Selection Algorithm may be the preferences expressed by the MN before the handover, in terms of maximum cost for the service for example. Anyway, at the time of handover, it may only be required that the only input to the TAR Selection Algorithm is the reachability of neighbouring ARs from the MN. The advantage of this approach is that the handover can be executed quickly because the AR has already collected much of the information needed by the TAR Selection Algorithm. Moreover, this approach does not generate much radio traffic for performing CAR discovery as the capability exchange happens over the wired network. However, it is not sensible to changes in the network topology, since routers must know contingent changes happened in network topology before identifying CARs.

Dynamic CAR discovery

In this approach, an MN dynamically obtains information about the ARs available for a handover, along with their capabilities. When the MN detects an AR that has capabilities to support its preferences, the MN may require the currently serving AR to perform a handover to this AR using one of the protocols designed for a seamless handover. The advantage of this technique is that the MN retains fine-grained control over the TAR selection algorithm and the adaptability to changes in network topology. However, this approach generates more radio traffic during the CAR discovery process, since the MN receives its capability information over the air. In addition, it is required that the MN can receive IP-level capability information from the GAARs and negotiate requirements with the GAARs, while it is still attached to the current AR, quite a problematic situation. A hybrid approach, which holds the features of the two described above, is also possible.

6.6.4 Specific issues in CAR discovery

The process of discovering a CAR is not straightforward, and a number of issues must be taken in consideration when such a process is performed, either in an anticipated or in a dynamic way. The most important of these issues are identifying routers that can be deemed GAARs, and subsequently checking their capabilities.

Identifying GAARs

In order for an AR to be considered a CAR for MN's handover, the coverage area of this AR must be “geographically" adjacent to or overlapping with that of the AR currently serving the MN. In other words, a new AR must be a GAAR. The geographical vicinity of the coverage areas of two ARs does not necessarily entail their "logical" adjacency and, in the same manner, the logical adjacency does not imply the geographical vicinity of the two coverage areas. Logical adjacency of two ARs means that there is just one IP hop between the two ARs, while the vicinity of the coverage areas of two ARs implies that the MN can actually move from the coverage area of one AR to another, even if the IP addresses of the two ARs belong to different administrative domains. A major requirement to perform a handover correctly is that the current AR owns a list of ARs to which the MN could attach, that is, the current AR must hold a list of CARs. The only requirement these ARs have is that their APs must be geographically adjacent, even if they are in different administrative domains.

In order to identify the GAARs, and thus to build the list of CARs, one approach is to manually configure this geographical neighbourhood at each AR. However, such an approach has disadvantages and in many cases, may not be feasible, since the GAARs may be under different administrative control. In this case, GAARs would not be acquainted with each other's presence. Moreover, the manual configuration approach is not suitable in the case of quick changes in network topology, or in the case that the ARs can be physically relocated and their coverage areas changed, thus altering their geography. Such a scenario is very common when ARs are temporarily introduced in hot spots, in order to satisfy particular resources demands. Clearly, a more automatic and dynamic mechanism to discover GAARs without administrative intervention is necessary to exploit the mechanism of CAR discovery in the best way.

This process is performed in a different way, based on the kind of discovery, either anticipated or dynamic. For the Anticipated CAR Discovery, the process of identifying GAARs requires a protocol that allows ARs to exchange the information about the geographical adjacency of their APs, in advance of handover. In Dynamic CAR Discovery, the MN automatically decides whether an AR is a GAAR, based on the information that ARs sent to it, such as the strength of the received signal.

Identifying capabilities of GAARs

In the future generation mobile networks, there may be ARs that are GAARs of each other, but are situated in heterogeneous administrative domains, and have different functionalities, link layer technology and resources. An MN attached to a given AR may have specific requirements as regards these parameters; for example, the MN can run an application that requires a particular hardware or software support from the AR. For these reasons, identifying the features and the capabilities of a GAAR is an important task in the difficult challenge to provide a correct identification of a CAR

In addition, support for QoS, security, multicast, header compression must be taken into account when choosing the CAR for an MN's handover. It could be required to assure some security association, policy agreement or billing contract between the current AR and its GAARs in order to allow the MN's handover between them. The MN might prefer to not handover to an AR of a different administrative domain, even if it has all the requested capabilities, if the cost of the services provided is high compared to the user's requests. Finally, the ARs need to exchange information with their GAARs regarding the correspondence between their IP addresses and the identities of the APs connected to them.

In the Anticipated CAR Discovery, the process of capability mutual knowledge between ARs occurs over the wired Internet. Thus, precious radio link resources are saved, but a protocol to allow ARs to exchange capability information is necessary. The process of capability exchange between ARs or between GAARs should also be periodic, in order to maintain at each AR an updated snapshot of the capabilities of the other ARs, either neighbouring or not. In the Dynamic CAR Discovery approach, instead, the MN has the task to solicit the receiving the capabilities of GAARs directly, exchanging with them messages over the air. A protocol is required to allow the MN to perform this task and the ARs to communicate their capabilities.

6.6.5 Applicability of new and existing protocols to the CAR discovery problem

The simplest possible solution to the CAR discovery problem is to statically configure the ARs in two geographically adjacent domains, keeping in their memory the IP addresses of the routers that belong to the other domain. However, this solution is impractical, particularly when the geographically adjacent domain belongs to another administrative entity, as routers of a domain could not know the IP addresses of entities belonging to other administrative domains. Another solution is to use a simple directory-based discovery solution, such as DNS [80], LDAP [81] or SLP [82].

Discovery mechanisms based on attributes, such as LDAP and SLP could potentially handle non-geographic capabilities, such as radio technology; moreover, the client-server nature of these protocols has the associated scalability and robustness concerns. One issue in this approach is that geographic information may be difficult to be converted into the form of attribute/value pairs. In addition, information on changes in AR availability needs to propagate dynamically between ARs rather than requiring the ARs to explicitly ask for it.

A new distributed hybrid protocol (see section 6.6.3) that aims to identify CARs before handover is proposed in [83]. In this Internet Draft, the authors try to accomplish the difficult task to dynamically discover CARs and build and maintain a suitable list of candidates (the Physical Neighbourhood list, PNL) that the MN will use. The protocol structure foresees that an MN identifies the GAARs, which exchange information about their capability on the wired Internet, by listening to beacons arriving from neighbouring APs and forwarding the information contained in the beacon to its current AR.

When a MN moves from its current AR to the new one, it forwards the IP level identity (usually the IP address) of the current AR to the new AR. The new AR adds this identity to its PNL if the entry does not exist, associating a lifetime, or simply refreshes the lifetime if the entry for that identity already exists. The MN sends this message, called “Router Identity” to the new AR as an ICMP option. A set of capabilities is associated to an entry in the PNL; when an AR receives a message containing the identity of a GAAR, it sends to it a capability exchange message over the wired Internet enlisting its own capabilities. The target AR, to which the handover will be actually performed, is chosen based on the capabilities information contained in the PNL.

The format of the PNL, the meaning of the related fields and examples of capabilities can be found in [83].

7 Considerations on Quality of Service context transfer

We have already introduced the context transfer problem; in this chapter, we want to focus our attention on the way with which the transfer of QoS context can be achieved. A number of proposals have been presented within the Seamoby IETF group, aiming to describe protocols and architecture that allow the transfer of specific feature contexts, such as the Header Compression feature context [84]. In particular, we will describe how the QoS parameters are managed during a context transfer and a solution for a context transfer that meets the strict timing requirements of real time applications.

7.1 Quality of Service context transfer

A seamless support of QoS to an MN during handovers is a crucial challenge for enabling a variety of services on the future Internet. The context transfer solution is being designed mostly to handle seamless handovers, in order to avoid disruption of service, which could be critical for certain kinds of applications. When an MN performs a context transfer, it is necessary that it also maintains at the new AR the amount of QoS it had guaranteed at the old AR: the choice of a proper target AR is crucial from this point of view, since it must support the QoS instantiated for the MN, otherwise it would be impossible to grant the required level of service.

The failure for the new AR to possess the QoS state, could lead to reestablish the QoS from scratch, which is an unsustainable situation for certain applications, since high delay is experienced when a connection is settled from the beginning. The draft [85] defines QoS contexts, showing how they could be transferred to enable seamless operations during handovers and also shows the message formats required to perform QoS context transfer. Syed and Jaseemuddin [86] describe the requirements for the transfer of DiffServ context, with details of the specific data that must be transferred in order to maintain the service provisioning to the IP QoS flows.

7.1.1 Differentiated Services feature context

The information related to a DiffServ feature context does not need to be entirely transferred to the AR the MN attaches to. An important role in the DiffServ context transfer is played by the DiffServ Boundary node (DSBN), which handles the traffic that passes through it (in both directions) and selects the flows that need to be provided with a particular forwarding treatment, specifying the treatment itself. Conceptually, a DSBN performs the following functions:

- Traffic classification
- Metering
- Marking and dropping packets
- Queuing packets

Some of these functions are specific to a particular DSBN: marking and dropping packets, as well as the determination of the forwarding queue depends on the configuration of the device. Therefore, the information associated with these functions, must not need to be transferred at each packet arrival as part of the DiffServ context. On the other hand, classification and metering constitute the part of QoS information that needs to be transferred at each packet arrival, since it is required at the new DSBN to correctly reconstitute the context associated to an MN.

Classification is performed by the so-called classifier elements; packets are assigned to a particular class by filters that match the content of the packets based on the following packet header fields:

- Source and destination IP address
- Source and destination ports
- Protocol ID
- DiffServ Code Point (DSCP)

A meter measures the temporal state of a flow, i.e. the rate at which the traffic stream passes it and is also used to perform conditioning actions on real time traffic. Typical parameters of a traffic profile are:

- Committed Information Rate
- Peak Information Rate
- Committed Burst Size
- Peak Burst Size

Based on the values assumed by the above parameters, meters can carry out the proper conditioning operation. [86] explains how these parameters can be used, showing examples of typical meters.

7.1.2 Requirement analysis for the Differentiated Service context transfer

The information contained in the DiffServ context can be categorized in information pertaining to the overall context that does not change during a session of a microflow and the one that is updated on every packet arrival at the AR. Examples of the first kind of information, referred to as the "configuration context" are packet classification and traffic profiles, since these functions involve parameters settled at the beginning of the connection and never modified during it. The information that changes at every packet arrival, such as flow state in a meter, represents the "state context".

Configuration contexts and state contexts obviously pose different requirements for the analysis of the overall DiffServ context requirements. Nevertheless, some considerations are common, such as the fact that the transfer of DiffServ context must be secure. The timing requirements that the DiffServ context transfer poses on the context transfer protocol must be met accurately, especially because it is likely that the involved application are real time.

Since the state context is updated at every packet arrival, the state context transfer must not be initiated before the last packet arrives at the old AR. Similarly, a retransmission of the state context, due for example to radio link impairments, could be dangerous; rather, an update of the lost state context with the most recent state present at the old AR should be transferred to the new AR.

7.1.3 Definition of Quality of Service context parameters

A detailed description of the parameters involved in QoS context transfer, cannot be presented here fore reasons of space; however, the drafts [77, 85] face this challenge, illustrating also the formats of messages involved in QoS context transfer. In particular, [77] gives a general data structure representation of context information and packet formats for encapsulating the context data structure that must be transferred and [85] builds on these concepts defining the packet formats for QoS context transfer.

A QoS Profile specifies the structure of the state variables used for QoS. The QoS Profile Type (QPT) provides a way to indicate the kind of QoS provided to a particular packet stream. A QoS context thus includes QPT and the corresponding QoS Profile. The values of the state variables, retrieved from the QoS profile, are interpreted according to the data structure and format defined by the QPT.

Examples of QPT are:

- Best effort
- IntServ Controlled Load service
- IntServ Guaranteed Service
- DiffServ default
- DiffServ srTCM
- DiffServ trTCM
- Generic

Note that the size of each QPT is fixed, so that it can be uniformly interpreted at ARs and no length fields are necessary in the protocol definition. The QPTs listed above, constitute only a part of those that can be implemented; for example, DiffServ srTCM or trTCM refer to the kind of marker utilised in the DiffServ architecture (either a single rate three colour marker [87] or a two rate three colour marker [88]), and different meters are possible in a DiffServ architecture.

The best effort QPT is the context that a router that does not distinguish between the packet streams maintains and is the default QPT if no other is provided. In the IntServ architecture, the controlled load service is the default profile and is used when the IntServ QoS profile is not known. The old AR uses the generic profile when it does not know the capability of the new AR. The format of the QoS profile is shown in [85], together with an illustration of how the QoS Profiles and QPTs can be transferred along with handover signalling and the format of messages required in this signalling exchange.

7.2 Time efficient context transfer

Delays that may occur during a context transfer seriously affect those applications that require a timely presence of the context in the new AR. When a MN moves too quickly between ARs, thus resulting in a condition of continued change of the point of attachment to the network, this could lead to a situation where a lot of context transfer signalling information is present in the radio link, without a context transfer being actually performed. The problem is that, before context transfer completion, the MN has changed its point of attachment again. Nakhjiri and Singh [90] present a proposal for achieving a context transfer outside timing critical path for handover.

The proposed framework uses the concept of bi-directional edge tunnels, illustrated in [89], an extension to the fast Mobile Ipv6 handover protocol described in [59] that reduces the amount of layer three handover latency in Mobile IPv6 to zero. The basic idea behind bi-directional tunnels is that a MN can move through AR and receive packets without changing its care of address, thanks to the tunnels established in order to ensure the correct forwarding of the packets. It is up to the MN to decide when obtaining a new care of address (operation that produces a lot of signalling time consuming wireless traffic), based on the fact that real time traffic is absent or the MN is moving slowly enough to be able to obtain the new care of address without affecting real time applications. Describing this method, however, is outside the scope of this thesis; more details can be found in [89].

The Internet Draft [90] seeks to eliminate the context transfer signalling out of the timing critical path of a network layer handover, suggesting to keep the context, after a link layer handover, at the old AR, until the MN begins a layer three handover and adopts the new AR as the one in charge of forwarding its traffic. In the interval of time comprised between a layer two handover and a layer three handover, the old AR must process MN's features.

7.2.1 Protocol overview

The main motivation for time efficient context transfer development is to eliminate the context transfer delay from the critical path of a network layer handover. The revolutionary idea is that the context transfer can be started and completed along with delivery of data traffic and independent of exact timing of handover completion. This way, the proposal ensures continuous traffic forwarding and feature support to applications that require strict timing requirements (but in general to every application).

Before summarizing the steps of the protocol, we introduce useful key concepts:

- Anchor AR (aAR): [90] defines it as the AR that handles the routing of the MN's traffic, ensures support of features associated to MN's applications and has given the MN its current care of address (anchor care of address).
- Context anchor AR (CaAR): It is the AR that holds the MN's context; in general it corresponds to the anchor AR.

A bi-directional tunnel between the aAR and the new AR is established to exchange traffic between the MN and the aAR via the new AR until the MN acquires its new care of address. The MN's context resides at the aAR until the MN has acquired a new care of address and the new AR keeps on forwarding traffic between MN and aAR, regardless of the context information contained in the tunneled packets. Context transfer can begin only when the MN has acquired a new care of address, performed from the aAR to the new AR; the context transfer data can be sent as next header information associated with a flow or in separate packets. When the context transfer is completed the tunnel between aAR and the new AR can be torn down and the new AR can handle routing and support features for the MN’s traffic.

7.2.2 Discussion on time efficient context transfer

There are two main advantages of this approach: one is that, by choosing an opportune context transfer trigger ([90] explains how and when to choose properly the trigger), context transfer can be performed independently by the timing of handover messages. The second is that this proposal reduces the network traffic load due to context transfer, typical of quick movements of the MN between ARs; only when the MN is ready to acquire a new care of address context transfer is done. In fact, in case an MN moves to a third AR before acquiring a care of address, the aAR simply moves the end of the tunnel to this third AR, tearing down the connection with the second AR; the exchange of signaling messages required to perform the change of the tunnel wireless end between is accomplished along the wired network connecting ARs, so that unnecessary traffic on the critical radio link is avoided.

In order to provide a robust and reliable context transfer, a number of factors should be taken in account: fragmentation of context data could be necessary, if the total amount of context data is larger than the MTU of the path for example. When the MN moves to a third AR before acquiring a new care of address, the AR that was in charge of handling the context must abort the context immediately because it does not longer need the context itself and to avoid misunderstanding in the network about the AR that is actually the MN’s aAR.

Since the reliability mechanism of the transport layer can provide retransmissions of data, attention must be paid to the feature contexts that are highly dynamic, such as the header compression states. A retransmission of such a feature context could lead to an obsolete context present at the AR, which instead must always keep the current state associated to the mobile; in such a case, an update of the context would be rather preferable to a retransmission. The proposed protocol foresees these issues allowing to set a bit at the sender AR to distinguish if the context sent after a failure is a copy of the lost one, i.e. a retransmission, or represents an updating of the data.

The described proposal seems suitable to manage the context transfer for real time applications, since it meets the requirements stated in the context transfer requirements specification draft [74]. The protocol supports the usage of the feature context profile types, which facilitates the transfer of various kinds of context (thus also QoS profiles), and the identification of these context data. Moreover, as we noted above, the number of signaling exchanges is minimized, since the MN performs context transfer only when it is necessary, and the problem of fast movements between ARs is solved.

8 Test arrangements

In this chapter we describe the methodology used for the tests we have run: the aim of these tests was to verify that the deployment of context transfer is really helpful for improving the performance of a network in the occurrence of a handover.

8.1 Test methodology

We have executed different kinds of tests: for all of them the common denominator was the transfer of a flow from a source to a recipient through a path loaded by other flows (called background flows) and its redirection along another path. In all the executed tests, which were run in order to observe the effect provoked on a real-time flow by the change of its path to the destination, the whole path was wired and the handover was emulated by changing the routing tables in the network, so that the flows could follow a different path.

To emulate the context transfer, instead, we have supposed to enable the QoS on the path where the flow is routed to; the most interesting results are those relative to a delayed application of context transfer. In such a situation, in fact, it was easy to notice the improvements the context transfer carries out: immediately after handover the flow is treated as best effort, and after some time the QoS is enabled, that is, context transfer is deployed, and the service received is much better. We have realized, though in a wired network, a situation as close as possible to what would happen in a real context transfer occurrence, to evaluate faithfully the performances of this new approach in the complex field of QoS provisioning in a mobility scenario.

The principal aim of these tests was to show the effectiveness of context transfer as a way to provide QoS to real time applications. For this reason, the principal parameters used to measure performance are related to time, the timeliness of packet delivering being the most important factor in such a situation. In order to evaluate the performance, the common guideline in most of the executed tests is the use of the same workload in both the traversed paths: of course this is not a likely situation in a real environment; nevertheless, the only way to evaluate performance is to realize a scenario where the same conditions in both the paths are assured to the analyzed flow. Moreover, we have always chosen as analysed flow a CBR one, being easier to accomplish time statistics on a CBR flow than on a VBR one. In order to evaluate many of the various scenarios that a real implementation can present, we have considered many different situations, analysing each of the implemented DiffServ classes, checking out what happens if the bandwidth reserved for a class is overcome, trying to find a threshold for the deployment of context transfer.

