Adaptive Multimedia Multi-party Communication in Ad Hoc Environments

Pedro M. Ruiz
and Juan A. Sanchez
and Emilio Garcia
Agora Systems S.A, Aravaca, 12 3-B,
28040 Madrid, Spain
Email: pedro.ruiz@agoratechnologies.com,
jantonio.laguna@agoratechnologies.com,
emilio.garcia@agoratechnologies.com

Antonio F. Gomez-Skarmeta
and Juan A. Botia
University of Murcia,
Campus de Espinardo,
30100 Espinardo (Murcia), Spain
EMail:skarmeta@dif.um.es
juanbot@um.es

Andreas Kassler
and Teodora Guenkova-Luy
Department of Distributed Systems
Faculty of Computer Science
University of Ulm, Germany
Andreas Kassler is also with SCE,
Nanyang Technological University, Singapore
Email: kassler@ieee.org
guenkova@vs.informatik.uni-ulm.de

Abstract*— Adaptivity is a key issue for enhanced multimedia communication in wireless environments, where the network QoS for a whole session cannot be guaranteed. Within this paper we propose a set of mechanisms to provide adaptive multimedia multi-party communication according to user-defined QoS levels over self-organizing ad hoc network extensions connected to IP access networks. We use special extensions to SIP/SDPng based on XML to negotiate alternative application defined QoS levels and adaptation paths and optionally couple local resource management with network layer QoS. At the network layer, we propose the MMARP (Multicast MANET Routing Protocol) which provides efficient multicast communications within ad hoc network extension to public infrastructure based access networks. MMARP is able to deal with the complexity of supporting traditional IP nodes whilst inter-operating smoothly with fixed IP networks. An adaptive application architecture is demonstrated that uses the E2ENP and allows multimedia applications to reconfigure themselves in real-time to match the specific network conditions in order to preserve the user-perceived QoS. Thus, an integration of application layer QoS as defined by the user and network layer QoS is achieved.

I. Introduction

Ad hoc networks offer connectivity in highly dynamic environments, where routers can move and signal quality changes rapidly. The variation of resource availability is a real challenge for the delivery of high quality real-time multimedia streams. In such environment, the Quality of Service (QoS) is hard to maintain during the whole session. Using network-layer reservations is not the best idea as the topology changes frequently. Therefore, applications should be able to adapt to changes in network conditions, resource availability and thus cope with short-term and long-term QoS violations. Application layer adaptivity can be seen as complement to any network layer QoS mechanisms, which should allow applications to preserve well defined user-perceived QoS even when the available network resources vary during the session. Typical examples of application adaptation is to change frame-rate, frame size, visual quality (by change e.g. quantization) or even to switch to a different encoding scheme that consumes less bandwidth at the expense of lower quality.

The idea of using adaptive applications is not new in fixed network environments (see e.g. [27]). However, the variation of QoS delivery at network layer is much higher in wireless and mobile scenarios. Besides congestion there are many other factors which affect the user-perceived QoS e.g. fading, mobility, multipath, vertical handover etc. Several other proposals focus on wireless networks ([13], [2], [19], [20]). Their requirements are very difficult to be met in multi-hop ad hoc network extensions. One of the most important tasks is to minimize application reconfiguration time in order to allow fast but also flexible adaptation. Adaptation should be carried out to match resource availability (end-system and network) with subjective quality.

Future network topology will be based on an IP core network, where several access networks are attached to. Each access network (e.g. based on UMTS, 802.11) provides access to packet delivery services of the core, forming thus an 'all-IP' and 'beyond 3G' mobile network. In such an environment, the number of user terminals is high and many services (e.g. paging) are available that benefit from multipoint communications in order to reduce bandwidth consumption for group communications. Typically, access networks are infra-structure based. In addition, ad hoc extensions connect a truly mobile user to an access network and thus to the internet core. Such ad hoc extensions are self-organising multihop networks, in which a user terminal employs those of other users as relay points to provide multihop paths between the mobile nodes and the fixed network architecture. Therefore, it is important to provide efficient IP multicast communications in such extensions, which is not easy to achieve.

The typical IP multicast protocols used in the Internet consists of the IGMPv2 [7] protocol for group membership in combination with PIM-SM[6], which is in charge of the IP multicast routing. These protocols are not able to offer a good performance in ad hoc network scenarios. The quick and unpredictable link changes which characterise ad hoc networks would cost these protocols a high overhead to keep efficient routing paths. Other multicast routing protocols (e.g. CAMP[8], ODMRP[16], ...) have been proposed particularly for ad hoc networks. These protocols incorporate specific mechanisms that enable them to efficiently operate in scenarios with the particular characteristics of ad hoc networks; however, they are only suitable for isolated ad hoc networks and can neither interoperate with a fixed IP network nor support standard-IP multicast sources or receivers.

In this paper we propose a set of mechanisms to provide adaptive multimedia multi-party communication according to user-defined QoS levels over self-organizing ad hoc network extensions connected to IP access networks. We manage QoS end-to-end from the users and media point of view.

Well-defined adaptation strategies are invoked, if QoS delivery at the network level is violated (e.g. due to vertical hand-over). Special extensions to SIP/SDPng are used to negotiate alternative application defined QoS levels and adaptation paths together with local and network resource management. Several phases structure the negotiation protocol to optimize negotiation flexibility. At the network layer, we propose the Multicast MAnet Routing Protocol (MMARP), a new multicast ad hoc routing protocol which is able to deal with the complexity of supporting traditional IP nodes whilst interoperating smoothly with fixed IP networks. MMARP nodes are able to intercept and process IGMP messages. They further permit standard-IP nodes to participate in IP multicast communications as they do when attached to a fixed IP network. We believe that a combination of adaptive applications, together with an optimized negotiation and signaling protocol allows flexible and efficient application layer QoS provisioning in such heterogeneous and dynamic environments.

The rest of this paper is organised as follows: Section II discusses the problem of negotiating and managing QoS in distributed multimedia systems. It also introduces the E2ENP - a protocol that supports the negotiation of capabilities and QoS parameters at different layers. Section III shows, how adaptive applications should be structured and presents our adaptation logic, which is based on well defined and negotiated adaptation paths. Section IV introduces multicast routing in ad hoc network extensions and our solution. Section V presents empirical results from the E2ENP, MMARP and adaptive application architecture implementations, and finally section VI concludes the paper.

II. QoS Negotiation and Management in Distributed Multimedia Systems

We first present a system model of a distributed multimedia system (see Fig. 1). Typically, an end-device runs an application to communicate with its peer. The application uses services provided by the Operating System or an intermediate middleware. During the multimedia computing process, timely access to local resources is essential and packets are sent using one or more network devices to intermediary network nodes, that process these packets. System software on (fixed or mobile) network nodes is responsible for e.g. routing and admission control functionality and the nodes typically manage their resources (packet queues,...). Within this model several abstraction levels of QoS can be identified. Ideally, resources on end systems, network nodes and software interactions must be properly managed and orchestrated to provide predictable end-to-end QoS.