8.2 Experimentation environment

The network configuration adopted, depicted in Figure 24, contains three routers and two workstations. All five machines are Intel Pentium III 1 GHz and run the Linux Operating System (kernel version 2.4.17). The traffic was generated by means of two traffic generators: MGEN [91] for the UDP flows, and JTG, a modification of the TTCP [92] traffic generator by Jukka Manner, at the University of Helsinki, to create TCP data transfer. The advantage to use the MGEN traffic generator is double: on one hand it allows to generate traffic and to assign each flow to the proper DiffServ class, on the other hand, the MGEN package encloses tools that permit to gather information and log packets in order to calculate the various time parameters and effect the desired statistics.

All the links in this configuration, except that between Kemi-4 and Kemi-12 are 10Base-T Ethernet links belonging to an isolated LAN and they simulate the wired network immediately after a radio access network. The link between Kemi-4 and Kemi-12 is a 100Base-T Ethernet link and it simulates a high-speed network trunk, in order to create a situation of congestion at the bottleneck router, Kemi-12.

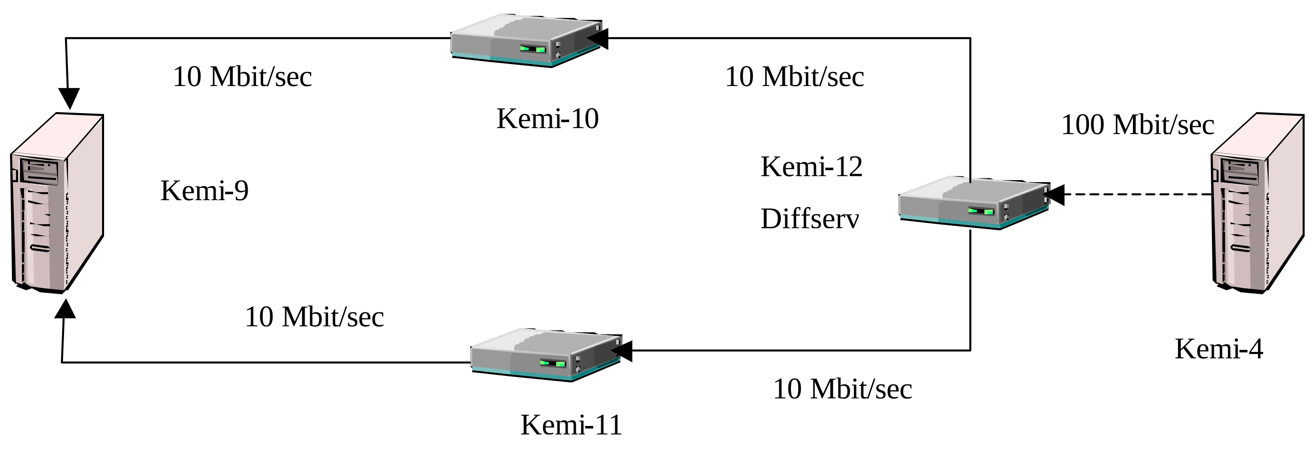


Figure 24: Emulation environment

With regard to the QoS architecture deployed, we have chosen the DiffServ one, implemented via the Dsmark queuing discipline [93]; in the configuration adopted, only the bottleneck router, Kemi-12, was DiffServ enabled by means of the TC tool [94]. According to the generated workload, the traffic entering the edge router is scheduled into four classes: EF, AF 1.1, AF 2.3, and BE. A simple PRIO qdisc separates the EF first priority class from the others; the EF class is managed via a token bucket filter. A class-based queuing discipline is instead used to separate traffic within AFs and BE classes; within these classes, traffic is handled via GRED qdiscs for the AFs traffic and via a RED for the BE one. Figure 25 describes the TC model of the queuing discipline in the DiffServ node.

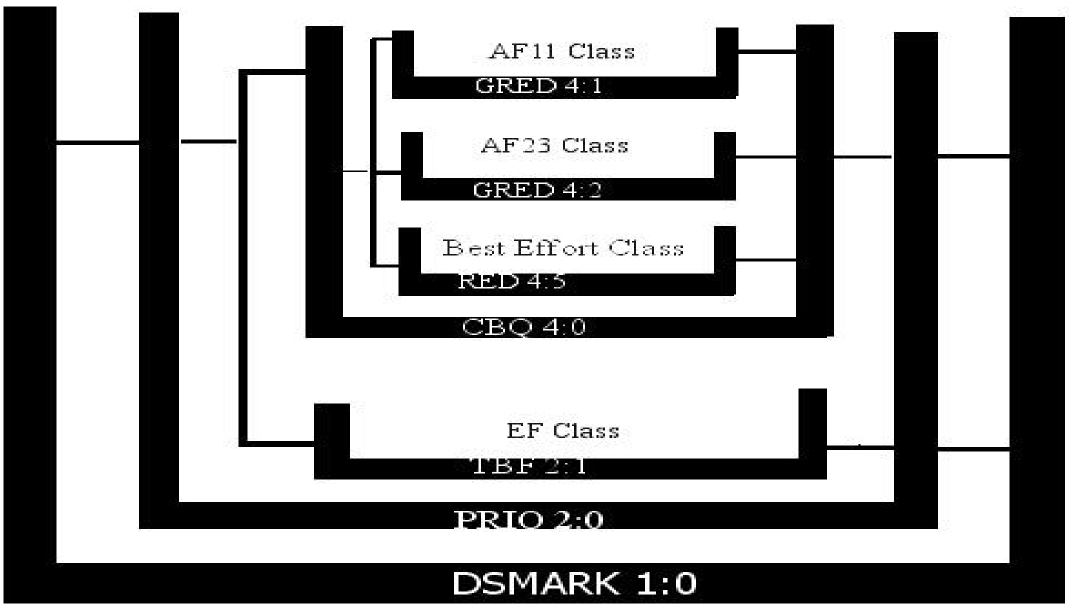


Figure 25: Queuing discipline in the DiffServ node [98]

The total available bandwidth, 10 Mbit/sec, was divided statically among the classes; an example of such an allocation could be the following: 1 Mbit/sec for EF, 1 Mbit/sec for AF 1.1, 5.5 Mbit/sec for AF 2.3, 2.5 Mbit/sec for BE. This choice is due to our decision to assign higher rate flows (512 Kbit/sec) to the AF 2.3 class, but any allocation that does not exceed the limit of 10 Mbit/sec in total is feasible.

8.3 Utilised metrics

Fundamentally, the two most important telecommunication standard organizations have defined two different QoS parameters schemes: IETF IP Performance Metric (IPPM) and ITU-T proposal. The two entities essentially agree on the list of parameters that can be used to gauge the performance of an IP link and hence its quality, but they use similar or identical acronyms with different meanings. The most significant difference is that ITU-T proposes a statistical definition of QoS parameters, while IETF is willing to provide a precise measurement procedure for each parameter. The agreed parameters for IP Performance QoS are one-way delay, instantaneous packet delay variation (IPDV), bandwidth and packet loss.

In our experiments we used a metrics system composed of the above-mentioned and other parameters in order to extrapolate the key parameters for each kind of traffic; the two most important parameters for application that have stringent QoS demands are delay and IPDV, and for these we report a standard definition.

One-way delay is defined formally in RFC 2679 [95]. This metric is measured from the wire time of the packet arriving on the link observed by the sender to the wire time of the last bit of the packet observed by the receiver. The difference of these two values is the one-way delay. The delay we are referring to is an application delay, calculated as the difference between the time the packet has been sent and the time when it has been received by the application, supposed the clocks at the end sides are synchronised, thanks to NTP protocol [96], running at the end hosts.

The application delay referred to above is comprised of the network transmission delay, typical of the particular analysed network and of the queuing delay due to buffers, scheduling and packet elaborations at intermediate routers; the application delay is the most significant one to take into account since it is the actual delay experienced at the receiver. A multimedia application requires that the variation of the delay with which packets are received is as small as possible; for such a reason, the mean and the variance of the delay have been calculated each time and the value again averaged out based on the total number of repetitions run.

Another important parameter is the *"instantaneous packet delay variation (IPDV)",* formally defined by the IETF IP Performance metric (IPPM) working group draft in [97] as the difference between the one-way delays of pairs of consecutive packets at the receiver. If we let D_i be the one-way delay of the i^{th} packet, then the IPDV of the packet pair is defined as $D_i - D_{i-1}$. Based on the common usage, IPDV is also known as jitter and is computed according to the following formula:

$$\text{IPDV-jitter} = |\,\text{IPDV}|$$

The IPDV is itself a measure of the variation of the delay from a constant value; we calculate and average out the mean and the variance of this parameter.

We have also computed, for both delay and IPDV, the 95-percentile of their values. This parameter represents the maximum value assumed by either delay or IPDV without considering the 5% of the total higher values. In other words, if we have 100 values of a particular analysed quantity, say from 1 to 100, the 95-percentile of this quantity would be 95. To calculate it, we order all the values assumed by a quantity in a test repetition, we discard the 5% higher ones, and then we consider the new maximum value. The introduction of this parameter is necessary since sometimes we have experienced delay and jitter values much higher than the rest of the repetition. This is because the Linux operating system is not dedicated to the only task of packet collecting and logging. Therefore, sometimes it can happen that for reasons internal to the operating system own principle of working, an operation on a packet (delivering to an application, releasing from buffer and so on) can be delayed so that the operating system can perform other tasks, obtaining a value not coincidental with reality. By eliminating the 5% higher values, we obtain an idea of the one that is likely to be the maximum real value for that quantity. Moreover, the percentile gives an idea of the wideness of the distributions of the values for a parameter, the closer it is to the average value, the less the values are spread.

Finally, we compute the distribution of the values assumed by the delay and the IPDV during the tests. We obtain graphics that show how the possible values assumed by these parameters are distributed. A wide distribution of values indicates that their variations, for the analysed parameter, are remarkable; we expect to find out narrow graphics if referred to a lightly loaded network or without QoS enabling, wider graphics vice versa. In these graphics, on the x axis are plotted all the values that the analysed parameter has assumed during the test, divided in intervals of size (step) most of the cases set to 0.1 msec. On the y axis are reproduced (in log scale for clarity) the number of times that the analysed parameter assumed a value comprised in the relative interval on the x axis. As an example, if we know that the maximum value assumed by a parameter is, say, 60 msec, the x axis will be divided in 60/step-size intervals.

8.4 Test cases description

We have run several test cases, analysing different network configurations and collecting results for all the DiffServ classes present in our configuration, to have a wide and the most complete scenario possible of the situations that may occur in a real network. The real-time and best-effort traffic that was used in the measurements was generated, as written above, by using MGEN (version 3.3a6) and another simple workload generator: JTG. All UDP flows were generated by MGEN, which generates real-time traffic patterns to unicast and multicast IP destinations over time according to a script file. Packet sizes and transmission rates for individual information flows can be controlled and varied and also the TOS byte can be set up using MGEN's script file.

The real time data emulated IP phone calls and IP video streams. The characteristics of the IP phone streams were collected from IP phone measurements and two different voice codecs were considered: ADPCM and GSM. The speeds of the video streams were various and they represent bursty streams of different quality depending on the rate. Best-effort traffic was composed of a TCP bulk transfer and a variable (depending on the test case) number of UDP flows without any reservation generating the congestion situation. The traffic characteristics are summarized in Table 5, where HQ means high quality while MQ means medium quality. The packet size indicates the UDP packet payload size for UDP flows and the buffer’s length of TCP bulk transfer. The packet rate and data rate parameters for VBR flows indicate the average transmitted packet rate and the average transmitted data rate of a Poisson process packet generation. A complex workload was chosen with several kinds of flow ranging in a set of different packet sizes and packet rates in order to examine network behavior in quite a realistic situation.

STREAM	Packet size (bytes)	Protocol	Bit-rate type	Packet rate (packets/sec)	Data rate (kbit/sec)	Number of buffers
VoIP (adpcm)	104	UDP	CBR	50	41.6	
VoIP (gsm)	36	UDP	CBR	50	14.4	
Video (HQ)	320	UDP	VBR	200	512	
Video (HQ)	320	UDP	CBR	200	512	
Video (MQ)	500	UDP	CBR	40	160	
BE	320	UDP	VBR	200	512	
BE	1024	TCP				20000

Table 5: Generated flows characteristics

With reference to Figure 24, in all the executed tests it has been supposed that there is the transfer of a multimedia flow (UDP) from Kemi-4 to Kemi-9, lasting 120 seconds. During the first 60 seconds of the experiment, the flow follows the path Kemi-4 > Kemi-12 > Kemi-10 > Kemi-9 (also called in the remainder of the thesis "path A"). Afterwards, the handover happens and the flow is redirected along the path Kemi-4 > Kemi-12 > Kemi-11 > Kemi-9 ("path B"). The two paths are loaded in a different way, and have QoS enabled, depending on the test case.

Table 6 below summarizes the characteristics of all the executed tests. The column "QoS enabling" indicates whether the DiffServ architecture is enabled in that test case and when; the column "workload characteristics" describes briefly the workload used in the test case and finally, the column "principal results" underlines the main result obtained for the principal flow in that case.

In the remainder of the thesis, we will use the term principal flow, or equivalently analysed flow, to refer to the only one, directed to Kemi-9, for which we calculate the statistics and trace graphics. The relative column thus refers to the features of the principal flow in that test case. The DiffServ class indicated for the analysed flow refers only to the periods when QoS is enabled for this flow. For further particulars, we refer the reader forward to the description of the single cases.

Test case	Analysed flow	QoS enabling	Workload characteristics	Principal results
A1	CBR 14.4 Kbit/sec	All best effort network	UDP network highly loaded after ho	Performance worsening
A2	CBR 14.4 Kbit/sec EF class	Only before handover	All UDP lightly loaded network	Worsening of IPDV
A3	CBR 14.4 Kbit/sec EF class	Before and after handover	All UDP lightly loaded network	No sensible variations
A4	CBR 14.4 Kbit/sec EF class	Delayed context transfer	All UDP lightly loaded network	Co. tra. improves performances
B1	CBR 512 Kbit/sec EF class	An. flow before ho others always	All UDP network Limit conditions	High performance worsening
B2	CBR 512 Kbit/sec EF class	Before and after handover	All UDP network Limit conditions	No sensible variations
B3	CBR 512 Kbit/sec EF class	Delayed context transfer	All UDP network Limit conditions	Co. tra. improves performances
C1	CBR 160 Kbit/sec	All best effort network	UDP + TCP network highly loaded after handover	Performance worsening TCP increases variance
C2	CBR 160 Kbit/sec AF 1.1 class	An. flow before ho others always	UDP + TCP flows Limit conditions	Like case C1
C3	CBR 160 Kbit/sec AF 1.1 class	Before and after handover	UDP + TCP flows Limit conditions	No sensible variations
C4	CBR 160 Kbit/sec AF 1.1 class	Delayed context transfer	UDP + TCP flows Limit conditions	Co. tra. improves performances
D1	CBR 512 Kbit/sec	All best effort network	UDP + TCP network highly loaded after handover	Performance worsening TCP increases variance
D2	CBR 512 Kbit/sec AF 2.3 class	Delayed context transfer	UDP Bandwidth not overcome	Co. tra. improves performances
D3	CBR 512 Kbit/sec AF 2.3 class	Delayed co. tra. bounded case	UDP Bandwidth overcome	Overcoming is harmful
D4	CBR 512 Kbit/sec EF-AF 2.3 classes	Delayed context transfer	All UDP change in DiffServ class	EF better than AF 2.3
E1	CBR 512 Kbit/sec AF 2.3 class	Delayed co. tra. unbounded case	UDP Bandwidth overcome	Trade-off bounded and unbounded case
E2	CBR 512 Kbit/sec AF 2.3 class	Variously delayed context transfer	Referred to test case D2 and D3	Threshold comparisons
F	TCP BE stream	Only for UDP flows	All UDP network + 1 TCP flow	Discussion on TCP flow

Table 6: Test cases characteristics summary table

9 Test result

This chapter presents the results obtained for the tests introduced above. We have collected them by monitoring the principal flow at the destination, keeping the network loaded by the background flows. All the results were obtained by manipulating the log files provided by DREC and MCALC, tools included in the MGEN package. For each test case, at least 10 repetitions have been run, and then the results averaged. The graphics are traced based on a single repetition, while the sum-up tables present values averaged on all the repetitions executed for that test case and pertinent to the repetition analysed. For reasons of brevity, we study in detail only the most interesting test cases, though we have collected tables and graphics for all the cases in Table 6.

9.1 Test case A2

The characteristics of the workload utilised in this test case are summed up in Table 7 below:

Stream	Packet size (bytes)	Protocol	Packet rate (pkts/sec)	Data Rate (Kbit/sec)	Number of flows	Recipient	DiffServ class
Video	320	UDP/CBR	200	512	14	Kemi-10/11	AF 2.3
VoIP (ADPCM)	104	UDP/CBR	50	41.6	3	Kemi-10/11	AF 1.1
VoIP (GSM)	36	UDP/CBR	50	14.4	7	Kemi-10/11	EF
VoIP (GSM)	36	UDP/CBR	50	14.4	principal flow	Kemi-9	EF

Table 7: Workload characteristics, test case A2

In this test case, the principal flow was a CBR low bit rate VoIP stream and the background flows properly selected to create a lightly loaded network. The QoS is instantiated only before handover; this means that we are describing a situation where no context transfer is deployed. After the handover, all the flows are treated as best effort; anyway, to better evaluate performances, both path A and path B are equally loaded besides the fact that in the path A QoS is enabled. The DiffServ class indicated in Table 7 is valid obviously only when the QoS is enabled.

In Figure 26 the trend of delay and IPDV over time are represented. It is possible to see that the improvements that the deployment of DiffServ carries out in a lightly loaded network are not so significant. The delay after handover has slightly increased, since, as shown in [98], the performance of DiffServ are not remarkable in such a situation. Rather, and always according to [98], the IPDV after handover is noticeably higher. In any case, both the values of jitter and delay are under the sending rate of 1 packet every 20 msec, which is acceptable and predictable as the network is not close to overload conditions. By the way, Figure 27 is added to compare the trend of delay in a case of totally best effort configuration. It is impossible to notice any significant difference after handover (60 sec) and moreover, the variance of delay before has increased, which shows that the introduction of QoS improves the performance with regard to IPDV.

The spikes localised at handover time are present in all the executed tests; they are due to the fact, underlined above, that the Linux operating system is not dedicated to packet managing. Handover involves many processes, for example to change routing tables, so that the operation of packet delivering may have been delayed during its execution. The information that can be recovered by the 95 percentile is a good way to overcome the misleading values that can be obtained computing all the values of delay or jitter.