A. QoS Parameters and Layers

The following types of QoS parameters are identified (see also [28] for comparison):

End-to-End perceivable QoS: This set of parameters defines the end user view on the performance of the distributed application. Typically, such information is not expected to be very detailed as a precise quantitative description might be very complex. However, the user may specify presentation features of the application as user's QoS preferences (e.g. "motion rated most important") and target quality levels (e.g. "high-quality video") in a "descriptive" style. Such parameters should not be subject to negotiation as different users may have different technological

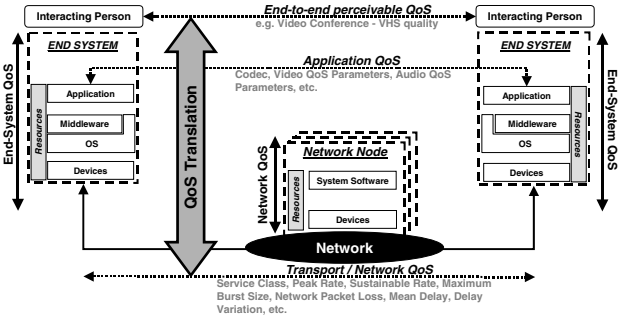

Fig. 1. Systemmodel for distributed multimedia application

background and understanding what "good" or "bad" QoS means. The translation of the perceivable QoS characteristics in more technical terms is typically implemented by the application.

Application QoS: These parameters are used to describe end-to-end application performance (like video frame rate, size, or visual quality) and are are subject to negotiation with the peer. We assume intervals for single parameter dimensions, inspired by Alfanos psycho-visual study [1]. An application thus defines a set of n-dimensional QoS parameter cubicles (e.g. frame-rate between 5 and 7 fps, framesize=CIF, visual quality between 90% and 94%). Each cubicle is also denoted as QoS Contract which is used by the system to set up a multimedia session and enforce application QoS. An adaptation process can then be seen as the well defined transition from one cubicle to another. For example, if the QoS Contract "AQC1" cannot longer be fulfilled (e.g. due to handover), "AQC2" should be enforced and so on. The system should derive all possible Application QoS parameters based on available system resources and user QoS preferences in form of QoS Contracts. Intermediaries (e.g. SIP proxies) may use these QoS Contracts together with capabilities (e.g. codecs) to derive, negotiate and enforce Transport/Network QoS parameters on behalf of those end-systems which are not able to explicitly signal (e.g. using RSVP [17]) or implicitly request a specific packet scheduling/droppping discipline (e.g. based on DiffServ [3]) configuration for e.g. access/core network.

Transport/Network QoS: These parameters are used to describe the end-to-end requirements of the application with respect to network packet delivery. The system must derive these parameters based on the actual capabilities used, Application QoS parameters and media characteristics used as an input to the compression algorithm. This may be easy to achieve for audio as a codec together with its Application QoS parameters defines exactly the values for Transport QoS. However, dealing with variable bitrate video codecs often requires traffic models to derive a set of parameters that may change based on the scene complexity. Transport/Network QoS parameters are then signalled to transport domain entities for reservation using e.g. RSVP [17]. Intermediaries (like SIP proxies) may apply this Transport/Network QoS information to reserve network resources on behalf of the end-systems which are either not allowed to do so or do not have OS support for network QoS signaling.

*End-System QoS:*These parameters are used to specify local resource requirements (e.g. for memory, CPU or battery power).

The local system must be able to manage its resources and provide enough capacity to fulfill application layer QoS locally. Therefore, such parameters are not subject to negotiation.

In addition to the QoS parameters described above, end-systems negotiate and agree on a common set of input/output configurations (like addresses and ports) and capabilities (like codecs, RTP packetization rules,) for the multimedia streams in order to set up a valid end-to-end multimedia session.

B. Hierarchical QoS Specification and QoS Adaptation

Multimedia sessions may contain one or several basic media streams (i.e. audio, video, data). In our model, every single media stream can be associated with one or many Application QoS Contracts. Every QoS Contract corresponds to an unique QoS configuration of this stream. Application QoS Contracts for a single basic stream form thus a directed graph that describes the desired adaptation behavior. The graph nodes are associated with the single stream QoS Contracts and the edges thereof with the respective adaptation condition.

For multistream multimedia sessions the users may wish to specify not only the desired QoS for each single stream, but also any parameter that determines inter-stream requirements like synchronization. Additionally, correlation may be required between streams belonging to the same logical context. Multi-stream Time Synchronization and QoS Correlation constraints are modelled also as QoS Contracts at a higher abstraction level (denoted as QoS Contexts). Additionally, it could be necessary to define additional constraints, e.g. that the whole stream bundle does not exceed a certain aggregated bandwidth. The model for correlating QoS Contracts allows the recursive building of high-level Application QoS Contracts - leading to a hierarchical QoS Specification scheme in a treelike structure. Each leaf of the tree represents a single Application QoS Contract associated with a single media stream, e.g. audio, video, data. An upper level parent node is associated with an Application QoS Context, grouping the Application QoS parameters of its children and specifying their Time Synchronization and QoS Correlation constraints.

Time Synchronization and QoS Correlation conditions for related media streams may change over time. This requires the definition of multiple QoS Contexts for a multistream session, which specify the QoS adaptation of the session as a whole. As for the stream QoS Contracts, the QoS Contexts also form a directed graph that describes the adaptation behavior of multi-stream multimedia sessions. A video streaming application can thus specify and negotiate that e.g. a vertical handover to a lower bandwidth access technology should result in dropping the video stream and only the audio should be further played at a reduced quality and bandwidth.

C. End-to-end QoS negotiation protocol – E2ENP

The End-to-End QoS negotiation protocol (E2ENP) is designed to support the negotiation of capabilities, configurations and QoS parameters at different layers for multi-stream multimedia applications. It enables the dynamical and flexible application of negotiated information for QoS adaptation purposes during ongoing multimedia sessions. By using the E2ENP it is possible to integrate provider subscription management of the users, network QoS reservations and mobility management. E2ENP uses extensions of the IETF SDPng XML description forms [15] and is based on the *Offer-Answer Model* [22], [5], [23] of the Session Initiation Protocol (SIP).

The idea behind E2ENP is that end-peers exchange complete information about the QoS they can and would like to support based on their static capabilities, hard- and software limitations. Application QoS, transport QoS and capabilities are separated to allow the definition of flexible combinations. A specific codec (e.g. H.263) is then associated with one or more Application QoS Contracts (e.g. frame rate between 10 and 15 fps, frame size QCIF,...). Each association can be bound to a specific transport QoS Contract (e.g. peak band width, delay, jitter, packet loss). The E2ENP design considers application and transport QoS parameters as two orthogonal issues of the QoS provisioning step.