It is possible to notice, looking at the value distributions for delay and IPDV in Figure 28 how the graphics are spread after handover. Even if this cannot seem evident, it must be considered that for reasons of visual clarity the scales of the x axis are different from case to case, and the maximum value after handover is always higher than the one before. The maximum number of occurrences for the delay before handover is close to 500, in correspondence of a delay value of about 6 msec. This is not the average value (from Table 8 it is 4,463 msec), but we can see that the highest concentration of values is around this value. This is a general issue valid for all the test cases, when the QoS is enabled the maximum number of occurrences is higher and closer to the average value. After handover, this number drops to 200, proof of the major wideness of the experienced values.

Table 8 summarizes all the relevant metrics for the analysed repetition. There are no packet losses (and this happen always), since the network is not overloaded. As is evident from the figures, the major worsening is noticeable for the IPDV values, while the average delay, and its variance, are quite constant.

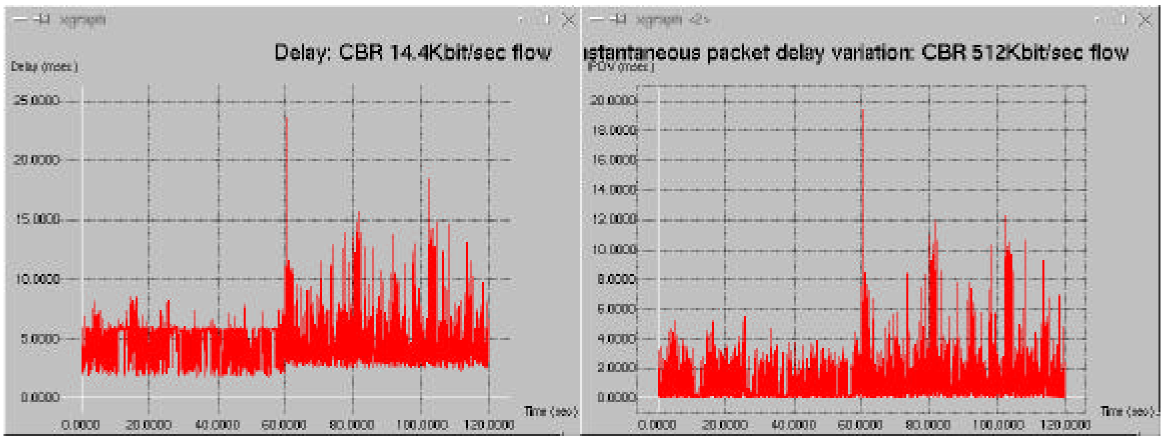


Figure 26: Delay and IPDV, test case A2

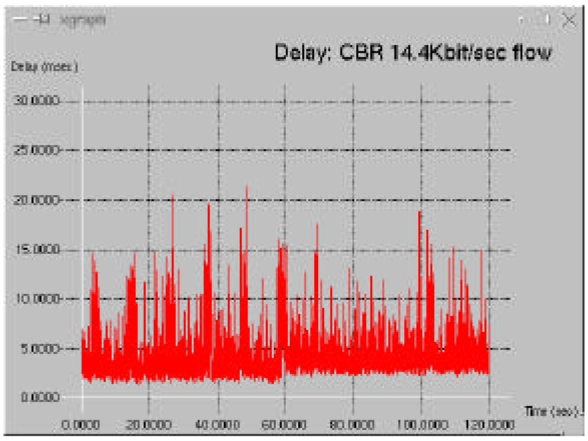


Figure 27: Delay in best effort configuration

Cbr flow	Results: rep.1		
	Before ho	**After ho**	**Total**
Packets received	3000	3000	6000
Packet drops	0	0	0
Average delay (msec)	4,463	4,704	4,583
Delay variance (msec)	1,291	1,465	
Average IPDV (msec)	0,874	1,489	1,181
IPDV variance (msec)	0,7355	1,23	
Delay 95-percentile (msec)	6,047	9,62	
IPDV 95-percentile (msec)	2.805	5.668	

Table 8: Metrics values, analysed repetition

CBR FLOW	RESULTS: REP.1		
	Before ho	**After ho**	**Total**
Packets received	3000	3000	6000
Packet drops	0	0	0
Average delay (msec)	4,292	4,792	4,542
Delay variance (msec)	1,292	1,664	
Average IPDV (msec)	0,878	1,548	1,213
IPDV variance (msec)	0,741	1,261	
Delay 95-percentile (msec)	5,887	10,137	
IPDV 95-percentile (msec)	2,808	5,568	

Table 9: Average values test case A2

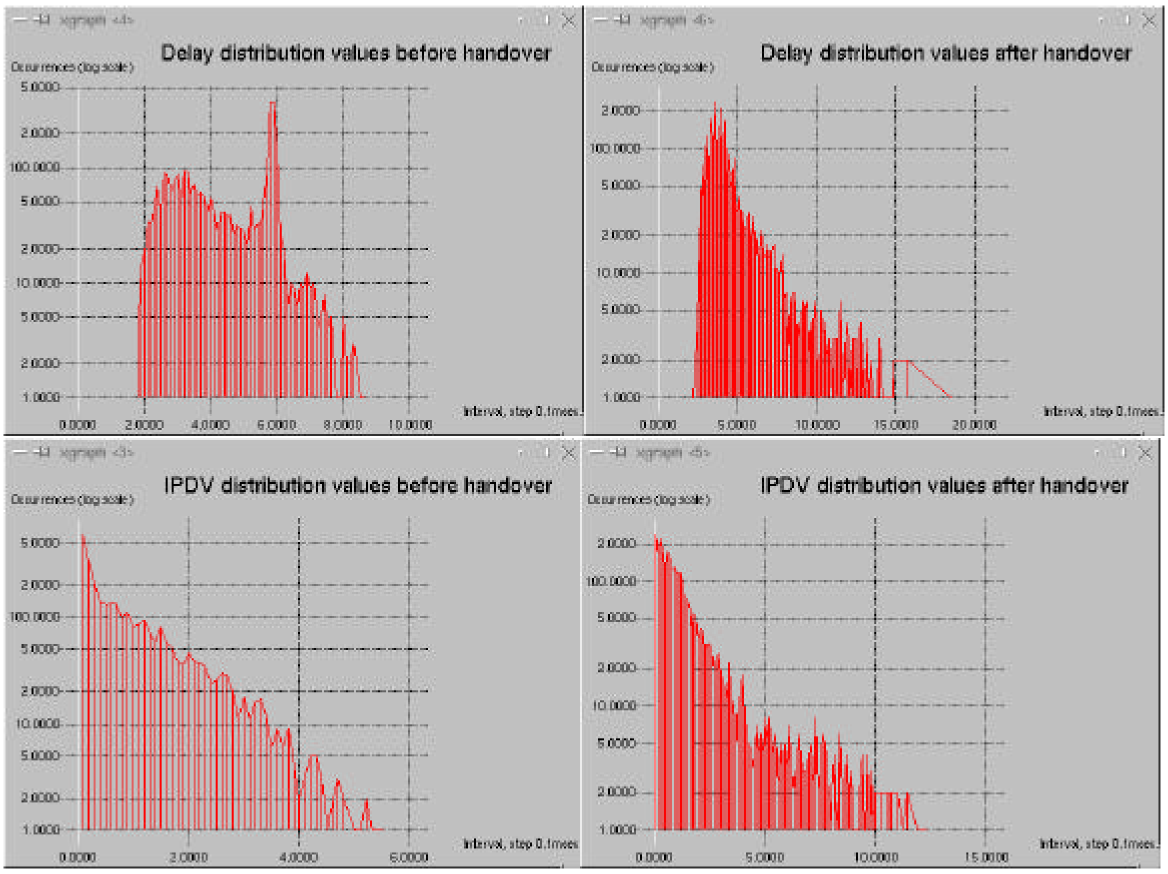


Figure 28: Distribution of delay and IPDV, case A2

9.2 Test case B1

The workload utilised in the test case B1 is the same that Table 7 shows, with the only difference that the principal flow is a high quality video stream of 512 Kbit/sec CBR. In this test case, the high quality video flow is again re-routed along a path where it no more receives the QoS before provided, but we have enabled the QoS on the new path for all the background flows, so that only the principal flow is treated as best effort after handover. This is an even worse situation with respect to that examined in the previous test, when QoS on the new path was not enabled and all the flows received a best effort service. The results found confirm that the worsening of performances in this case is much more sensible than in the case of handover towards an all-best effort network.

In this sense, Figure 29 is quite explicative. The delay after handover increases enormously and passes from an acceptable average value of 5 msec to that of 32 msec with peaks up to almost 200 msec. Similar considerations are valid for the IPDV too.

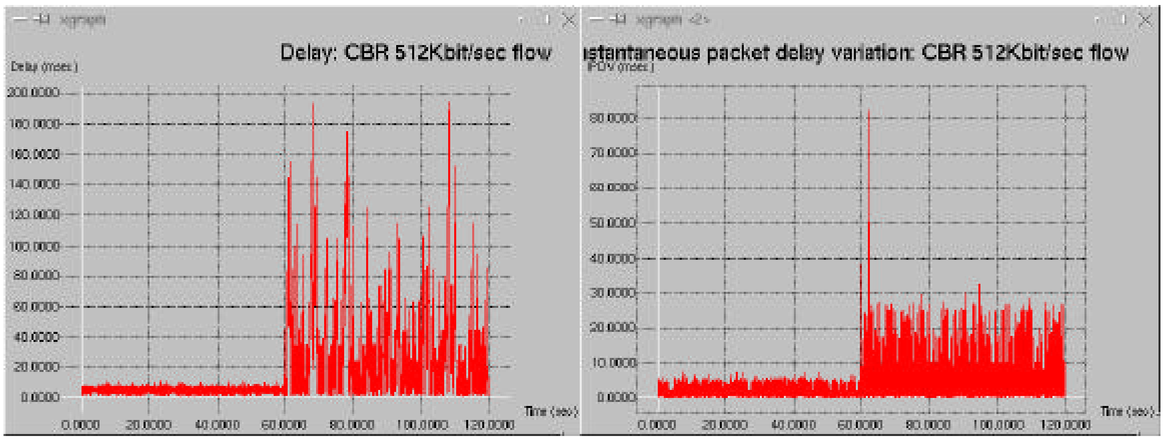


Figure 29: Delay and IPDV, test case B1

The graphics in Figure 30 show the distribution of the values; it is to consider that the scale of x axis is different before and after handover, for reasons of visual clarity. The upper left graphic shows that the delay distribution before handover is quite narrow, as the values range in an interval of about 9 msec, while after handover the same interval is wide almost 200 msec. These considerations find a confirmation in Table 10: delay variance after handover is much higher than before. Moreover, the percentile is much closer to the average before handover, further proof that values are more concentrated around the average.

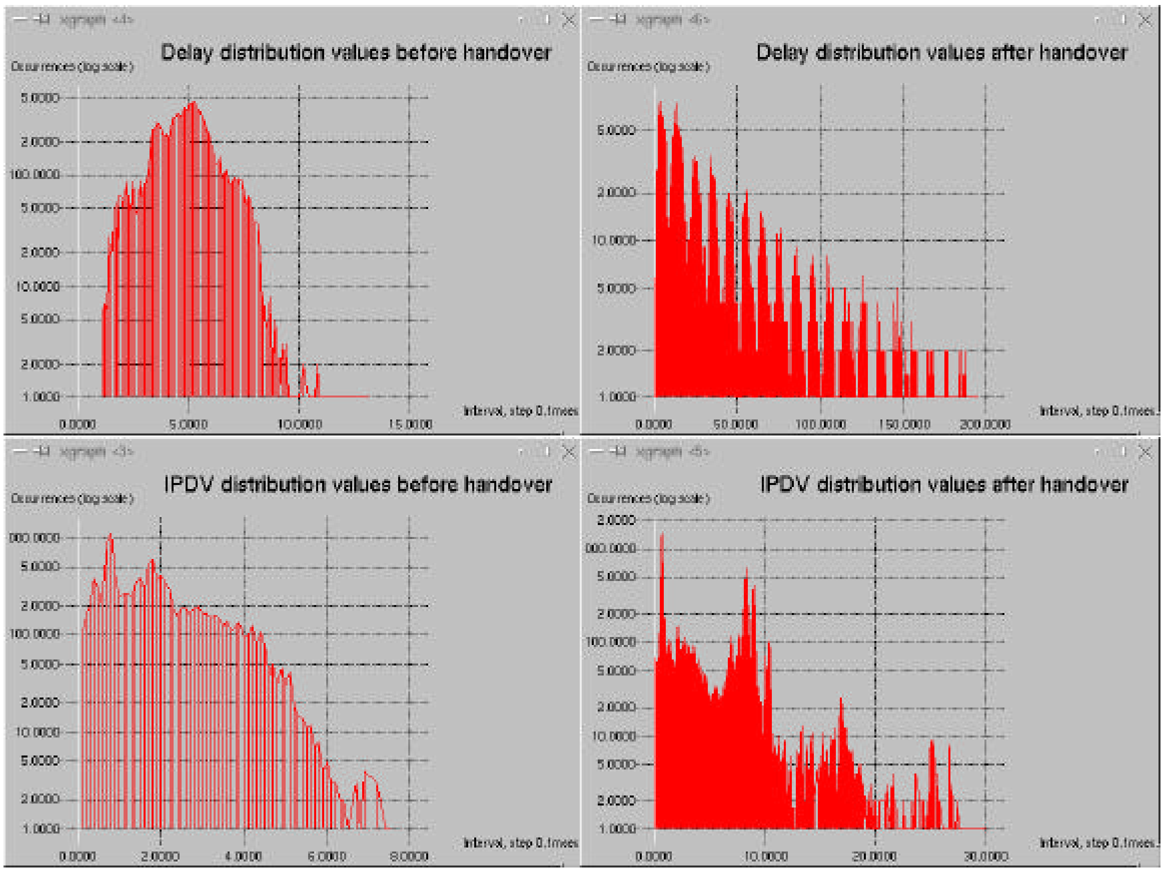


Figure 30: Delay and IPDV distribution values, test case B1

Cbr flow	Results: rep.4		
	Before ho	After ho	Total
Packets received	12000	12000	24000
Packet drops	0	0	0
Average delay (msec)	4,838	31,854	18,346
Delay variance (msec)	1,062	24,655	
Average IPDV (msec)	1,829	5,068	3,449
IPDV variance (msec)	0,991	3,825	
Delay 95-percentile (msec)	7,231	104,709	
IPDV 95-percentile (msec)	4,241	11,131	

Table 10: Metrics values, analysed repetition

Cbr flow	Average results		
	Before ho	After ho	Total
Packets received	12000	11961	23961
Packet drops	0	9	9
Average delay (msec)	4,884	41,107	22,996
Delay variance (msec)	1,030	35,717	
Average IPDV (msec)	1,674	5,116	3,395
IPDV variance (msec)	0,919	3,851	
Delay 95-percentile (msec)	7,133	148,345	
IPDV 95-percentile (msec)	3,954	10,773	

Table 11: Average values, test case B1

The analysed one is the only repetition where no packet drops have been experienced; in all the others, few packets have been dropped after handover. The results show how the IPDV increases slightly, even after handover, while the delay becomes significant. The average value is a magnitude order higher and the percentile even two, due to the spikes that can be noticed in Figure 29. It has been chosen, as repetition to be plotted, the best case; in fact, the peaks of delay in the worst cases arrived up to 350 msec.

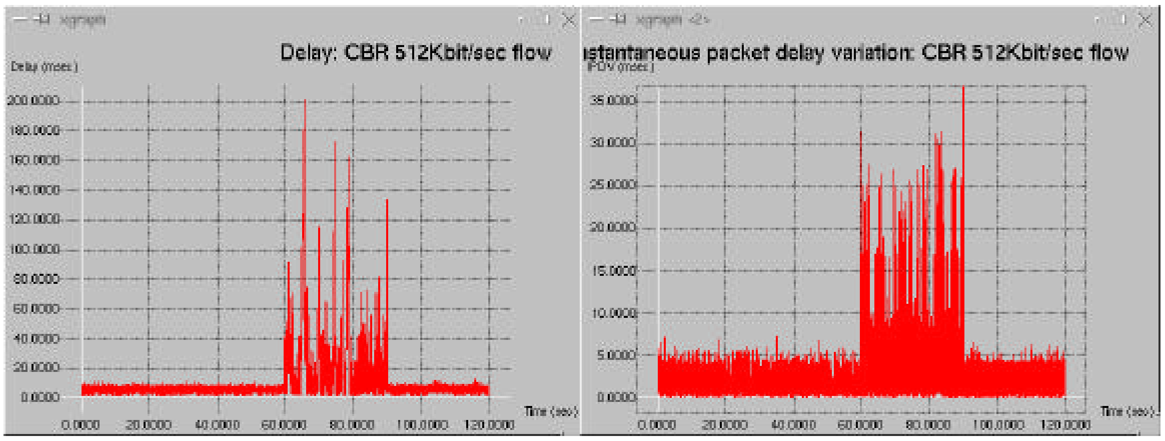


Figure 31: Delay and IPDV, test case B3

We include Figure 31, relative to test case B3, to show how context transfer can improve performance and help in QoS achieving. The test cases D* and following will be based on the delayed context transfer, so for now we include just this figure. After handover, in fact, the principal flow is treated as best effort and the background ones receive QoS, thus resulting in a situation similar to the one described in test case B1, but 30 seconds after handover (90 after the beginning of the test), the principal flow begins to receive QoS. The advantages that context transfer produces are evident, in terms of reduction of delay and IPDV average values and their variance.

9.3 Test case C2

The main difference between this test case and the previous lays in the fact that now the examined flow is a CBR medium rate (typical for example in a medium quality video) with high packet size, 500 bytes, however common for this kind of applications. Moreover, the workload has been slightly modified; TCP traffic has been added, and the number of VBR flows reduced. Finally, the principal flow has been assigned to the AF 1.1 class, instead of EF class, to test the performance of this class. As well as test B1, the QoS is instantiated for all the flows of the workload on path A, and for all but the principal one on path B after handover. The characteristics of the workload utilised in test case "C2" are summed up in the following table:

Stream	**Packet size (bytes)**	**Protocol**	**Packet rate (pkts/sec)**	**Data Rate (Kbit/sec)**	**Number of flows**	**Recipient**	**DiffServ class**
Video (MQ)	500	UDP/CBR	40	160	Principal flow	Kemi-9	AF 1.1
VoIP (ADPCM)	104	UDP/CBR	50	41.6	6	Kemi-10/11	AF 1.1
VoIP (GSM)	36	UDP/CBR	50	14.4	7	Kemi-10/11	EF
Video (HQ)	320	UDP/VBR	200	512	11	Kemi-10/11	AF 2.3
Data	1024	TCP	15000 buffers sent		1	Kemi-10/11	BE

Table 12: Workload characteristics, test case C2

The introduction of TCP traffic, assigned to a best effort class, has provoked a significant increase in the delay variation, according to the results obtained by [98]. Preliminary tests have been run and they prove that without TCP traffic these variations reduce significantly. The performance of the network are seriously influenced by the TCP flow, as Figure 32 shows. In fact, the delay has significantly increased (such as in case B1, without TCP flow), moreover its variance and mostly the IPDV have become very high. For IPDV in example, several peaks higher than 120 msec in the analysed repetition have been experienced; such a thing is obviously intolerable for delay-sensitive applications.