As E2ENP messages are exchanged within the service domain, provider intermediary entities (like enhanced SIP proxies) may inspect the E2ENP messages and perform admission control on session level. Each network provider is thus free to indicate, if the given QoS Contract (either application of transport) is supported for the given session (exchanged during the negotiation phase) or if the capability or static configuration (exchanged during a pre-negotiation phase) is applicable based on e.g. the users subscription. It is important to note that the provider should only mark those contracts as applicable or not and does not delete the corresponding configuration information. Thus, the providers limitations are propagated to the remote peer and both peers can mutually agree to switch to a different provider that offers a better quality or use the available information during a vertical handover process. If a provider by intercepting the peer signaling would delete the information that the provider does not support, the remote peer would never know that the initiating peer would like to use this configuration. However, after the handover the new provider might be able to support the high quality conference, and in our model, the peers simply would issue a re-negotiation phase to upgrade the quality of the session.

1) E2ENP phases: The E2ENP consists of the following three phases: *Pre-negotiation Phase:* In this phase, peers agree on a common set of capabilities, static configuration information (like maximum supported data rate 64 kbps), and general user QoS preferences (like streaming at maximum 10 fps with codec X), independent of any session. Based on the availability of pre-negotiated information, session establishment can be accelerated as only QoS Contracts would be offered that can be supported principally; *Negotiation Phase:* During this phase, configuration information is exchanged together with QoS Contracts for a specific session to be established. Examples are the number of media streams that belong to the specific session, the possible codecs used to compress the streams, the Application layer QoS Contracts for a given stream, the QoS Correlation and Time Synchronization constraints as well as the Network layer QoS Contracts. QoS Context is formed that describes a complete configuration necessary for the media session establishment and QoS adaptation process during session lifetime. If end-peers cannot (due to limited memory) or are not allowed to (due to system policies) perform Pre-negotiations, static capability information (see pre-negotiation phase) can be included and negotiated, which leads to a combined Pre- and negotiation phase; *Re-negotiation Phase:* Once a session is established, QoS aware and adaptive

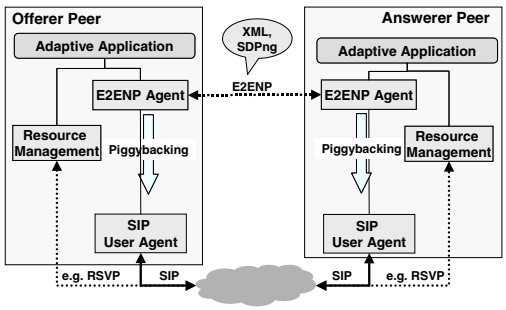

Fig. 2. E2ENP and its relation to SIP and adaptive applications

applications have to monitor QoS delivery and decide, when, how and to what extent to adapt. If a change of configuration or QoS should be enforced, this phase allows to switch to a new QoS Contract, signal a codec change, etc. As any prior negotiated information (gathered during the pre-negotiation or negotiation phase) is already available, information exchanged during this phase can be represented in a compact way by referencing. This phase can also enhance the common QoS configuration of end-peers with new QoS Contracts (if streams ceased to exist thus freeing local and network resources), or capabilities (e.g. if the system upgraded due to dynamical codec download).

2) E2ENP Implementation Issues: The E2ENP is designed to be network and system independent, since its main purpose is to describe system capabilities, QoS Contracts and QoS Contexts in a declarative manner. The E2ENP uses Protocol Data Units (PDUs) to convey system and session configuration information. The information of a PDU can be referenced from within another PDU, thus providing information links. E2ENP does not define its own transport protocol. Instead, it can use already available protocols like SIP. An E2ENP Agent (see Figure 2) implements the E2ENP state machine and PDU handling and provides interfaces towards an adaptive application for initiating a given negotiation phase. E2ENP defines its own addressing [18] to uniquely identify E2ENP negotiation sessions and the communicating peers at application level.

Negotiation and Re-negotiation phases are interleaved with *optional* local and network resource reservation. Adaptive applications provide hooks for the E2ENP agent. They are thus informed, when it is beneficial to allocate local resources to ensure end-to-end QoS. Adaptive applications should check, if enough resources are available locally to support a given QoS Contract/codec combination before making any offer that contains the proposed contract to the negotiation partner. Once the offer is received, the remote peer should also perform local admission control for all received QoS Contracts to determine an appropriate answer that contains mutually supportable QoS Contract/codec combinations.

The E2ENP agent can also coordinate the start/end of the network reservation. But it is then up to the application to issue e.g. RSVP PATH and RESV messages. If no end-to-end QoS signaling is possible, E2ENP provides special commands that can be used to signal the remote peer that the call-setup should be continued without resource reservation. Thus the protocol is very flexible and can be used with or without resource reservation

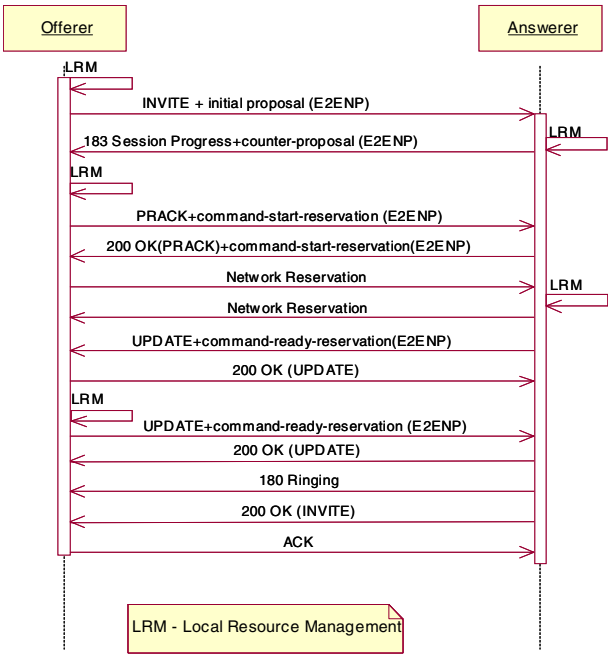

Fig. 3. Negotiation Phase in Push Mode

(both locally and network).