For this reason, we have executed other tests where the TCP flow is terminated before the end of the repetition, and the network becomes lightly loaded, being unchanged the rest of workload. The results show that the experienced delay, plotted in Figure 33, is not only drastically reduced when the TCP flow finishes (about 90 seconds), but also that the performance of the network in best effort configuration and in conditions of light load, is better than the DiffServ configuration. A good explanation might be that the FIFO queuing applied to best effort flows in light loaded networks, is able to handle this kind of traffic, and is faster than DiffServ, which, with its scheduling mechanisms, requiring processing at routers, wastes unnecessary time for operations. Anyway, in the final tests the duration of TCP flow, selected by choosing the amount of data to send, is greater than the duration of UDP flows, 120 seconds, in order to analyse the transfer of a UDP flow entirely affected by the presence of a TCP data stream. It is possible to see how delay variation is less when the TCP flow finishes, even if the flow is treated as best effort, since after the termination of the flow the network is lightly loaded (the resulting workload is the same as that reported in Table 12 without the TCP flow).

An explanation to the apparently puzzling matter of the TCP influencing UDP flows is that even if TCP adapts its sending rate to network conditions, in any case it does not transmit as well as a CBR flow, causing an increase in the values of IPDV. According to this reasoning also VBR flows should seriously affect IPDV, but statistically the effect of many VBR flows can be compared to that relative to a CBR flow.

The distribution values graphics are quite significant; the distribution for delay before handover has values ranging up to 20 msec. This is due to the presence of the TCP flow that contributes to increase the delay variation, and thus the jitter. Moreover, as it is possible to notice from the bottom graphics in Figure 34, IPDV values are exceptionally spread if compared to test case B1, because of the presence of TCP. Before handover, the maximum value is about 20 msec and after it grows up to more than 100 msec. Note that to not make the graphic too spread, the highest values for IPDV (more than 120 msec) have been cut off. The summarizing tables confirm this analysis: the IPDV after handover has exceptionally increased, mostly if compared to the values of test case B1.

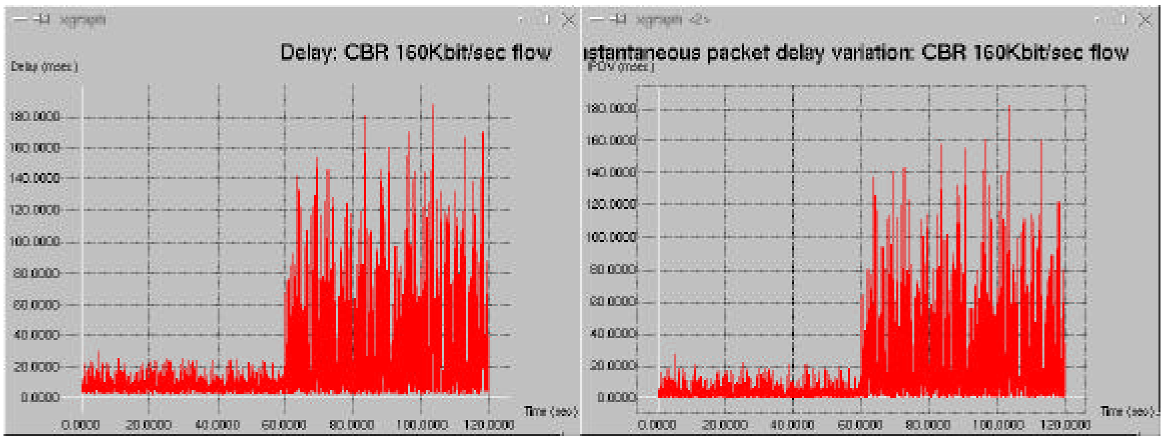


Figure 32: Delay and IPDV values, test case C2

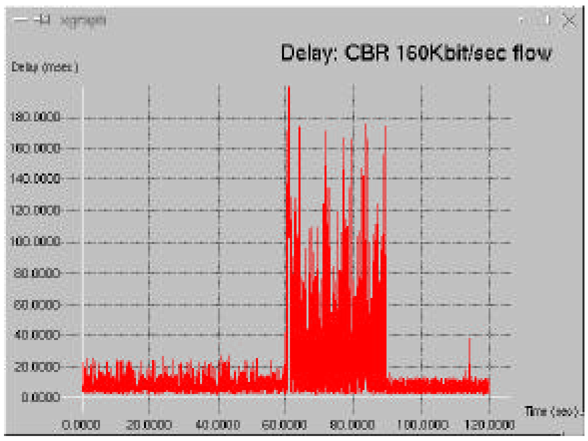


Figure 33: Delay, end of TCP flow at 90 seconds

Cbr flow	**Results: rep.6**		
	Before ho	**After ho**	**Total**
Packets received	2400	2386	4786
Packet drops	0	24	24
Average delay (msec)	7,376	34,184	20,78
Delay variance (msec)	2,534	28,690	
Average IPDV (msec)	3,757	20,149	11,953
IPDV variance (msec)	2,766	15,230	
Delay 95-percentile (msec)	15,659	109,052	
IPDV 95-percentile (msec)	12,865	73,488	

Table 13: Metric values, analysed repetition

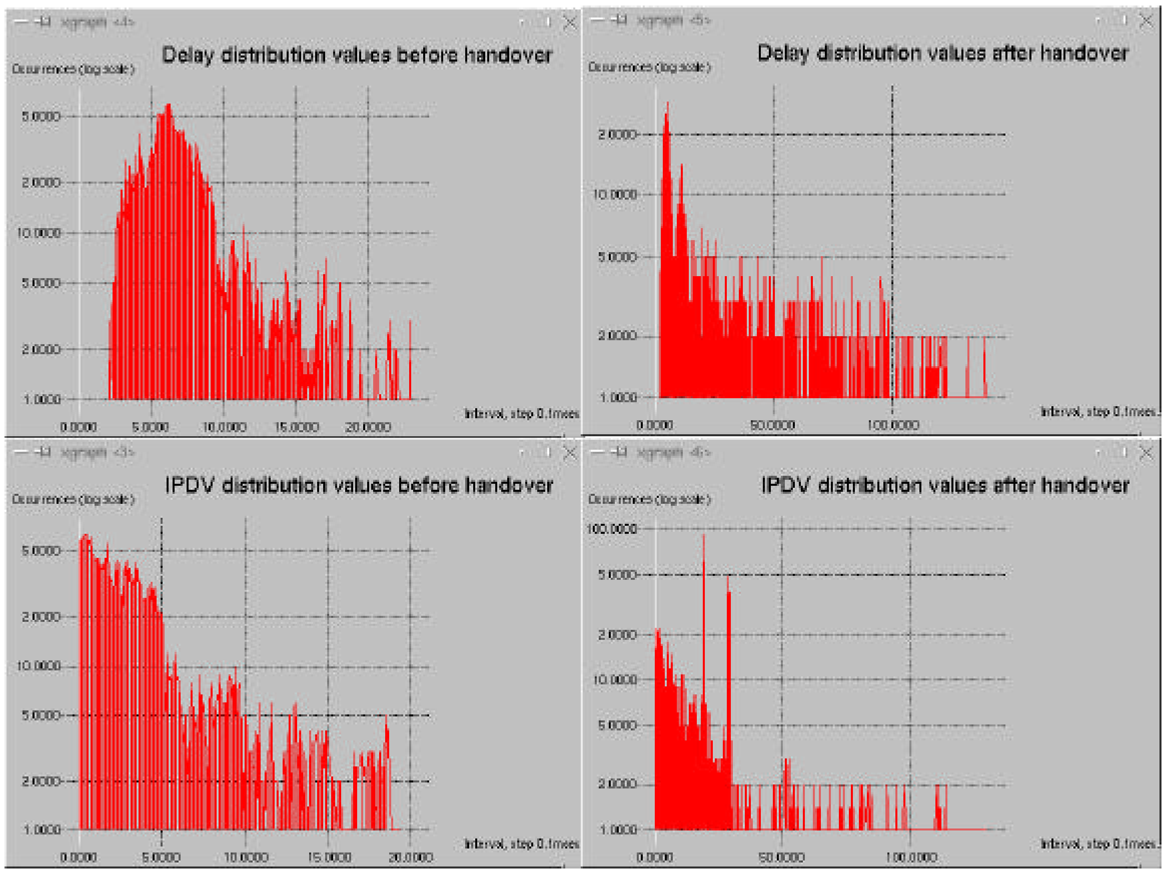


Figure 34: Delay and IPDV distribution values, test case C2

Cbr flow	**Average results**		
	Before ho	**After ho**	**Total**
Packets received	2400	2385	4785
Packet drops	0	15	15
Average delay (msec)	8,216	34,700	21,458
Delay variance (msec)	2,712	29,211	
Average IPDV (msec)	3,875	20,078	11,976
IPDV variance (msec)	2,890	15,330	
Delay 95-percentile (msec)	16,568	109,837	
IPDV 95-percentile (msec)	13,149	74,387	

Table 14: Average results, test case C2

9.4 Test case D2: not-congested DiffServ class

All the test cases belonging to the D group have in common that the principal flow is a video high quality stream, 512 Kbit/sec, CBR, assigned to the DiffServ class AF 2.3. There is QoS instantiated for all the background flows before and after handover, while the principal flow receives QoS before, after handover for 20 seconds is treated as best effort and then is re-assigned to the AF 2.3 class. We have definitely simulated a case of delayed context transfer. The delay of context transfer execution is high to better perform evaluation studies, but test cases E2 has been run to show the effectiveness of an early deployment of context transfer. Unlike the cases discussed till now, the results obtained are showed divided into three parts. They refer to the period of time before handover (0-60 seconds), to that comprised between the executions of handover and context transfer (60-80 seconds) and finally to the period after context transfer (80-120 seconds), in order to better compare network performance in the three different periods of interest.

The characteristics of the utilised workload in the test cases D* are summed up in the following table; in particular, in test case D2, 10 video HQ flows, VBR have been assigned to the AF 2.3 class, and 4 to the BE class.

Stream	**Packet size (bytes)**	**Protocol**	**Packet rate (pkts/sec)**	**Data Rate (Kbit/sec)**	**Number of flows**	**Recipient**	**DiffServ class**
Video (HQ)	320	UDP/CBR	200	512	Principal flow	Kemi-9	AF 2.3
VoIP (ADPCM)	104	UDP/CBR	50	41.6	4	Kemi-10/11	AF 1.1
VoIP (GSM)	36	UDP/CBR	50	14.4	7	Kemi-10/11	EF
Video (HQ)	320	UDP/VBR	200	512	variable	Kemi-10/11	AF 2.3 BE

Table 15: Workload characteristics, test cases D*, E*

The bandwidth reserved for the AF 2.3 class is 5.5 Mbit/sec: as noticed above, there are 10 flows VBR, with average rate 512 kbit/sec belonging to this class. To these flows, the principal one has to be added, resulting in a total amount of bandwidth consumed by the AF 2.3 class higher than 5.5 Mbit/sec.

Anyway, no problems have been encountered since the AF 2.3 class borrows the needed bandwidth from other classes having the same or lower priority; in our particular DiffServ configuration, thus, bandwidth can be shared by the AF 2.3 class only with the BE one. This behaviour is the reason for the particular workload utilised in test case D3, when we have overloaded the AF 2.3 class, as we will explain later on.

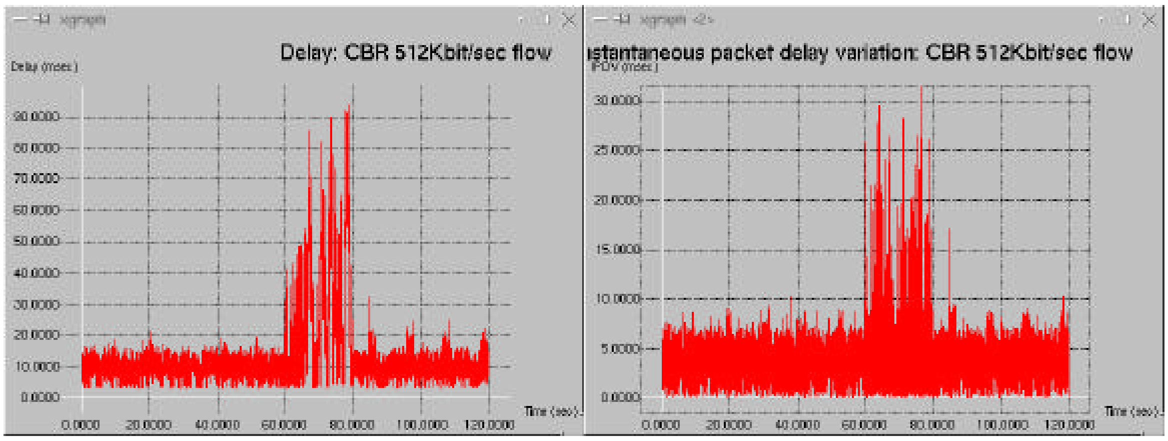
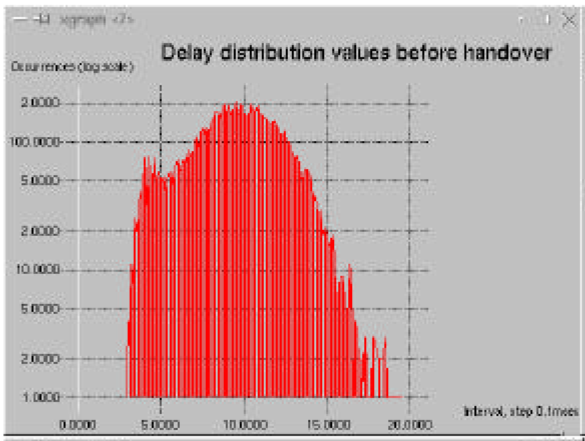
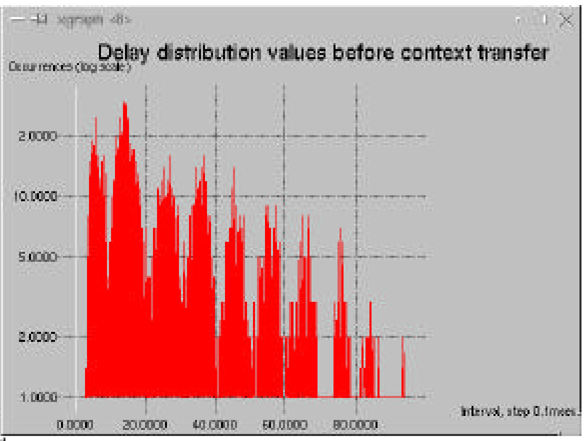


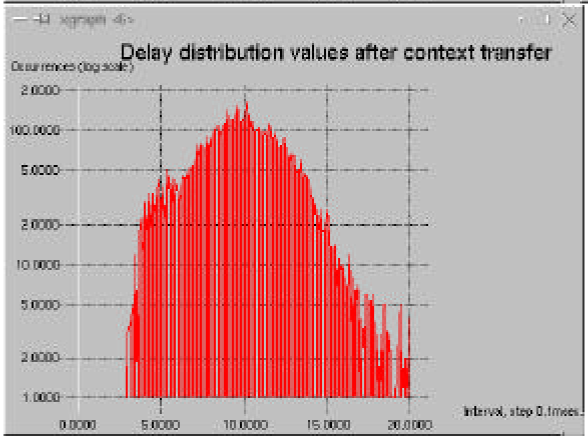
Figure 35: Delay and IPDV, test case D2

As Figure 35 shows, the context transfer happens at 80 seconds and the improvements that it carries out are evident, both in terms of delay and of IPDV average value and variations diminution. Delay, significantly reduces, passing from an average value of 28 msec before context transfer to about 10 after, comparable with the results before handover. Delay variance passes from 16 msec to about 2. Similar consideration are valid for IPDV too.

The trend of delay distribution values, as can be seen from Figure 36, is quite similar before handover and after context transfer, both as wideness of the assumed values and as maximum number of occurrences. Before handover, this maximum number is slightly higher, but it has to be considered that the period before handover lasts 20 seconds more than the one after context transfer, and thus there are more values registered. Moreover, as foreseeable, these two graphs are altogether much less spread than the one before context transfer, being the maximum value in the x axis 20 msec against 90 msec. Similar consideration about the distribution values can be carried out for the IPDV and thus we omit the pertinent graphics. The maximum number of occurrences passes from a value of about 200 (and in correspondence of the respective average values) before handover and after context transfer to about 30 before context transfer. Moreover, before context transfer it is not possible to match neither the maximum number of occurrences nor at least a significant concentration of values around the average, just because of the wideness of the experienced delays.







Looking at the upper right-hand graph of Figure 36, it is possible to notice how values are concentrated not around one in particular, but around several different intervals. This is reasonable looking at the trend of delay before context transfer where the delay variation is higher and there are several spikes of delay.

Figure 36: Delay distribution values, test case D2

Cbr flow	Results: rep.2		
	Before ho	**Before co. tra.**	**After co. tra.**
Packets received	12000	11980	
Packet drops	0	20	
Average delay (msec)	9,519	27,895	9,876
Delay variance (msec)	2,133	16,212	2,224
Average IPDV (msec)	3,531	4,759	3,477
IPDV variance (msec)	1,143	3,144	1,139
Delay 95-percentile (msec)	13,731	66,648	14,51
IPDV 95-percentile (msec)	6,011	10,433	6,109

Table 16: Metrics values, analysed repetition

Cbr flow	Average results		
	Before ho	Before co. tra.	After co. tra.
Packets received	12000	11981	
Packet drops	0	19	
Average delay (msec)	9,434	25,658	9,600
Delay variance (msec)	2,124	15,066	2,210
Average IPDV (msec)	3,507	4,739	3,515
IPDV variance (msec)	1,128	3,025	1,172
Delay 95-percentile (msec)	13,587	61,462	14,003
IPDV 95-percentile (msec)	5,962	9,878	6,082

Table 17: Average values, test case D2

Looking at the previous tables, it is possible to notice few packet drops after handover, they are all experienced before context transfer, and an increase in delay average value and variance, while values before handover and after context transfer are, as expected, comparable. The same considerations are valid for the percentiles; for example, the values in the first and third periods for the delay are much closer to the average value than the counterpart is in the second period. Even more significant the worsening of the percentile for IPDV, which is about six time higher when QoS is not enabled than when it is.

This is further proof of the minor delay variation due to the deployment of QoS.

9.5 Test case D3: congested DiffServ class, bounded case

In order to verify that DiffServ is a static architecture and performs badly when the rigid limits of reserved bandwidth are overcome, this test case has been executed. Yet, since AF class can borrow bandwidth from the BE one, which has reserved a considerable amount in our configuration, 2.5 Mbit/sec, before a worsening of performances could be noticed, many VBR 512 kbit/sec flows have been assigned to the AF 2.3 class.

To add flows to the AF 2.3 class, it was not possible to simply add new ones to the others already existing, otherwise delay, IPDV and packet drops would have increased up to a few hundreds or even thousands (as we have checked with preliminary tests) during the second period of interest (before context transfer), the workload in Table 15 being suited for a limit situation. For this reason, we have changed the 4 BE flows to AF 2.3, only after context transfer, in such a way that before handover QoS is ensured, since the reserved bandwidth is not overcome and after context transfer this is no longer true. This number, 4, is not casual; we have in fact increased gradually the number of new flows belonging to AF 2.3 class (and no more to BE) and before noticing a significant worsening of performances for the analysed AF 2.3 flow, it has been necessary that all the 4 flows marked before as BE were converted to AF 2.3.

Another phenomenon gives the proof that when the reserved bandwidth for AF 2.3 is overcome, the needed is shared with the BE class. If more flows were added to the AF 2.3 class, the average delay, its variation and the number of packet drops for the analysed VBR flow increased in the period before context transfer, when it was treated as BE and thus suffered from a diminution of the bandwidth now shared with another class. Table 18 illustrates what was written above about the workload configuration; all the other flows present in Table 15 are unchanged.

Before handover	Before context transfer	After context transfer
10 VBR flows 512 kbit/sec EF 2.3	10 VBR flows 512 kbit/sec EF 2.3	14 VBR flows 512 kbit/sec EF 2.3
4 VBR flows 512 kbit/sec BE	4 VBR flows 512 kbit/sec BE	0 VBR flows 512 kbit/sec BE
Principal flow: CBR 512 kbit/sec AF 2.3	Principal flow: CBR 512 kbit/sec BE	Principal flow: CBR 512 kbit/sec AF 2.3

Table 18: Workload variations, test case D3

The results obtained are valid only for the particular DiffServ configuration chosen. The lower classes are “bounded”, that is, they cannot share bandwidth with higher priority classes when these latter do not need it (they can always share bandwidth instead, regardless of DiffServ configuration, with lower level classes). Test case E1, described below, illustrates what happens in such a situation.

Figure 37 shows how the delay, when 13 flows belong to the AF 2.3 class after context transfer and only 1 to BE, for the CBR principal flow during the phase before context transfer, when it is treated as best effort, increases significantly, up to 175 msec. Among all the repetitions run in case D2, the maximum delay value in BE configuration was 105 msec; this is due to the fact that more bandwidth was available for the BE class, which did not lend bandwidth to the upper level classes.

Figure 38 instead, shows how performances become worse for a flow when the bandwidth reserved for its class is overcome and it is no longer possible to borrow more from lower priority classes. Even if the performances are sensibly better than when it was treated as best effort, the principal flow after the context transfer has a higher delay average value and variation than before the handover. Sensible variations cannot be noticed, after context transfer, for the IPDV, but this is due to the lack of a TCP flow in our workload configuration.

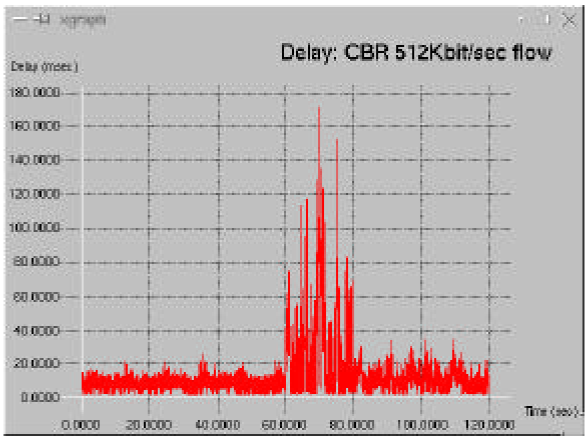


Figure 37: Delay with 13 AF 2.3 flows and 1 BE flow. (VBR 512 kbit/sec)

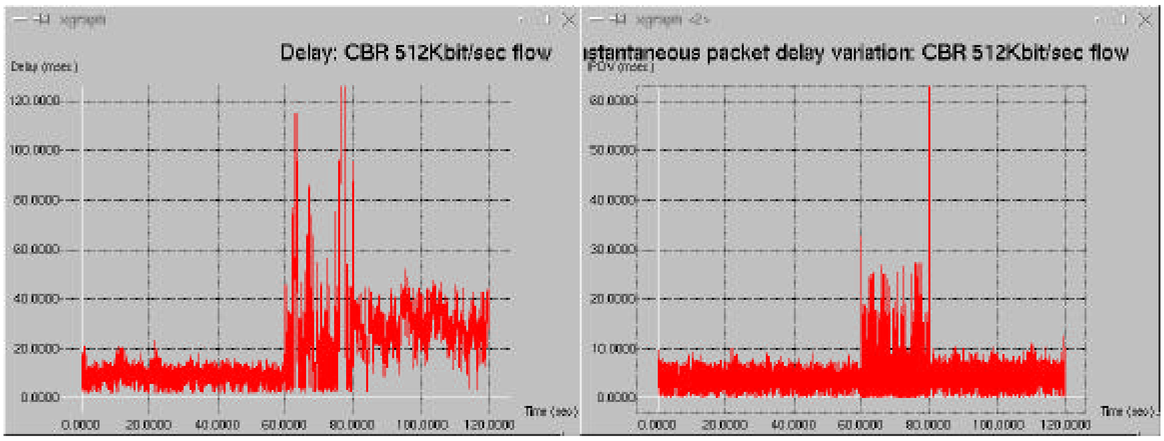


Figure 38: Delay and IPDV, test case D3

The results obtained can be explained also thanks to delay distribution values, illustrated in Figure 39. The graph before handover is spread along the x axis 20 msec, whereas this wideness increases to 90 msec before context transfer (where the values concentration has a behaviour similar to that described in case D2, Figure 36). Finally, after context transfer the graph is less spread than before its execution, but more than before handover, with a maximum number of occurrences lower. Similar considerations are valid for the IPDV; it is possible to underline, from Figure 40, that the distribution after context transfer is slightly more spread than before handover, but such a phenomenon was expected looking at Figure 38, where it was clear how IPDV variations were very similar before handover and after context transfer.

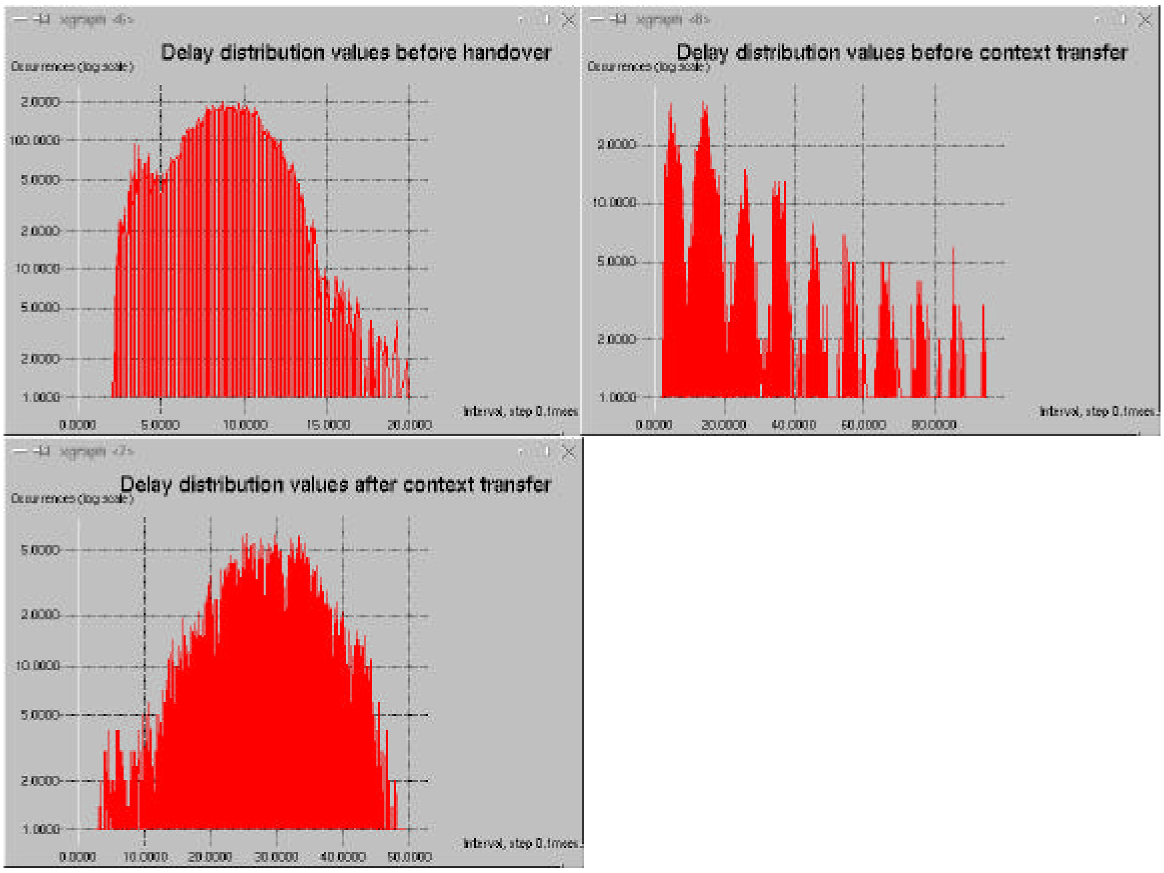


Figure 39: Delay distribution values, test case D3

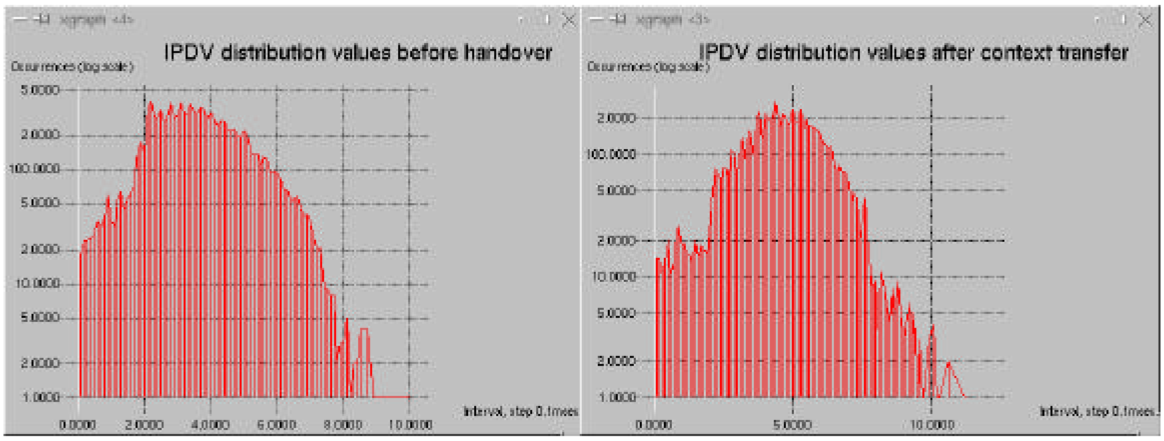


Figure 40: IPDV distribution values, test case D3

The tables below further confirm what was stated above; when the principal flow is treated as best effort, due to the fact that the bandwidth reserved for its class is shared with the AF 2.3

one, such a flow experiences more delay than case D2. The maximum peak is, in this repetition 265 msec, among all the executed repetitions 314 msec; values are also worse than the preliminary test run, whose graphic for delay is showed in Figure 37, because more AF 2.3 flow borrow bandwidth from the BE class. The number of packet drops has significantly increased up to hundreds of drops; as we got confirmation by executing test E2, these hundreds of drops happen after context transfer execution. Thus, even if the delay is reduced, it must be checked whether so many drops can be tolerated. There are no appreciable variations in the values of IPDV, but we expected this based on the discussion above. Moreover, the results after context transfer are better than before its execution, but not as good as before handover, as the previous figures confirm.

Cbr flow	Results: rep.3		
	Before ho	**Before co. tra.**	**After co. tra.**
Packets received	12000	11771	
Packet drops	0	229	
Average delay (msec)	8,797	39,579	28,468
Delay variance (msec)	2,169	35,163	6,142
Average IPDV (msec)	3,593	4,765	4,689
IPDV variance (msec)	1,128	3,692	1,298
Delay 95-percentile (msec)	12,999	191,365	40,758
IPDV 95-percentile (msec)	6,092	10,418	6,987

Table 19: Metrics values, analysed repetition

Cbr flow	Average results		
	Before ho	**Before co. tra.**	**After co. tra.**
Packets received	12000	11757	
Packet drops	0	243	
Average delay (msec)	8,563	37,976	28,078
Delay variance (msec)	2,156	30,605	5,952
Average IPDV (msec)	3,572	4,866	4,670
IPDV variance (msec)	1,125	3,688	1,278
Delay 95-percentile (msec)	12,747	139,597	39,739
IPDV 95-percentile (msec)	6,050	10,475	7,036

Table 20: Average values, test case D3

9.6 Test case D4: comparison between DiffServ classes

This test case has been executed to compare the different levels of service that two different DiffServ classes, EF and AF 2.3 in this specific situation, give to flows that belong to them. Even if this is not so likely in a real situation, we have supposed that before handover the principal flow, always a CBR 512 kbit/sec one, is assigned to the EF class, after handover it is treated as best effort and finally it receives QoS but is scheduled as AF 2.3. The workload utilised is the same in case D2, with the only difference that before handover the principal flow is assigned to the EF class.

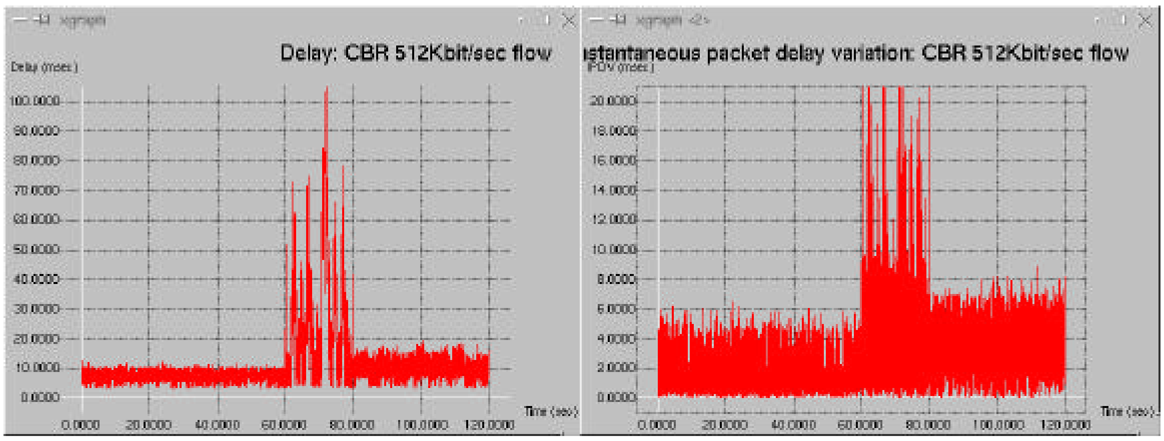


Figure 41: Delay and IPDV, test case D4

Figure 41 above and Figure 42 below clearly show that the EF class guarantees better performance in terms of average delay and its variation, as well as for the IPDV. Looking at the upper graphs in Figure 42, it is possible to see how values are more spread after context transfer than before. The only explanation to this phenomenon as the workload (the same of Table 15, with the difference that the principal flow before handover is EF) is suited in such a way that the reserved bandwidth for each class is not overcome, is that the different behaviour is due to the scheduling class of the flow. The same reasoning is valid for the IPDV, too. EF performs better due to its qdisc, the token bucket filter, which guarantees high performance in terms of constancy of the sending rate and absence of important variations.

The tables below sum up the results obtained. We can see that the number of packet drops, as well as the value of delay, is again similar to case D2. This is correct, since delay increases happen only when the flow is treated as best effort and drops when the bandwidth reserved for the DiffServ class is overcome, and in this test case the BE class does not share bandwidth with the AF 2.3, exactly as happened in case D2. Moreover, the numeric values reported in the tables are consistent with how much was inferred executing a qualitative analysis of delay and IPDV and they show how the service that the EF provides to its flows is better than the counterpart of AF 2.3.

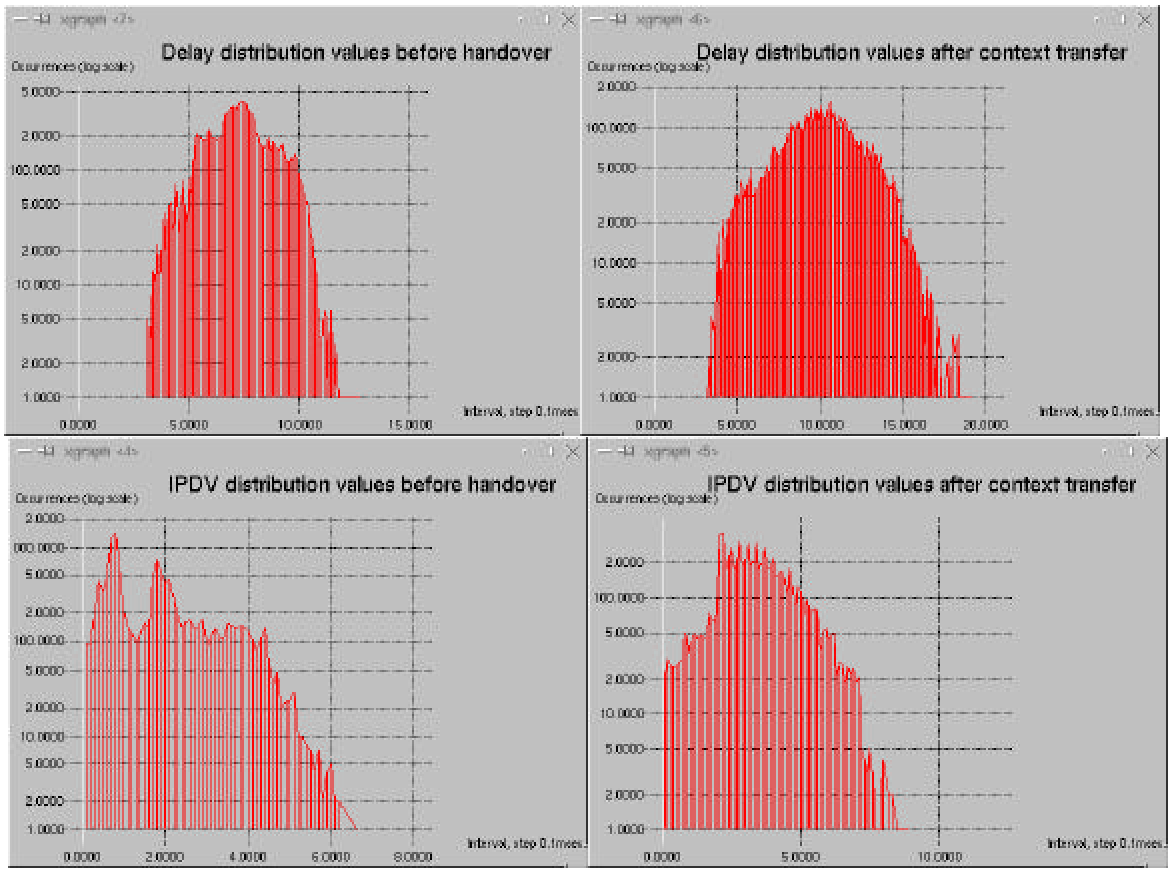


Figure 42: Delay and IPDV distribution values, test case D4

Cbr flow	Results: rep.4		
	Before ho	**Before co. tra.**	**After co. tra.**
Packets received	12000	11987	
Packet drops	0	13	
Average delay (msec)	7,263	24,752	10,015
Delay variance (msec)	1,155	15,911	2,087
Average IPDV (msec)	1,714	4,512	3,377
IPDV variance (msec)	0,979	2,770	1,099
Delay 95-percentile (msec)	9,763	66,136	14,312
IPDV 95-percentile (msec)	4,122	8,995	5,856

Table 21: Metrics values, analysed repetition

Cbr flow	Average results		
	Before ho	**Before co. tra.**	**After co. tra.**
Packets received	12000	11981	
Packet drops	0	19	
Average delay (msec)	5,563	25,808	8,083
Delay variance (ms ec)	1,189	16,410	2,129
Average IPDV (msec)	1,733	4,822	3,458
IPDV variance (msec)	1,009	3,066	1,105
Delay 95-percentile (msec)	8,064	65,159	12,351
IPDV 95-percentile (msec)	4,200	9,758	5,905

Table 22: Average values, test case D4

9.7 Test case E1: congested DiffServ class, unbounded case

One of the most important results of test case D3 (see section 9.5) was that the DiffServ architecture performs badly when the rigid limits for the bandwidth reserved for each class are overcome. This is not exactly true, since the DiffServ configuration used in all the executed tests does not allow to requiring classes to borrow bandwidth from the upper level ones that do not need all the bandwidth reserved for them. Upper level classes can always share bandwidth with lower level ones, as confirmed by the results of test case D3 (see in example Figure 37), but lower level classes can obtain bandwidth by the upper level ones only if these latter do not need it and if the network is explicitly configured for such a situation.