3) E2ENP message sequence for negotiation phase: The negotiation phase is performed before or at the actual start of a session. Based on results of previously applied phases, the end-systems agree which capabilities and QoS Contracts to enforce for the given streams within the session. At completion of this phase, peers have agreed upon the QoS Contracts they are going to enforce for the given streams and the right set of capabilities to apply. The Offerer and Answerer are network peers (see [18]). Each phase can be executed in either *Push mode*, where the Offerer sends an initial proposal to the Answerer, or in the *Pull mode*, where the Offerer sends an empty bid and the Answerer starts with proposed contracts. More information on negotiation modes can be found in [22]. Figure 3 explains the *Push mode*, where the Offerer sends an initial proposal that contains its desired capabilities and QoS levels together with the potential stream adaptation paths in a SIP INVITE to the Answerer. As the Offerer knows its desired QoS levels, it optionally reserves local resources beforehand. The Answerer proves the Offerer's proposal and also reserves appropriate resources, locally. The Answerer replies (183 Session Progress) with a sub-set of capabilities and QoS parameters. This information matches both the Answerer and the Offerer views about the session establishment and future QoS adaptations. The Answerer may also reply in its counter-proposal with additional capabilities and QoS that the Answerer would like to apply for the session, thus enabling the Offerer side to upgrade its capabilities dynamically, e.g. by downloading appropriate codecs. By receiving the Answerer's counter-proposal the Offerer proves locally if the resources need to be relocated to match the Answerer's reply. The Offerer sends PRACK to the Answerer to inform it about the start of the network reservation and the Answerer acknowledges this action.

Network resource reservation proceeds at the transport/network layer using e.g. RSVP. The next sequence depends on the model used for network resource reservation (sender initiated/receiver initiated). If RSVP is not available end-to-end or terminated at the network edge, RSVP PATH will not be received by the Answerer. Therefore, by using UPDATE the Offerer and the Answerer inform each other about the state of the network reservation and about the final capability/QoS level to be enforced for the session to be established. Once network resources are reserved end-to-end or only partially, the Answerer sends a 180 RINGING to the Offerer. The call setup has completed by sending a 200 OK for the initial INVITE from the Answerer to the Offerer. This is acknowledged by the Offerer sending an ACK.

There may be situations, where the local resource reservation may fail, e.g. at the Answerer after receiving the initial proposal. As the initial proposal may contain alternative QoS levels and alternative capabilities that the Offerer would also tolerate (within the adaptation path), the Answerer may chose an alternative capability/QoS level. This information must be propagated in the 183 Session Progress that the Offerer is able to relax resource requirements and reconfigure its media stream engine. Also, network resource reservation may fail due to insufficient resources. In this case, the Offerer would try an alternative QoS level with less network resource requirements until resources are reserved successfully. The final agreed upon QoS level must be notified to the Answerer within the UPDATE message so as to release additional resources. E2ENP itself does not provide messages for resource reservation. E2ENP only coordinates the network resource reservation, which should be accomplished using e.g. RSVP triggered by the hooks provided through the E2ENP agent (i.e. the reception and sending of *command-start-reservation*).

Re-negotiation message sequence is similar to Negotiation, but has less signaling overhead due to the E2ENP referencing mechanism. Re-negotiation can be started by any of the peers detecting a QoS Violation, thus the initial Offerer may become by Re-negotiation an Answerer and vice versa.

There may be situations, where a service domain entity (e.g. enhanced SIP proxy) is involved in the negotiation process by e.g. intercepting the 183 Session Progress in order to authorize network resources on behalf of end-systems. In this situation, the proxy would only indicate which QoS Contracts/capabilities can be supported.

4) E2ENP Messages: E2ENP messages are transmitted as payload of SIP messages. E2ENP itself consists of header and payload. E2ENP enhances the XML description forms of the IETF SDPng [15] schema. A more flexible scheme is introduced to enable the separate specification of capabilities, application and transport QoS parameters and their association. E2ENP also enables linking and mutual referencing of negotiated information to form a compact representation. The full specification, including all XML statements, can be found in [18].

Capabilities and QoS Contracts at Application and Transport Layer are typically exchanged between end-peers during the Pre-negotiation phase but may also be part of a negotiation phase, if pre-negotiation is not carried out. Once a session should be established, peers have to bind application QoS, transport QoS and capabilities and agree on session configuration information. They do so by referencing previously defined QoS Contracts and

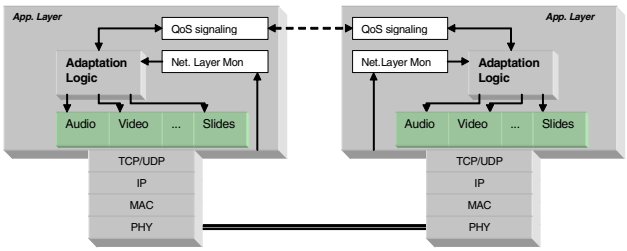

Fig. 4. Framework for Adaptive Applications

alternative configurations per media stream. E2ENP messages also allow to define adaptation possibilities for single streams, the association between streams (i.e. QoS Context) and the adaptation possibilities of the whole multimedia session. Once the session is established, it is up to adaptive applications (or middleware, [9], [21]) to monitor QoS delivery and react to QoS violation. E2ENP messages then allow to reference negotiated data in order to provide peers with signaling mechanism to switch to a new capability/QoS Contract binding during such re-negotiation phase (which can also interleave with resource re-reservation). Typically, the re-negotiation PDU content is much smaller because it references information from previous phases by using identifiers instead of full descriptions. This saves bandwidth and reduces re-negotiation time. However, E2ENP also supports the full re-negotiation mode, where complete QoS Contract definitions can be exchanged. The MSC for the Re-negotiation phase can be found in [18].

III. Real-Time Adaptive Multimedia Applications

As a complement to the E2ENP, we have deployed an adaptive middleware, which allow multimedia applications to adapt in real-time to the specific network conditions. These adaptations take in account not only the network conditions but the QoS Contracts which have been previously negotiated with the E2ENP. The main items in this architecture are shown in Fig. 4. The QoS signaling mechanism is the protocol in charge of sending and receiving reports describing the network conditions from the other end. When such report is received it is passed to the Adaptation Logic as an additional input. The Adaptation Logic is in charge of deciding which set of parameters is best suited to the current network conditions. It therefore decides, *when*, *how* and to *what extent* to adapt based on information negotiated with E2ENP.

In true heterogeneous multicast scenarios, where capabilities and QoS requirements of receivers differ, we propose to tailor the media streams to match receivers capabilities and QoS requirements by using content adaptation nodes. Several nodes can be deployed to form an application layer overlay adaptation and transcoding network. If a layered encoder was used, such adaptation process is a simple layer drop/add decision. Otherwise, more complex steps are necessary, which may even result in transcoding. These nodes enable receivers to join an ongoing conference, even if no agreement on capabilities can be met. Nevertheless, the E2ENP can be used to determine, what transcoding steps are necessary. More information on heterogeneous multicast can be found in [12]

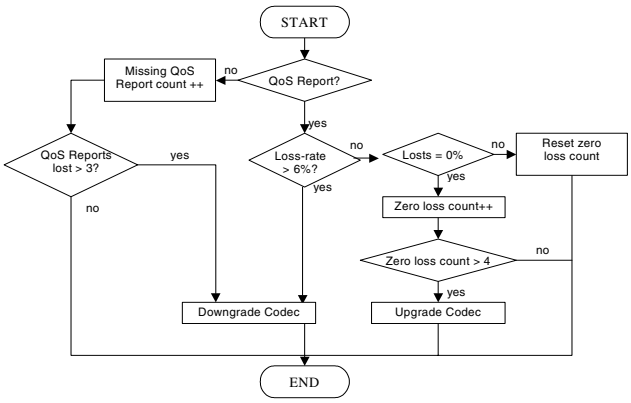

Fig. 5. Schematic diagram of the adaptation logic

A. Adaptation Logic

The main problem at the application layer with respect to mobile and wireless networks is that of adapting data transmitted to or received from the application to the characteristics of the different networks, including throughput and delay concerns. Therefore, the Adaptation Logic can be seen as a somewhat complex function, which receives inputs (e.g. QoS Contracts, jitter, bandwidth, loss-rate, user-preferences, etc.) and produces a combination of application settings like audio codec, video codec, frame rate, video size, etc. These settings are then used by the applications until the adaptation logic decides that new adaptation is required. The complexity lies in making the output consider the subjective aspects of the user's perception of QoS and their concrete preferences.

In general, packet losses are the major cause of degradation in the user-perceived QoS, and therefore one of the most important input parameters for the adaptation logic is the end-to-end loss rate per reporting period. End-to-end delay problems may also make the user experience bad quality. This could be avoided with adaptive buffer management without reducing the bandwidth that the application uses.

The detailed description of the Adaptation Algorithm which is applied to every QoS report received is presented in Fig. 5. As shown, a downgrade in the quality will be only performed when the end-to-end packet loss-rate exceeds 6% or 3 consecutive QoS Reports are lost – possibly due to congestion or interference. Additionally, the quality is upgraded whenever 4 consecutive QoS reports indicating 0% packet loss arrive. These parameters have been set according to our own experience. The complexity of calculating them automatically lies on the existence of subjective components in the user-perceived QoS which may vary from person to person and which are very difficult to model mathematically. For the automatic calculation we are currently exploring the the utilization of machine learning algorithms.

B. QoS Signaling Mechanism

The QoS signaling is another major component in our adaptation architecture as long as it is used to get instantaneous feedback on the quality perceived at the other end. It is basically an end-to-end transport mechanism for signaling data; no special protocol is needed. In a future revision, we will use a more sophisticated approach based on well established standards like RTCP receiver reports. In our implementation this quality will be measured in terms of loss rate and mean delay experienced by the data packets in the network. This information is carried in a special signaling packet called 'QoS Report', which is used by receivers to send back to the sender information on network status and quality of packet delivery.

Some experiments performed in [14] demonstrate that a UDP transport is much more appropriate to carry the feedback than a transport using TCP. TCP retransmissions result in stale QoS information especially on a congested network. An additional issue is that the feedback packet itself has to traverse the network back to the server, and the probability of it actually making it there on time is inversely proportional to its importance. That is, a feedback packet is most important when it carries information about a congested network and it is not important when it is just saying that all is going well. There are at least two approaches for solving that: prioritization of QoS reports, and periodical sending of reports. The first one is related to the idea of giving a higher priority in every router within the intermediate network to these signaling messages. In this way, when intermediate routers have data to send, the will firstly forward signaling messages and then the rest of the data packets. Thus signaling messages are given a higher probability to reach the destination. This mechanism works very well but its main drawbacks are related to its difficult implementation – especially in ad hoc networks. We have used the second approach which is based on the idea of using periodic reporting, by which the clients send periodic and sequenced reports towards the sources. This way, whenever network problems come up, the sender can detect missing reports. The sender uses the heuristic of downgrading the quality when a certain number of QoS Reports are lost. We demonstrated [26] this approach to be very effective in scenarios with abrupt changes in bandwidth.

IV. MMARP for Ad hoc Network Extensions

Finally, to complete our proposed set of mechanisms for adaptive multimedia and multi-party communications, we present our ad hoc multicast routing protocol called MMARP[25].

The MMARP protocol is especially designed for mobile ad hoc networks (MANETs). It is fully compatible with the standard IP Multicast mode and it allows standard IP nodes to take part in multicast communications without requiring any change because MMARP supports the IGMP protocol as a means to inter-operate both with access routers and standard IP nodes. The inter-operation with access routers is performed by the Multicast Internet Gateways (MIGs) which are just MMARP nodes situated just one hop away from the fixed network. Every MMARP node may become a MIG at any time. The only difference between a MIG and a normal MMARP node is that the MIG is responsible for notifying the access routers about the groups memberships within the ad hoc fringe. The mechanism allows MMARP to work with any IP multicast routing protocol in the access network and, therefore, it shields the MMARP operation from the protocols performing the intra-domain or inter-domain multicast routing.

For the remaining text we use the terms standard IP source or standard IP receiver to refer to a traditional IP Multicast source or receiver and we use the term ad hoc sender or ad hoc receiver

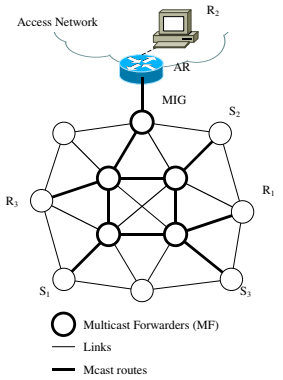

Fig. 6. Multicast mesh after request/reply phase

to refer to pure ad hoc nodes. MMARP uses a hybrid approach to construct a multicast distribution mesh. Multicast routes among ad hoc nodes are established on-demand, whereas multicast routes to the multicast sources in the fixed network are established proactively.

A. Overall operation

MMARP uses a mesh-based distribution structure, similar to the one used by ODMRP, which offers good protection against the mobility of the nodes (see Fig. 6). Both the proactive and reactive parts of the protocol are responsible for building the mesh.

The reactive part consists of a request phase and a reply phase. When an ad hoc node has new data to send, it periodically broadcasts a MMARP_SOURCE message which is flooded within the entire ad hoc network to update the state of intermediate nodes as well as the multicast routes. These messages have an identifier which allows intermediate nodes to detect duplicates and avoid unnecessary retransmissions. When such message is received by an ad hoc node for the first time, it stores the IP address of the previous hop and rebroadcasts the packet. When one of these messages arrives at a receiver, or at a neighbour of a standard IP receiver, it broadcasts a MMARP_JOIN message including the IP address of the selected previous hop towards the source. When an ad hoc node detects its IP address in an MMARP_JOIN message, it recognises that it is in the path between a source and a destination. It then activates its MF_FLAG (Multicast Forwarder Flag) and rebroadcasts a MMARP_JOIN message containing its previously stored next hop towards the source. In this way, a shortest multicast path is created between the source and the destination. When there are different sources and receivers for the same group, the process results in the creation of a multicast distribution mesh. (see Fig. 6).