In this test case the limitation was removed, and lower level classes could borrow bandwidth from the upper level ones, modifying properly the TC script that handles the DiffServ architecture. The workload utilised in test case E1 is exactly the same as that employed in case D3, see Table 15 and Table 18, the difference is in the QoS architecture; such a workload allows to realize this scenario the upper classes being lightly loaded compared to the bandwidth reserved for them. In particular, for both the EF and AF 1.1 classes, 1Mbit/sec of bandwidth was reserved, while the total amount of resources wasted by the flows that belong to these classes is only about 100Kbit/sec for EF and 200Kbit/sec for AF 1.1.

The results we have obtained show that the service guarantees for the overloaded class have been better satisfied when lower classes are “unbounded”, that is, can share bandwidth even with the upper level classes, when they do not need it.

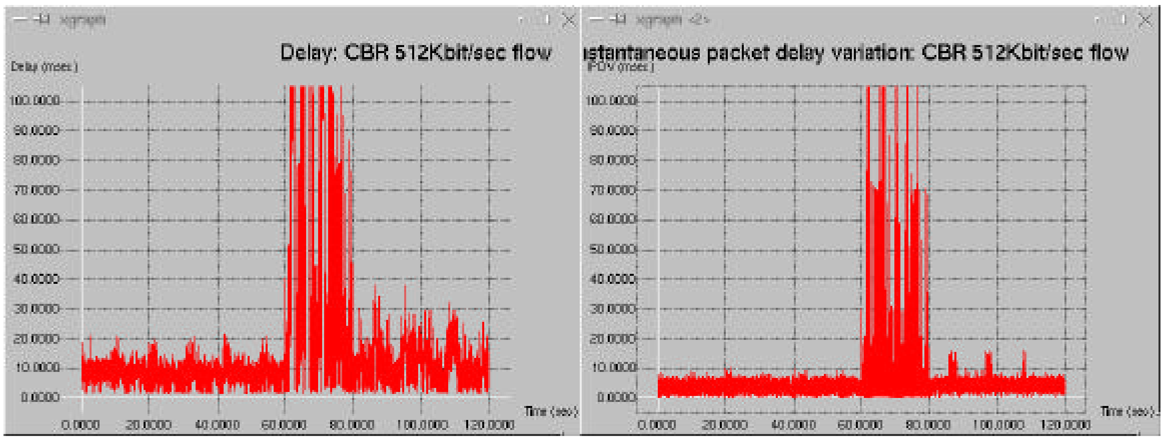


Figure 43: Delay and IPDV, test case E1

We can in fact notice, looking at Figure 43 above, how the delay after context transfer, when the analysed flows belong to the overloaded class, is lower than in the "bounded" case. Nevertheless, a slight increase in delay variance passing from the period before handover to the one after context transfer can be noticed, due to the fact that overload conditions cannot be completely faced by the QoS architecture. Moreover, the number of packet drops has been reduced significantly, passing from the few hundreds of the bounded case to the few tenths of this one.

The price to pay for this performance improvement is that when the flow is best effort, in the 60-80 seconds period, the service it receives from the network is considerably worse than in the bounded case. The maximum delay peak experienced in this repetition is in fact 376 msec, but in other repetitions peaks of even 600 msec have been experienced. In the bounded case, at worst, among all the executed repetitions, the maximum peak was of 315 msec. The explanation to this behaviour is that in the bounded case, packets were scheduled in such a way that if the bandwidth was overcome, they were dropped, otherwise they were transmitted. Once that this limitation was removed, the exceeding packets are stored in transmission buffers waiting to be transmitted rather than being dropped. In fact, in the bounded case, more drops were experienced but less delay, while in the unbounded one, few packets are dropped but their storing leads to a higher delay.

With regard to the IPDV, the behaviour keeps on being pretty similar before handover and after context transfer also in this unbounded case, no significant improvements have been obtained with this modification; however, the worsening is important before context transfer when peaks of 250 msec have been experienced, something that is absolutely intolerable for real time applications. A trade-off must be thus carried off to decide whether it is better to deploy the bounded configuration or the unbounded one. Especially one must take into account whether the worsening of the performance, happening in the best effort period, can be tolerable, otherwise the improvements an unbounded configuration leads to would be wasted by the low service provided in the best effort period. In other words, the decision is if it is better to drop more packets but experience less delay (might be the case of a voice stream) or vice versa to accept more delay in order to achieve less packet drops. A solution, achieved from the results found, could be to bound only the BE class and remove the restrictions for AF classes

Table 23 below shows what has been explained till now: the average delay, for example, is still higher after context transfer than before handover, but this increase is negligible, only about 3 msec (versus the 20 of the bounded case, see Table 20); similar considerations are valid also for the delay variance. The same improvement, as noticed above, is not obtained in the phase after handover and before context transfer when the average delay and its variance have risen up to about 60 and 50 msec respectively. In the bounded case, the same average values in the same period of interest were around 40 and 30 msec. Similar considerations are valid for the percentiles: higher before context transfer and much lower afterwards.

With regard to the IPDV, in the only period when the results change in some manner, the one before context transfer, the average value is slightly higher than in the bounded case. This slight difference means that there are only isolated packets that experience a high IPDV, while on average the others get less IPDV. Yet, the presence of these peaks make the 95-percentile go up from an average of 10 msec in the bounded case to 17 msec as in this case.

Table 23 summarises what written above.

Cbr flow	**Results: rep.4**		
	Before ho	**Before co. tra.**	**After co. tra.**
Packets received	12000	12987	
Packet drops	0	13	
Average delay (msec)	8,070	57,800	11,748
Delay variance (msec)	2,236	50,701	5,070
Average IPDV (msec)	3,586	6,323	4,463
IPDV variance (msec)	1,152	5,242	1,071
Delay 95-percentile (msec)	12,597	209,564	23,689
IPDV 95-percentile (msec)	6,131	17,315	6,593

Table 23: Metrics values, analysed repetition unbounded case

Taking a look at the average values of this test case it is possible to get a confirmation of what the analysed repetition had pointed out, among all the considerable increase in the average delay and mostly delay percentile before context transfer. Instead, the increase of the average IPDV and its variance and percentile are not so evident. This can be due to the fact that only

UDP flows are presents. Any of them are CBR and thus do not affect the IPDV, while many others are VBR but so high in number that the overall effect is the one produced by a CBR flow. The presence of a TCP transfer is an annoyance because even if TCP adapts its sending rate to network conditions, in any case it does not transmit as well as a CBR flow, causing an increase in the values of IPDV; the tests previously executed (test case C2, section 9.3) confirm the explanation to this behaviour.

Cbr flow	**Average results**		
	Before ho	**Before co. tra.**	**After co. tra.**
Packets received	12000	11976	
Packet drops	0	24	
Average delay (msec)	8,090	53,984	11,853
Delay variance (msec)	2,151	52,835	5,112
Average IPDV (msec)	3,525	5,940	4,542
IPDV variance (msec)	1,146	4,638	1,211
Delay 95-percentile (msec)	12,279	224,696	22,510
IPDV 95-percentile (msec)	6,057	12,736	6,655

Table 24: Average values, unbounded case

The following figures show the delay probability distribution for the unbounded case and a comparison of these distributions with the bounded one. These graphics are in some way similar to the “delay distribution values” ones, in fact, in the x axis the same variable is represented, the values of experienced delay in interval of 0.1 msec size. The y axis indicates the percentage fraction of the packets arrived with a delay comprised in the interval specified by the correspondent value in the x axis, i.e. the quantity

$$\frac{nr\_of\_pkts\_with\_specified\_delay...}{total\_number\_pkts\_arrived} \bullet 100$$

Figure 44 below shows the probability distributions for the period before context transfer; in the left-hand one all the delay values experienced during these 20 seconds are represented, while in the right-hand one a zoom of the measured delays until 50 msec is depicted. The right figure has been inserted to compare the trends with the one in Figure 45: as is evident, values are not concentred around an average, but are rather floating.

Looking at Figure 45, delay probability distributions after context transfer, in fact, it is possible to notice that the 1% of packets, the highest percentage for a single interval on the x axis, experience a delay around 10 msec, and that most of the values are concentred around this value. The left part of Figure 44 refers to the whole period before context transfer. The values are again not concentred around a single value and, moreover, also high delays are present, and a single delay value is not assumed as often as after context transfer being the maximum percentage of occurrence of the 0.8 %, for low delay values anyway.

Figure 46 shows the overall delay probability distribution. It is possible to ideally divide the left-hand plot into two parts. The leftmost one, which refers to low delay values, until 50 msec, (the zoom for this period is illustrated in the right-hand Figure 46) corresponds to the period after context transfer, the other to the period before. Since the graphics are now reported only in one, it is easier to notice how before the context transfer values are sparsely distributed. All the contributes to the delay higher than 40 msec, in fact, belong to this time period.

Moreover, the probability of occurrence is lower, the maximum value now being around 0.7%, because this plot sums up contributes of the whole period after handover and thus more values are computed. The trace that the right part of Figure 46 illustrates has a shape pretty similar to the one depicted in Figure 45; this can be explained considering that the most significant contributes to low delay values come from the period after context transfer. Nevertheless, the maximum probability of occurrence is lower, but this is due, as explained above, to the fact that the plot comprises more values, pertinent to a 60 seconds time period, and not more to a 40 seconds time period.

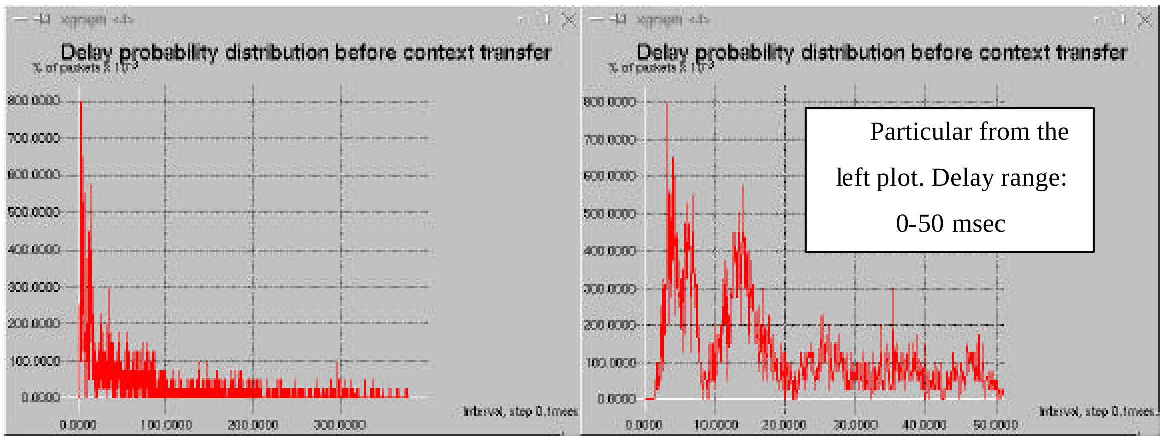


Figure 44: Delay probability distribution before context transfer, unbounded case

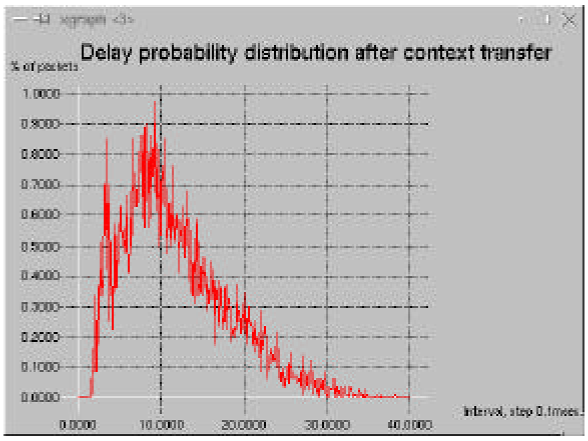


Figure 45: Delay probability distribution after context transfer, unbounded case

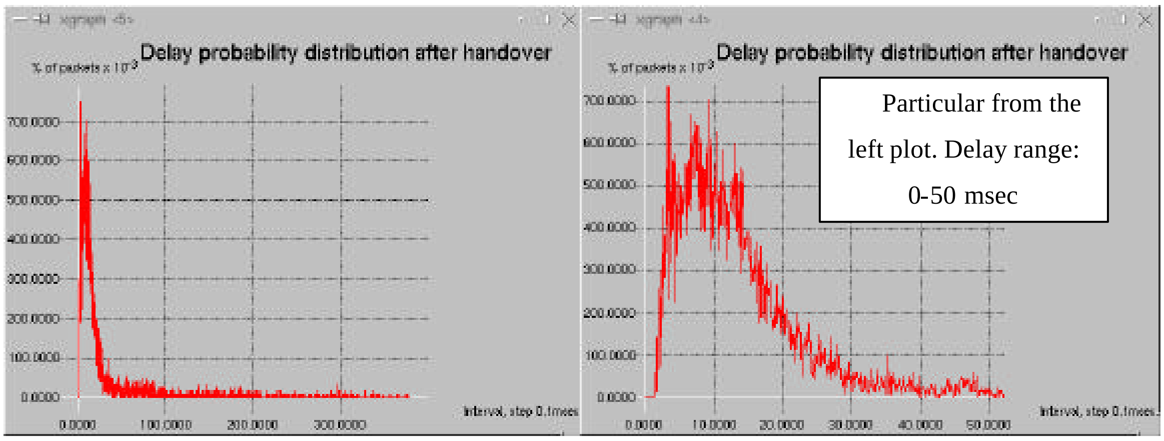


Figure 46: Delay probability distribution after handover, unbounded case

Figure 47 illustrates a comparison of delay probability distributions with the bounded case. The upper part refers to the period after context transfer. The main differences between these two cases in this time period were that in the unbounded configuration, the average delay and its variance were lower, and in fact the values in the unbounded case are lower than the bounded one and moreover, they are, though slightly, less spread. The bottom Figure 47 refers to the whole period after handover, and thus comprises the 20 second period when the analysed flow was treated as best effort. In this case the problem of the trade-off between higher performances in the best effort period and worse in the QoS enabled one and vice versa is evident: the unbounded case in fact, presents delay values higher than the bounded one, and they all have been experienced during the best effort period. In the bounded configuration, in fact, delays no higher than about 250 msec are presents, while in the unbounded one the graphics extends up to more than 350 msec. Anyway, the percentage of packets arriving with this high delay value is quite low, less than 0.05%.

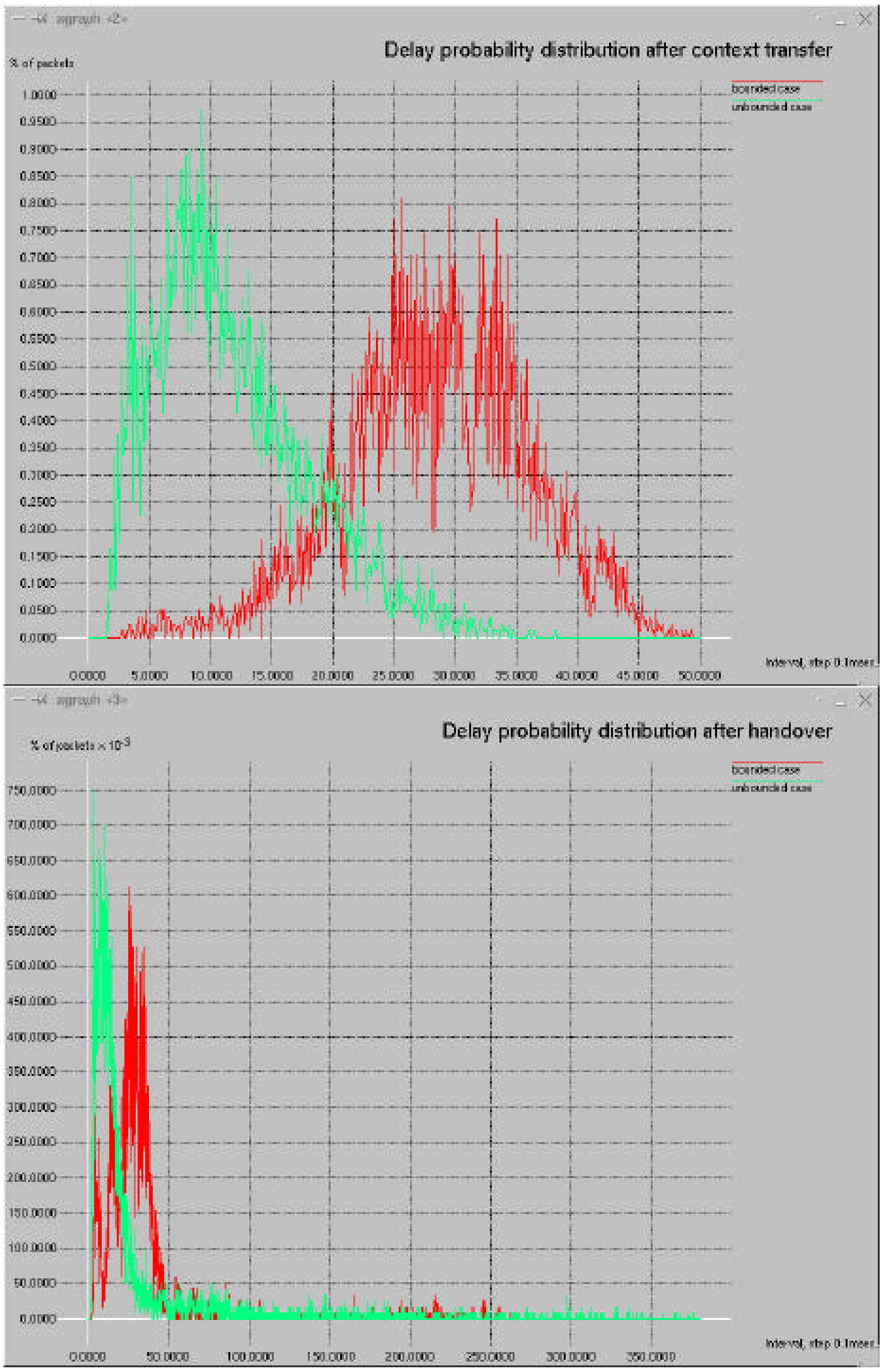


Figure 47: Delay probability distributions comparison

9.8 Test case E2: context transfer threshold comparisons

This test case is different from the others: it has been executed to find out a threshold for the context transfer deployment; in other words, it aims to discover the time after which instantiating QoS in the network leads to inadequate performance. The problem is to define when a performance can be defined as *inadequate*, because it depends on the kind of application. Of course, if packet losses begin to drop until almost zero and the average delay and its variations fall down (for example, if the delay variation is lower than the packet sending rate, this is an acceptable situation), this could be considered a good threshold for context transfer deployment.