The proactive part of the protocol is simply based on the periodic advertisement of the MIGs as default multicast gateways to the fixed network. As the TTL of IGMP messages is fixed at one, the reception of an IGMP Query can be used by ad hoc nodes to detect that they are MIGs and activate its MIG_FLAG. MIGs periodically broadcast a MMARP_DFL_ROUTE message which is flooded to the whole ad hoc network. The reception of this message informs intermediate nodes about the path towards multicast sources in the access network. When the MMARP_DFL_ROUTE message reaches a receiver or a neighbour of a receiver, this node initiates a joining process similar to the one that we have just described for the reactive approach. When the MIG receives the MMARP_JOIN message, it then sends an IGMP Report towards the FHMR, ensuring the IP multicast data from sources in the fixed network reach the destinations within the ad hoc network extension.

If a standard IP node becomes an active source, the process for creating the distribution mesh is similar except that the MMARP_SOURCE message is actually generated by the neighbouring ad hoc nodes which receive the data packets from the source.

The protocol incorporates local repair mechanisms to overcome a link break during the creation of the distribution mesh. Whenever a node is unable to deliver a MMARP_JOIN message to its next hop after four retries, it broadcasts a MMARP_NACK message to its one-hop neighbours. Upon the reception of this message, the neighbours use their own route to reach that next hop. Should any of them not know an alternate path, they repeat the process until a path is found. Although this recovery process does not offer optimal routes, it offers a quick recovery before the next topology refresh. Once the mesh is established, the data forwarding is very simple: data packets addressed to a certain multicast group are only propagated by ad hoc nodes which have their MF_FLAG active for that group. When such a data packet arrives at a node whose MF_FLAG for that group has not expired, it checks that it is not a duplicate and in that case retransmits the packet. In any other case the packet is dropped.

B. Support of standard IP multicast protocols

The protocols used by standard IP nodes to perform their basic operation (such as ARP, or IGMP) were designed to operate in BMA (Broadcast Medium Access) networks. However, in multihop ad hoc networks, the link layer has a different semantic. The neighbours of a node are able to receive the frames it sends but it is not guaranteed that they are able to directly communicate among all of them. In traditional ad hoc routing protocols without explicit support for standard IP nodes this is not a problem because each ad hoc node sends its own source announcement or join message.

In order to be compatible with the standard IP multicast model, MMARP nodes in the neighbourhood of a standard IP node have to send MMARP_SOURCE or MMARP_JOIN messages on behalf of the standard IP node. This means that messages generated by standard IP nodes, may be received by all neighbours and processed independently, creating unnecessary paths. The MMARP protocol has been designed to avoid unnecessary generation of these messages. It includes a field in its header that facilitates the identification of the node which actually triggered the sending of the control message; this allows ad hoc nodes to identify all the MMARP packets which are triggered by a specific standard IP node, independently of the ad hoc neighbour that actually generated it. Thus, ad hoc neighbours of standard IP nodes and intermediate ad hoc nodes are able to detect these types of MMARP_SOURCE and MMARP_JOIN messages as duplicate and avoid the creation of unnecessary paths.

V. Empirical Results

In order to evaluate the effectiveness of our proposal, we have set up a real MMARP-based ad hoc testbed, on which we will compare the performance of real-time videoconferencing both with traditional applications and with *adaptive applications*. The testbed has been deployed in the basement of the CS Faculty at the Univ. of Murcia. The route has been specifically selected so that link breaks and MMARP route changes take place during the videoconferencing session. Furthermore, the signal strength changes due to the variation of the distance to MMARP nodes and the number of intermediate walls to traverse. This makes the available bandwidth vary during the session.

The trials have been performed using our MMARP implementation for Linux. It is a user-space daemon which handles MMARP packets before they are processed by the TCP/IP stack. In addition, we have also extended the RTP-based ISABEL-lite videoconferencing application to use our adaptive application framework. The settings which are used by the application as adaptation steps are given in Table I. In order to guarantee a fair comparison of both approaches, the adaptive application starts with the quality step number 3, which is the only one used by the non-adaptive application, and is around the mean bandwidth which we calculated during the whole session. The results which we present are extracted from the RTP traces which are generated by the videoconferencing application. We have used the same route, at the same speed and in the same network conditions for the adaptive and non-adaptive trials.

The results presented in Fig. 7(a) show that the use of adaptive applications is able to reduce the overall packet losses both for audio and video to approximately 1/3. As expected, the differences are higher in the periods in which there is less bandwidth available. This is also noticed in the variation of the delays depicted in Fig. 7(b). In the same critical periods, the non-adaptive approach is not able to control the growing of the end-to-end delay, whereas the adaptive one is able to quickly restore the original state.

The overall packet losses is a good reference to identify the points of the trial in which the network conditions are most critical. This is identified by an increase in the slope of the total packet loss curve. However, what really affects the user perception of QoS is the instantaneous loss-rate, which is what causes the service disruptions. For example, a 20% packet loss-rate can be considered as the point in which an audio flow is perceived with poor quality.

In Fig. 7(c), we compare the statistical histogram for the distribution of the audio loss-rate for both approaches. The same statistical analysis is performed for the video flow in Fig. 7(d). For example, for the audio flow, the adaptive application approach is able to keep the loss-rate below 10% all the time. In fact, it keeps the loss-rate below 5% during the 91% of the time. For the video flow, the loss-rate is kept under the 5% the 64% of the time, and its has been under the 10% the 78% of the time.

We were also interested in how long it takes to signal QoS Contract and capability changes to the remote party. Therefore, we evaluated the E2ENP which was implemented (E2ENP agent) completely in Java, including a simplified SIP stack. The agent hosts the E2ENP state machine, a parser that translates from E2ENP payload to system internal object representation and a generator for the reverse step. We used the SAX XML parser (`http://www.saxproject.org/`) which generates events for each processed XML tag. The agent is connected to the adaptive application via a well defined API [18].

TABLE II
Parser and generator processing

Input size	Parser Processing	Generator Processing
100	17.33	6.20
500	51.42	27.64
1000	105.76	44.37
5000	306.67	186.74

Table II shows the time (in ms) needed by the parser and generator to process the E2ENP XML content, depending on the number of XML tags (input lines) within the PDU. It should be noted that, depending on the negotiation mode, a typical E2ENP PDU contains maximal 100 lines. Therefore, parsing and generating the XML representation is reasonably fast and scales with the number of lines processed.

Finally, we were interested in the total time to perform each phase. The duration of the negotiation phase corresponds then to the call-setup time. We ran an experiment where we connected two machines over a router, which runs the network emulator NistNet (`http://snad.ncsl.nist.gov/itg/nistnet/`).
For this experiment, we considered a multimedia scenario (one audio and one video stream) with three different audio and three different video codecs as capabilities. These codecs are used to build two different QoS contexts for defining the complete session adaptation path. This resulted in a E2ENP PDU sizes of 5 KB (Pre-negotiation), 4 KB (Negotiation), and 1 KB (Re-negotiation). Nistnet was configured to emulate different network types by changing bandwidth availability and delay. The following bandwidth limitations where used: 16, 64, 170, 384, 1500 kBit/s and 100 MBit/s. End-to-end delay was set to 60, 60, 45, 45, 10, 5 ms respectively corresponding to typical network types GSM, GPRS-1, GPRS-2, UMTS, WLAN, and LAN.

Figure 8 shows the total execution time (including parsing, generating XML, statemachine processing, sending and receiving SIP messages) of each phase depending on the type of simulated network. The MSC in Figure 3 was used for the negotiation phase but no RSVP messages have been exchanged. The MSCs for all other phases can be seen in [18]. Note, that one negotiation phase consists of multiple roundtrips and each E2ENP PDU processing requires one parsing step and state machine processing within the agent. While for high bandwidth connections the overhead for transmitting E2ENP PDUs can almost be neglected, this is not true for low bandwidth links. However, E2ENP PDUs may be compressed by any loss-less text compression scheme. Note, that call-setup time varies between 600 ms in the LAN environment, 1 s in UMTS environment and around 8 s in GSM environment. Re-negotiation in the LAN(WLAN) environment is around 400(450) ms and 650 ms in UMTS environment. Our figures show, that Re-negotiation is considerably fast and can be made even faster by using native C implementation. The long duration of the SIP

TABLE I

QUALITY STEPS FOR THE REAL-TIME ADAPTIVE APPLICATION

Step #	Audio Codec	Video Codec	Frame Rate	Frame Size	Video Quality	Estimated BW
0	GSM	-	0 fps	-	-	20 Kbps
1	GSM	MJPEG	4 fps	SQCIF	50	80 Kbps
2	G.722	MJPEG	8 fps	SQCIF	40	140 Kbps
3	G.722	MJPEG	6 fps	QCIF	50	190 Kbps
4	G.722	MJPEG	4 fps	CIF	30	230 Kbps
5	G.722	MJPEG	6 fps	CIF	50	350 Kbps

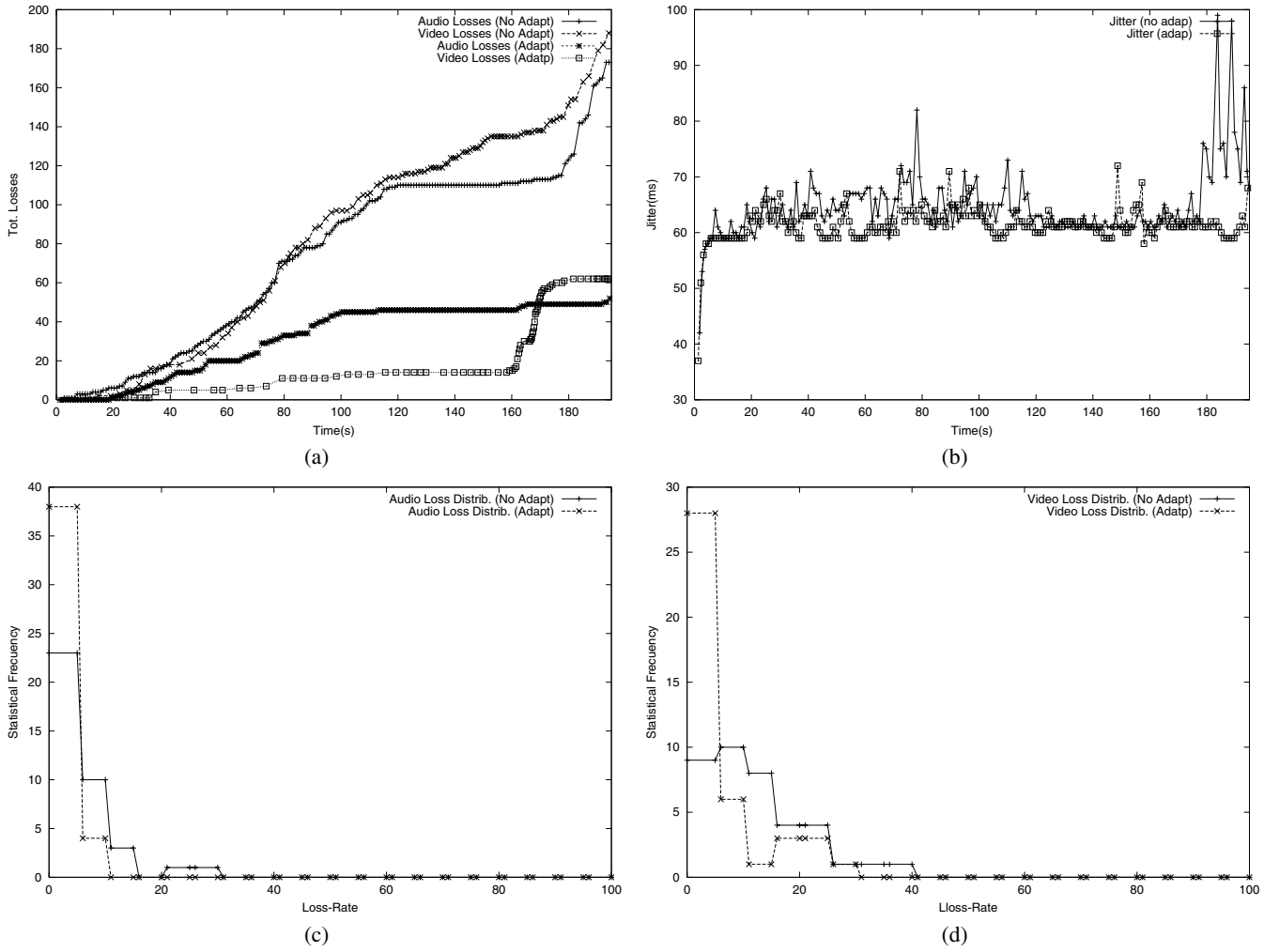

Fig. 7. Total Losses (a), Audio Jitter (b), Audio Loss-rate distribution (c), Video Loss-rate distribution (d)

Pre-negotiation time is due to stack initialization and due to the fact that E2ENP uses the `OPTIONS` method to perform Pre-negotiation compared to the SIP INVITE transaction used during Negotiation/Re-negotiation.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we presented an integrated solution to provide efficient real-time multiparty communications in ad hoc network extensions. We combine ad hoc routing with an adaptive application framework. The E2ENP allows to interact between local, peer and network resource management and provides versatile call-setup and QoS negotiation signaling for adaptive applications. We demonstrated through experiments that MMARP offers a good performance supporting IP multicast communications to standard-IP nodes in such scenario. In addition, we demonstrate that the use of our adaptive application framework offers a much better user-perceived QoS than traditional approaches. With E2ENP, peers are enabled to agree on common capabilities, QoS levels and even change them during a session.