The previous reference test cases are D2 and D3 (sections 9.4 and 9.5): we have analysed a situation of a non congested DiffServ class and the opposite of a congested class, to find out if an early execution of context transfer helps in providing QoS to requiring applications. The following Table 25 summarizes the values obtained for the metrics we are interested in, referred only to the period of time from the execution of handover to the context transfer effecting. Four thresholds have been represented, 20, 10, 5 and 3 seconds (after handover), meaning that the context transfer has been executed respectively 80, 70, 65 and 63 seconds after the beginning of the test. The test case we are analysing is D2, when the bandwidth for the AF 2.3 class is not overcome.

Cbr flow	**Average results**			
	Threshold values after handover (seconds)			
	20	**10**	**5**	**3**
Packets received	3977	1991	986	597
Packet drops	19	6	6	1
Average delay (msec)	25,658	13,954	13,855	9,657
Delay variance (msec)	15,066	8,234	8,819	3,614
Average IPDV (msec)	4,739	4,157	3,887	3,711
IPDV variance (msec)	3,025	2,131	1,775	1,459
Delay 95-percentile (msec)	61,462	47,423	41,605	20,875
IPDV 95-percentile (msec)	9,878	8,842	8,205	7,149

Table 25: Thresholds comparison, non-congested DiffServ class, before context transfer period

The number of packet drops is always quite low; in the worst case, when the context transfer is carried out 20 seconds after handover, packets dropped represent only the 0.16% of those sent. The first thing to notice is that, in this scenario, effecting context transfer after 10 seconds or 5 seconds leads to pretty similar results, both in terms of packet drops rather than of time parameters. Secondly, it is evident the gradual improvement obtained anticipating more and more the context transfer. For example, passing from 20 to 3 seconds of delay, the average delay reduces more than 50%, while an even higher reduction is experienced by delay variance. The delay percentile passes from about 60 msec in the 20 seconds threshold, through about 45 msec in the middle cases to only 20 msec in the latter case. Similarly, improvements, though of minor relevance, are obtained for IPDV too.

Table 26 instead compares the values of the analysed metrics in a scenario where the bandwidth reserved for the class AF 2.3 is overcome (see test case D3). The table has to be read in the same way as the previous; the only difference is that to better point out variations, the last threshold has been set at 1 second instead of 3. As can be seen, especially looking at the results about packet drops, the time of execution of context transfer does not impact so much; this is because most of the drops happen during the period after context transfer, as the bandwidth reserved for the AF 2.3 class is now overcome. The drops due simply to the redirection of the flow towards a best effort path are very few compared to those due to the overcoming of the reserved bandwidth, compare Table 26 and Table 28. Anyway, applying context transfer earlier can, in this situation too, improve performance in terms of delay and IPDV, the same considerations as above being valid also in this case.

Since there were no significant variations in the metric values in the period of time after context transfer, regardless of the delay of its execution, both in case of congested (beside the packet drops, as noticed above) and non-congested DiffServ class, the results relative to this period of time have not been represented. This is understandable, because whatever is the duration of the period that a flow receives QoS, it expects to obtain the same level of service, of course granted that network conditions remain unchanged. In other words, if no new flows are added to the network, a flow must receive the contracted QoS for 40 seconds as well as for, say, 55. This is the reason why the most significant changes are only in the period before context transfer. Anyway, we have also reported the results pertinent to the whole period after handover, comprised of the time before and that after context transfer.

Table 27 and Table 28 show the same parameters referred to the whole 60 seconds period after handover. The first one evidences the fact that when the bandwidth reserved for a DiffServ class is not overcome, no packets are lost, the number of packet drops being the same as that pertinent to the period before context transfer. The results obtained for the time parameters are consistent to those drawn in Table 25, and are gradually better the earlier the context transfer is deployed; moreover, since this time period also comprises the QoS enabled part, they are overall better than those reported in the abovementioned Table 25.

Cbr flow	Average results			
	Threshold values after handover (seconds)			
	20	10	5	1
Packets received	3990	1985	997	193
Packet drops	10	15	3	7
Average delay (msec)	37,976	35,798	24,569	23,137
Delay variance (msec)	30,605	23,155	8,946	6,758
Average IPDV (msec)	4,866	5,114	4,768	4,602
IPDV variance (msec)	3,688	2,842	1,824	1,425
Delay 95-percentile (msec)	139,597	120,903	52,048	35,183
IPDV 95-percentile (msec)	10,475	9,483	8,677	7,240

Table 26: Thresholds comparison, congested DiffServ class, before context transfer period

Cbr flow	Average results			
	Threshold values after handover (seconds)			
	20	10	5	1
Packets received	11981	11994	11994	11999
Packet drops	19	6	6	1
Average delay (msec)	14,930	10,252	10,234	8,825
Delay variance (msec)	8,583	4,179	4,184	2,620
Average IPDV (msec)	3,921	3,717	3,618	3,569
IPDV variance (msec)	1,835	1,463	1,343	1,238
Delay 95-percentile (msec)	47,602	23,260	20,110	14,032
IPDV 95-percentile (msec)	8,288	7,173	6,678	6,334

Table 27: Thresholds comparison, non-congested DiffServ class, after handover period

Table 28 is related to the case when the DiffServ class is congested. It is possible to notice a significant increase of packet drops with respect to the counterpart of the period before context transfer, proof that in a situation of congestion many packets are dropped, even when the QoS is enabled. For the time parameters, the same considerations carried out for the non-congested case are valid; in particular, we notice a high reduction of the delay 95-percentile. This phenomenon can be explained with the fact that now this quantity is computed taking into account the whole period after handover, where delays much lower than those experienced before context transfer are present.

Cbr flow	Average results			
	Threshold values after handover (seconds)			
	20	10	5	1
Packets received	11757	11722	11788	11772
Packet drops	243	278	212	228
Average delay (msec)	31,440	29,675	24,618	23,995
Delay variance (msec)	14,537	11,394	7,292	5,899
Average IPDV (msec)	4,736	4,775	4,698	4,616
IPDV variance (msec)	2,096	1,744	1,409	1,266
Delay 95-percentile (msec)	76,335	56,911	38,541	35,312
IPDV 95-percentile (msec)	8,881	8,385	7,571	7,043

Table 28: Thresholds comparison, congested DiffServ class, after handover period

9.9 Test case F: analysis of a TCP flow

In this test set, we have analysed a TCP stream, assigned to the best effort class, directed from Kemi-4 to Kemi-9, to check out variations in its performances in case of handover. As before, the handover is scheduled 60 seconds after the beginning of the simulation and the flow is redirected towards Kemi-9 through Kemi-11 instead of Kemi-10 after handover. The results were obtained by analysing the log file provided by the Tcpdump tool, at the sending interface. The hypothesised scenario is that before handover, that is on the path comprising Kemi-10, there is no QoS instantiated so all the flows are treated as best effort, while after handover two situations are possible:

1. The TCP flow traverses a path where it is the only BE flow while all the others receive QoS in a DiffServ configuration.
2. The network is entirely best effort.

The workload utilised is slightly different from the others: we suppose that only the TCP flow, the principal one, is directed towards Kemi-9, while UDP flows stop at Kemi-10/11. The two paths traversed by the TCP flow, before and after handover, are loaded in the same way, the only difference is that in the first case after handover the QoS is enabled and the background flows are assigned to a DiffServ class. In the second case, the TCP flow, thus, should not notice any variation in the performances before and after handover. Table 29 shows the workload described above:

Stream	Packet size (bytes)	Protocol	Packet rate (pkts/sec)	Data Rate (Kbit/sec)	Number of flows	Recipient	DiffServ class
Data	1024	TCP	20000 buffers sent		1	Kemi-9	BE
VoIP (ADPCM)	104	UDP/CBR	50	41.6	4	Kemi-10/11	AF 1.1
VoIP (GSM)	36	UDP/CBR	50	14.4	7	Kemi-10/11	EF
Video (HQ)	320	UDP/VBR	200	512	14	Kemi-10/11	AF 2.3

Table 29: Workload utilised, test case F

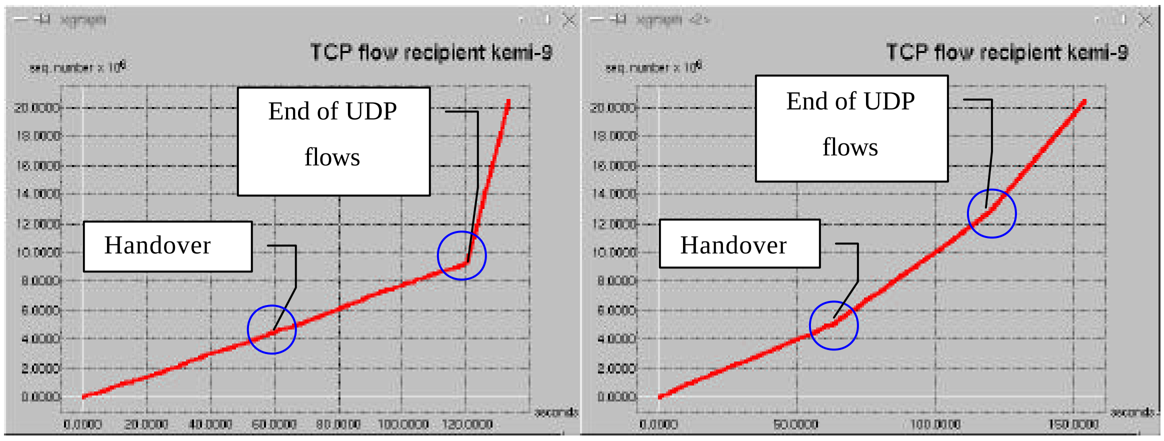


Figure 48: TCP sequence number, best effort network (left) and after QoS deployment (right)

In Figure 48 the trend of TCP sequence number over time, plotted in a relative way, is shown. This means that the first segment of the TCP connection has been assigned a sequence number of 0, while the others will get one consequently from this first one, according to the behaviour of the TCP protocol. This allows us to better compare the performance in the two examined cases, the total best effort network one (left-hand Figure 48) and the one with QoS deployment after handover (right-hand Figure 48).

Before handover, as expected, the performances are roughly the same and the sequence number after 60 seconds is almost the same (about 5); after handover the trends become different: the TCP flow receives a slightly better service when it is the only best effort flow among many others belonging to a DiffServ class. In fact when the UDP flows terminate, in the all best effort network the TCP sequence number is around 9, in the QoS enabled network this number grows up to about 13 instead. This improvement in the performance is besides evident directly looking at Figure 48 as the slope of the line that represents the sequence number is higher in the right case. Moreover, as expected, no significant changes in the performances are experienced when the network is everywhere best effort. Note that all the sequence numbers are intended to be multiplied for 10^6.

The following Figure 49 gives an explanation to this behaviour, previously described, of the TCP flow, which receives better service when the QoS is enabled even though it is the only best effort in the network. This figure has been obtained with a workload slightly different from that illustrated in Table 29. In fact, three of the fourteen AF 2.3 flows have been treated as best effort; in this latter class the TCP flow and those previously AF 2.3 are thus present. While before the TCP flow was the only belonging to the BE class, being able thus to utilise all the bandwidth reserved for that class, now it must share it with other flows. This fact is the reason for the performance worsening noticed in Figure 49.

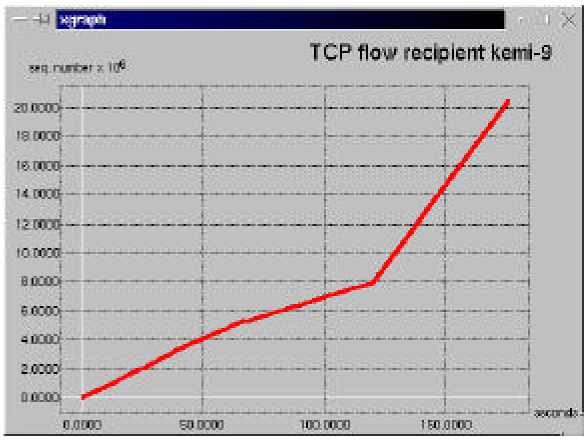


Figure 49: TCP sequence number, BE class loaded by UDP flows

Nevertheless, coming back again at Figure 48, the overall performance received by the TCP flow is better in the best effort case. The transfer of the data in fact now finishes in about 130 seconds, while if the QoS is enabled the transfer is completed after about 150 seconds; this can be noticed looking at the slope of the lines, higher in the best effort case than in the other. Such a phenomenon means that after the termination of the UDP flows, a best effort network offers a service much better to the TCP flow than a DiffServ enabled one, since it is able to transfer a higher amount of data in less time (this is the meaning of the higher slope of the curve). The DiffServ architecture is not only useless in a condition of light load for best effort flows, but also harmful and worsens performances; after the termination of UDP flows, in fact, the network is very lightly loaded the TCP one being the only active flow.

Table 30 below illustrates the throughput received by the TCP flow and the transfer time in both the examined cases. It is possible to see the overall better performance of the best effort network with respect to the DiffServ one:

Analysed flow	Network configuration	
TCP/BE	**Best effort**	**DiffServ**
Transfer Time (seconds)	133,19	154,27
Throughput (kbit/sec)	1201,27	1037,14

Table 30: TCP flow transfer results

Figure 50 below shows the results of a test run to further prove that DiffServ in a lightly loaded network performs badly for BE flows. The transfer of a single TCP BE flow in both the scenarios described above has been executed, so we have considered a very lightly loaded network. Until handover execution, the performances are approximately the same, as the sequence number is about 50 at the moment of its execution. After the redirection of the flow, in the best effort case the slope of the line obviously does not change at all, and the transfer is terminated after about 80 seconds; when QoS is instantiated, the slope decreases brusquely and the transfer is terminated in almost the double time. Note that in this case we have transferred a higher amount of data (70,000 buffers 1024 bytes each) otherwise the transfer of only 20,000 buffers would have finished after about only 30 seconds, as we verified in some preliminary tests.

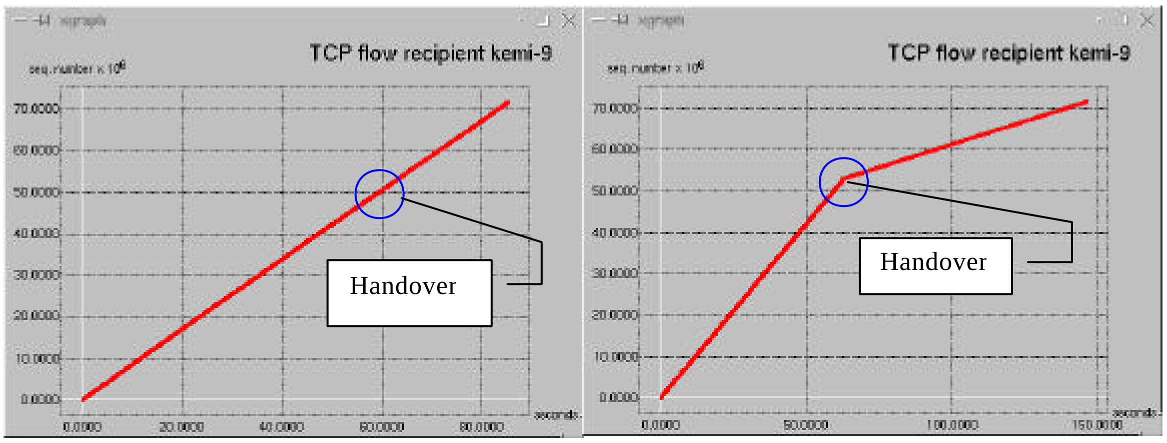


Figure 50: TCP sequence number, transfer of a single TCP flow. Network BE (left) and QoS enabled (right)

In any case, it is not possible to notice in all the reported graphics a more detailed TCP sequence number trend, made of retransmission and timeouts, since the number of packets logged is quite high, due to the high amount of data transferred.

9.10 Discussion on the remaining test cases

In this section, we describe briefly the test cases for which we have not reported the results. In case A1, there was the transfer of a CBR 14.4 Kbit/sec flow in an all best effort network, sensibly more loaded after handover; the results show an obvious worsening of performance for the principal flow. In case A3, which emulates a situation where the context is transferred before the completion of handover, the QoS is enabled since the beginning of the simulation in the entire network; the workload being the same for both the paths, the foreseen result is that the performance after handover is roughly the same as before. Case A4 deals with a situation of delayed context transfer, which is shown to be a good technique to reduce average delay and IPDV and their variations.

Test case B2 is similar to the A3 one, the differences are in the workload: the principal flow is CBR 512 Kbit/sec. The results are not changing in performance. Case B3, delayed context transfer for a CBR 512 Kbit/sec flow, whose delay and IPDV trends are shown in the section 9.2, Figure 31, again gives a confirmation of the effectiveness of context transfer deployment.

In the test cases C*, the principal flow was a CBR 160 Kbit/sec, assigned to the AF 1.1 class (beside case C1). Case C1 was the all best effort one, like A1, and the results obtained are consistent to that test case, as well as case C3 dealt with a situation of QoS already instantiated when the handover was performed and thus no significant change of performance due to flow re-direction was achieved. Case C4, delayed context transfer, gives the same results as cases A4 and B3. A common characteristic of the cases C* was the presence of a TCP bulk data transfer in the workload. The unpredictability of the TCP transfer rate made in all the tests increase the variations of delay and IPDV for the principal flow and made the application of context transfer more powerful. Context transfer is demonstrated to be valuable also for reducing IPDV and not only for reducing delay.

Finally, test case D1 describes the worsening of the performance from which a CBR 512 kbit/sec flow suffers in an all best effort network, high loaded after handover, with a TCP flow in the workload. The same considerations made for case C1 are valid.

10 Conclusions

In this thesis we have presented a study about a new technique, the context transfer, under development at writing time. Context transfer for QoS should ensure a good level of service to the requiring applic ations, even if its deployment is especially meant for multimedia applications in case of handover. The environment of deployment of this technique is the radio medium, being more and more oriented the market and the research towards this huge field. An example of application of context transfer could be in fact in the third generation cellular networks, UMTS.

The conditions that could be present in a real context transfer scenario have been faithfully reproduced, particular attention was paid to the scenarios of a delayed deployment of context transfer, mainly to find out a time limit for its deployment. A further object of studies could be the execution of similar experiments in a wireless environment, to consider a situation as much as possible similar to the real case.

From our comparative analysis of different network configurations, loaded with diverse kinds of workloads, we can notice the general effectiveness of context transfer in providing QoS in such a scenario and the sensible performance improvements experienced once the context was transferred along the new path. The focus of our studies was on the context transfer, and in general on the problems that the handover of a flow along a new path can produce; context transfer was considered as the solution to these potential handover problems. The experiments run also gave us the occasion to study the performance of the DiffServ architecture in a situation of traffic re-direction.

Context transfer always demonstrated itself as a good technique that allows reducing significantly the delay and the Instantaneous Packet Delay Variation (IPDV) experienced by a flow, under different network configurations. The timeliness of packet delivering is a very important issue for the forthcoming network architectures. The improvements are evident even in the presence of a TCP flow, which, though not being a trouble for UDP flows from the throughput point of view, obstacles a timely delivering of packets, since its sending rate is too unpredictable and depending on network conditions such as congestion and internal protocol way of working (retransmissions and timeouts). It was also shown that as before context transfer is deployed, paying attention to not overcome the bandwidth reserved for the DiffServ class, better performance is attained, both in terms of packet drops and reduction of delay and IPDV average values and variance.

Another important conclusion was obtained regarding the DiffServ architecture, having studied context transfer as the DiffServ deployment also on a radio access network. Particularly, it was found that the deployment of such an architecture in a lightly loaded network is not effective, and even the network performs better in best effort configuration, for a best effort flow, if its load is not close to the limit conditions. In other words, it is not wise to execute QoS DiffServ context transfer if the network is lightly loaded, for best effort flows; of course, the other flows, those receiving QoS, take advantage of context transfer performing. The FIFO queuing in such a situation seems a good compromise to offer the service to best effort flows.

Another result directly connected to the structure of the QoS architecture utilized is related to the flexibility problems that affect DiffServ management, in certain configurations. If the bandwidth is assigned statically to the various classes, and in case the bandwidth reserved for a class is overcome, it was found that DiffServ does not perform adequately, delay and packet drops being much increased if the lower classes are bounded and cannot share bandwidth with the upper classes when they do not need it. By setting the lower classes unbounded, a sensible improving of performance was noticed, but it was to the detriment of the best effort class, which does not receive any improvement from this change in network configuration, but even an increase in delay due to additional buffering delays. For this reason, at least for a configuration similar to the one we have used, we suggest to bound the best effort class and let AF classes unbounded.

The problem of handover towards a highly loaded path can be avoided with the execution of Candidate Access Router (CAR) discovery. Yet, this issue was outside the scope of this thesis, and it is a subject of future research. Though the objectives of this thesis are almost fully completed, a lot more research is still required in order to successfully achieve context transfer for QoS in radio access networks, ranging from a study of the problem in a radio environment to the effective deployment of the CAR discovery. Research can be extended in many directions, one among all the extension of context transfer also to other QoS disciplines (IntServ for example) and to other issues, such as header compression or security, but before exploiting experiments, the standardization of the context transfer protocol and the definition of the requirements that the whole context transfer architecture must possess must be completed. The IETF Seamoby working group is the task force entitled to achieve these results.

11 References:

[1] Postel J. ed., “Internet Protocol, DARPA internet program protocol specifications”, RFC 791 (Standard), September 1981.

[2] Postel J. ed., “Transmission Control Protocol, DARPA internet program protocol specifications”, RFC 793 (Standard), September 1981.

[3] Hays T., “Achieving QoS in IP networks”, in Quality of Service over next generation data networks, Mohammed Atiquazzam, Mahbub Hassan, Editors, Proceedings of SPIE Vol. 4524, pages 109-116. August 2001.

[4] Huston J., “Internet Performance: survival guide. QoS strategies for multiservice networks”, John Wiley & sons, Inc.

[5] Chalmers D., Sloman M., “A survey of QoS in mobile computing environments”, IEEE Communications Surveys, Second Quarter 1999.

[6] Bochmann G., Hafid A., “Some principles for QoS management”, Distributed Systems Engineering, vol. 4, pp. 16-27, IOP publishing, 1997.

[7] Knoche H., de Meer H., “Quantitative QoS mapping: a unifying approach”, Proceedings on 5th IFIP workshop on QoS 1997, 1997.

[8] Blair G., Stefani J.B., “Open distributed processing and multimedia”, Addison-Wesley, 1997.

[9] Hutchinson D., et al., ”QoS management in distributed systems”, network and distributed systems management, Sloman M., ed., pp. 273-302, Addison-Wesley 1994.

[10] Bennami F., Boutignon A., Simoni N., “IP QoS issues: efficient cooperation of DiffServ, MPLS and Management”, in Quality of Service over next generation data networks, Mohammed Atiquazzam, Mahbub Hassan, Editors, Proceedings of SPIE Vol. 4524, pages 109-116. August 2001.

[11] CCITT Rec. I.121

[12] Laubach M., “Classical IP and ARP over ATM”, RFC 1577 (standards track). January 1994.

[13] http://www.diit.unict.it/~spalazzo/Corso_reti/Cap11.pdf

[14] http://www.iec.org/online/tutorials/atm_fund/topic04.html

[15] Braden, R., Clark, D. and S. Shenker, "Integrated Services in the Internet Architecture: an Overview", RFC 1633 (Informational), June 1994.

[16] Wroclawski J., “Specification of the Controlled-Load Network Element Service”, RFC 2211 (Standard Track), September 1997.

[17] Shenker S., Partridge C., Guerin R., “Specification of Guaranteed Quality of Service”, RFC 2212 (Standard Track), September 1997.

[18] Braden R., et al., " Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205 (Standards Track), September 1997.

[19] Peterson L., Davies B., "Computer network: a system approach", Morgan Kauffman.

[20] Wroclawski J., "The Use of RSVP with IETF Integrated Services", RFC 2210 (Standard track), September 1997.

[21] Blake S., et al., " An Architecture for Differentiated Services", RFC 2475 (Informational), December 1998.

[22] Nichols K., et al., "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474 (Standard Track), December 1998.

[23] Heinanen J., et al., "Assured Forwarding PHB Group", RFC 2597 (Standard track), June 1999.

[24] Jacobson V., Nichols K., Poduri K., "An Expedited Forwarding PHB", RFC 2598 (Standard track), June 1999.

[25] Jacobson V., Nichols K., Zhang L., "A Two-bit Differentiated Services Architecture for the Internet", RFC 2638 (Informational), July 1999.

[26] Bernet Y., et al., "A Framework for Integrated Services Operation over Diffserv Networks", RFC 2998 (Informational), November 2000.

[27] Rosen E., Viswanathan A., Callon R., "Multiprotocol Label Switching Architecture", RFC 3031 (Standard track), January 2001.

[28] Awduche D., et al., " Requirements for Traffic Engineering Over MPLS", RFC 2702 (Informational), September 1999.

[29] Rodriguez A., et al., "TCP/IP tutorial and technical overview", IBM red books, in http://www.redbooks.ibm.com/pubs/pdfs/redbooks/gg243376.pdf.

[30] Andersson L., et al., "LDP Specification", RFC 3036 (Standard Track), January 2001.

[31] Thomas B., Gray E., "LDP applicability", RFC 3037 (Informational), January 2001.

[32] "MPLS: an introduction to multiprotocol label switching". Nortel networks. Retrieved in: http://etudiant.esigetel.fr/docs/mpls.pdf

[33] Casner S., et al., "RTP: A Transport Protocol for Real-Time Applications", RFC 1889 (Standard track), January 1996.

[34] Postel J., "User Datagram Protocol", RFC 768 (Standard track). August 1980.

[35] Rouhana N., Horlait E., "Differentiated services and integrated services use of MPLS", in proceedings on Computers and Communications, 2000. ISCC 2000. Fifth IEEE symposium. Page(s): 194 –199.

[36] Manner J., et al., "A Study on QoS Provision for IP-based Radio Access Networks", Tyrrhenian International Workshop on Digital Communications. Taormina, Italy, September 2001. (Invited paper) In http://www.cs.helsinki.fi/u/jmanner/

[37] Manner J., et al., "Mobility related terminology", Internet Draft (Work in progress). draft-manner-seamoby-terms-03.txt

[38] Manner J., et al., ”Evaluation of mobility and QoS interaction”, Computer Networks Journal (Elsevier Science publisher). To appear. In http://www.cs.helsinki.fi/u/jmanner/

[39] Perkins C., ed., “IP mobility support”, RFC 2002 (Standard Track), October 1996.

[40] Perkins C., “IP Encapsulation within IP”, RFC 2003 (Standard track), October 1996.

[41] Perkins C., Johnson D., “Route optimisation in Mobile IP”, Internet draft (Work in progress), September 2001.

[42] Johnson D., Perkins C., “Mobility Support in IPv6”, Internet draft (Work in progress), July 2000.

[43] Deering S., Hinden R., “Internet Protocol, Version 6 (IPv6) Specification”, RFC 2460 (Standard track), December 1998.

[44] Thomson S., Narten T.,"IPv6 Stateless Address Autoconfiguration", RFC 1971 (Standard track), August 1996.

[45] Johnson D., Deering S., “Reserved IPv6 Subnet Anycast Addresses”, RFC 2526 (Standard track), March 1999.

[46] Narten T., Nordmark E., Simpson W., “Neighbour Discovery for IP Version 6 (IPv6)”, RFC 2461 (Standard track), December 1998.

[47] Plummer D., “An Ethernet Address Resolution Protocol”, RFC 826 (Standard track), November 1982.

[48] Mouly M., Pautet M., “The GSM System for Mobile Communications”. Published by the authors, 1992.

[49] ETSI. GSM 03.02, Network architecture.

[50] ETSI. GSM 03.60 GPRS service description, Stage 2.

[51] 3G TS 22.101 Universal Mobile Telecommunications System (UMTS): UMTS phase 1 Release 99.

[52] 3G TS 23.101 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects. General UMTS Architecture (version 3.0.1).

[53] 3G TS 25.401 (1999-07) 3rd Generation Partnership Project; Technical Specification Group Radio Access Network; UTRAN Overall Description (version 3.0.0).

[54] Robert L., Pissinou N., Makki S., “Third generation wireless network: the integration of GSM and Mobile IP”, Wireless Communications and Networking Confernce, 2000. WCNC. 2000 IEEE , 2000 Page(s): 1291 -1296 vol.3.

[55] ETSI, GSM 02.34 v8.0.0, “Digital cellular telecommunications system (phase 2+); High speed circuit switched data (HSCSD)”, Stage 1. 1999.

[56] Priggouris, G., Hadjiefthymiades, S., Merakos, L., “Supporting IP QoS in the General Packet Radio Service”, IEEE Network , Volume: 14 Issue: 5 , Sept.-Oct. 2000 Page(s): 8 –17.

[57] Venken, K., De Vleeschauwer, D., De Vriendt, J., “Designing a DiffServ capable IP-backbone for the UTRAN”, 3G Mobile Communication Technologies, 2001. Second International Conference on (Conf. Publ. No. 477), 2001 Page(s): 47-52.

[58] Marques V., et al., " Enabling IP QoS in mobile environments", IST Mobile Communication Summit, Barcellona. September 2001.

[59] Dommety G., ed., "Fast Handovers for Mobile IPv6", Internet Draft (Work in progress). draft-ietf-mobileip-fast-mipv6-02.txt

[60] BRAIN: Broadband Radio Access for IP-based networks, IST-1999-10050, http://www.ist-brain.org.

[61] Chen J., et al., "A QoS architecture for future wireless IP networks", Twelfth IASTED International Conference on Parallel and Distributed Computing Systems (PDCS 2000), Las Vegas, NV, November 2000.

[62] Talukdar A., Badrinath B., Acharya A., "MRSVP: A Resource Reservation Protocol for an Integrated Services Packet Network with mobile hosts", in Proceedings of ACTS. Mobile summit '98, June 1998.

[63] The IETF Seamoby Working group. http://www.ietf.org/html.charters/seamoby-charter.html

[64] Manner J., Raatikainen K., "Extended Quality of Service for Mobile Networks", IEEE/IFIP Ninth International Workshop on Quality of Service (IWQoS 2001) Karlsruhe, Germany, June 6-8, 2001. Published in the Springer LNCS Series Vol. 2092, pp. 275-280. (© Springer-Verlag).

[65] Gai S., et al., "RSVP proxy", Internet Draft (Work in progress), July 2001. draft-ietf-rsvp-proxy-02.txt.

[66] Lee S., et al., "INSIGNIA: an IP based quality of service framework for mobile ad hoc networks", Journal of parallel and distributed computing, Vol 60 no 4, pp 374-406, April 2000.

[67] Banchs A., et al., "Service extensions for elastic and real time traffic in 802.11 wireless LAN", 2001 IEEE Workshop on High Performance Switching and Routing, Page(s): 245 –249.

[68] Aad I., Castelluccia C., "Differentiation mechanisms for IEEE 802.11", INFOCOM 2001. Proceedings. IEEE , Volume: 1 , 2001 Page(s): 209 -218 vol.1.

[69] Johnsson M., "HiperLAN/2 – The Broadband Radio Transmission Technology Operating in the 5 GHz Frequency Band", White paper. Found in http://www.hiperlan2.com

[70] Specification of the Bluetooth system, http://www.bluetooth.com

[71] Kadelka A., Masella A., "Serving IP Quality of Service with HiperLan 2", Computer Networks, Volume 37, Issue 1, September 2001, Pages 17-24

[72] MIND: Mobile IP based Network Developments (IST-2000-28584), http://www.ist-mind.org/

[73] Kempf J., ed., "Problem Description: Reasons for Performing Context Transfers Between Nodes in an IP Access Network", Internet Draft (Work in progress). draft-ietf-seamoby-context-transfer-problem-stat-04.txt

[74] Syed H., et al., ”General Requirements for Context Transfer”, Internet Draft (Work in progress). draft-ietf-seamoby-ct-reqs-02.txt

[75] Syed H., Kenward G., “General requirements for a context transfer framework”, Internet Draft (Work in progress). draft-hamid-seamoby-ct-reqs-01.txt

[76] Syed H., Kenward G., “Context transfer framework”, Internet Draft (Work in progress). draft-hamid-seamoby-ct-fwk-00.txt

[77] Koodly R., Perkins C., “A context transfer framework for seamless mobility”, Internet Draft (Work in progress). draft-koodly-seamoby-ctv6-02.txt

[78] Trossen D., et al., ”Issues in candidate access router discovery for seamless IP level hahdoffs”, Internet Draft (Work in progress). Draft-ietf-seamoby-cardiscovery-issues-02.txt

[79] Van der Zee M., Heijenk G., “Quality of Service in Bluetooth networking”, in http://ing.ctit.utwente.nl/WU4/Documents/Bluetooth_QoS_ING_A_part_I.pdf

[80] Postel J., “Domain Name System structure and delegation”, RFC 1591 (Informational), March 1994.

[81] Yeong W., Howes T., Kille S., “Lightweight Directory Access Protocol”, RFC 1777 (Standard Track), March 1995.

[82] Guttman E., et al., ”Service Location Protocol, version 2”, RFC 2608 (Standard Track), June 1999.

[83] Trossen D., et al., ”Protocol for Candidate Access Router Discovery for Seamless IP-level Handovers”, Internet Draft (Work in progress). draft-trossen-seamoby-cardiscovery-00.txt

[84] Koodly R., Perkins C., Tiwari M.,”Context Relocation for Seamless Header Compression in IP Networks”, Internet Draft (Work in progress). draft-koodli-seamoby-hc-relocate-01.txt

[85] Westphal C., Koodly R., Perkins C., “Context Relocation of QoS Parameters in IP Networks”, Internet Draft (Work in progress). draft-westphal-seamoby-qos-relocate-00.txt

[86] Syed H., Jaseemuddin M., “QoS (DiffServ) Context Transfer”, Internet Draft (Work in progress). draft-hamid-seamoby-ct-qos-context-00.txt

[87] Heinanen G., Guerin R., ”A Single Rate Three Colour Marker”, RFC 2697 (Informational), September 1999.

[88] Heinanen G., Guerin R., ”A Two Rate Three Colour Marker”, RFC 2698 (Informational), September 1999.

[89] Kempf J., et al., ”BiDirectional Edge Tunnel Handover for Ipv6”, Internet Draft (Work in progress). Draft-kempf-beth-ipv6-02.txt

[90] Nakhjiri M., Singh A., ”Time Efficient conteXt Transfer (TEXT)”, Internet Draft (Work in progress). draft-nakhjiri-seamoby-text-ct-00.txt

[91] MGEN traffic generator: ftp://manimac.itd.nrl.navy.mil/Pub/MGEN/dist/

[92] TTCP traffic generator: available via ftp://ftp.arl.mil/pub/ttcp/

[93] Almersberger, W., Salim, J., H., Kuznetsov, A., “Differentiated Service on Linux”, draft-almersberger-wajhak- diffserv-diffserv-linux-01.txt, May 1999. Available via http://lrcwww.epfl.ch/linux-diffserv/

[94] TC Traffic control tool, available via, available via ftp://ftp.inr.ac.ru/ip-routing/ version iproute2-2.2.4-now-ss991023.tar

[95] Almes G., Kalidindi S., Zekauskas M., ”A One-way Delay Metric for IPPM” RFC 2679 (Standard Track). September 1999

[96] NTP (Network Time Protocol) available via http://www.eecis.udel.edu/~ntp/

[97] Demichelis C., Chimento P., “IP Packet Delay Variation Metric for IPPM” draft-ietf-ippm-ipdv-08.txt Internet Draft (Work in progress)

[98] Costa G., “IntServ over DiffServ for IP QoS in Radio Access Networks”. Master’s Science thesis. University of Helsinki, Department of Computer Science. April 2002.

12 Acknowledgments

I would like to warmly thank my supervisors, Professors Sergio Palazzo and Kimmo Raatikainen who gave me the opportunity to prepare my thesis in Finland and for their invaluable directions and advice that made this thesis much better.

Many thanks also to Jukka Manner, for his suggestions and his precious and unfailing help along the entire course of the thesis. I would also like to thank all in the Computer Science Department of University of Helsinki for making this an enjoyable and stimulating year work.

I am also debtor to Davide e Giovanni, for their moral and practical support and encouragements during this period.

Last but not least, thanks to my family, even if physically far, I always felt them as if they were in Finland with me.