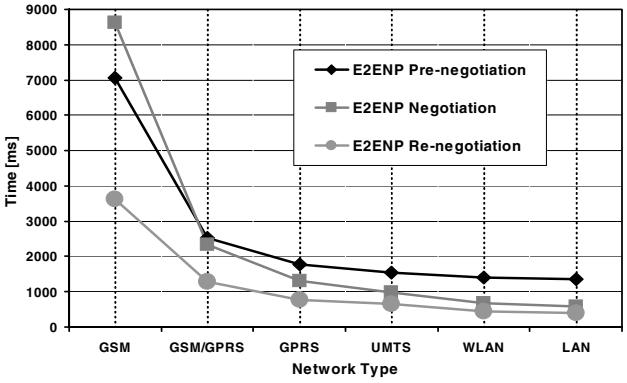

Fig. 8. Duration of an E2ENP phase

Our measurements demonstrate that adaptive applications and MMARP implementation has been able to offer a good user-perceived QoS in a scenario in which traditional solutions offer less performance. Furthermore, both for audio and video flows the adaptive applications approach outperforms the traditional real-time multimedia approach, demonstrating that adaptive applications are a good approach for dealing with complex network conditions by which ad hoc networks are characterized. The flexible E2ENP allows an interworking of adaptive applications, with optional local, peer and network resource management. This is important in such dynamic environments where QoS delivery on the transport/network layer significantly changes over time due to handover or temporal signal quality impairments.

As future work, we are working in making the adaptation logic to be completely automatic and intelligent using artificial intelligence techniques. Also, we are working towards intelligent SIP proxies that understand E2ENP and coordinate service domain and transport domain admission control.

ACKNOWLEDGEMENTS

This work has been partially funded by the Spanish Science and Technology Ministry, by means of the projects CROWN (PROFIT-FIT-070000-2003-662) and ISAIAS (TIC-2000-0198-P4-04). Partial funding has been provided by the Deutsche Forschungsgemeinschaft within the AKOM framework.

REFERENCES

[1] Alfano, M. and Sigle, R., *Controlling QoS in a Collaborative Multimedia Environment.* In: Proc. 5th IEEE Int. Symposium on High-Performance Distributed Computing (HPDC-5), August 1996.

[2] A. Alwan, R. Bagrodia, N. Bambos, M. Gerla, L. Kleinrock, J. Short, and J. Villasenor, *"Adaptive Mobile Multimedia Networks"*. In: IEEE Personal Communications, April 1996, pp. 34-51.

[3] Blake, S. et. al., *An Architecture for Differentiated Services.* IETF RFC 2475, December 1998.

[4] Bos, L. et. al., *SDPng extensions for Quality of service negotiation*, <draft-bos-mmusic-sdpng-qos-00.txt>, Work in progress, November 2001.

[5] Camarillo, G. et. al., *Integration of Resource Management and SIP*, <draft-ietf-sip-manyfolks-resource-07.txt>, Work in progress, April 2002.

[6] Estrin, D., Farinacci, D., Helmy, A., Thaler, D., Deering, S., Handley, M., Jacobson, V., Liu, C., Sharma P, Wei, L., *Protocol Independent Multicast Sparse Mode (PIM-SM): Protocol Specification.* RFC 2362, June 1998.

[7] Fenner, W., *Internet Group Management Protocol, Version 2.* RFC 2236, November, 1997.

[8] Garcia-Luna-Aceves, J.J., Madruga, E.L., *The Core Assisted Mesh Protocol.* In: IEEE JSAC, Vol 17, No. 8, August 1999, pp.1380–1394.

[9] Hartenstein, H., Schrader, A., Kassler, A., Krautgrtner, M. and Niedermeier, C., *High Quality Mobile Communication.* In: Proc. of KIVS 2001, Hamburg, Germany, April 2001.

[10] ITU-T Recomendation E.800 (0894), *Terms and definitions related to quality of service and network performance including dependability.*

[11] Jubin, J., Tornow, J.-D., *The DARPA Packet Radio Network Protocols.* In: Proc. of the IEEE, vol. 75 no. 1, January 1987, pp. 21–32.

[12] Kassler, A., Schorr, A., *Generic QoS aware Media Stream Transcoding and Adaptation.* In: Proc. Packet Video Workshop 2003, Nantes, France, April, 2003.

[13] Katz, R., *Adaptation and Mobility in Wireless Information Systems.* In: IEEE Personal Communication, Vol 1, No. 1, 1994, pp.6–17.

[14] Kazantzidis, M., Lee S.J., Gerla, M., *Permisible Throughput Network Feedback in AODV MANETSs.* In. Proc. ICC 2001, Helsinki, Finland, June 2001.

[15] Kutscher, D. et. al., *Session Description and Capability Negotiation*, <draft-ietf-mmusic-sdpng-06.txt>, Work in progress, March 2003.

[16] Lee, S.-J., Su, W., Gerla, M., *On-Demand Multicast Routing Protocol in Multihop Wireless Mobile Networks*. In: ACM/Kluwer Mobile Networks and Applications, 2000.

[17] Mankin, A. et. al., *Resource ReSerVation Protocol (RSVP).* IETF RFC 2208, September 1997.

[18] MIND Project: *Top-level architecture for providing seamless QoS, security, accounting and mobility to applications and services*, Deliverable D1.2, The MIND project IST-2000-28584, November 2002.

[19] M. Mirhakkak, N. Schult, and D. Thomsom, *"Dynamic Bandwidth Management and Adaptive Applications for a Variable Bandwidth Wireless Environment"*. In: IEEE JSAC, October 2001, pp. 1985-1997.

[20] Ramanathan, R. and Hain, R.,*"An Ad Hoc Wireless Testbed for Scalable, Adaptive QoS Support."* In: Proc. IEEE WCNC, November 2000, pp. 998-1002.

[21] Robles, T. , Kadelka, A., Velayos, H. , Lappetelainen, A., Kassler, A., Li, H., Mandato, D., Ojala, J., and Wegmann, B., *QoS support for an all-IP system beyond 3G.* In: IEEE Com. Mag., pp. 64-72, August 2001.

[22] Rosenberg, J., Schulzrinne, H., *An Offer/Answer Model with SDP.* IETF RFC 3264, June 2002.

[23] Rosenberg, J., Schulzrinne, H., *Reliability of Provisional Responses in the Session Initiation Protocol.* IETF RFC 3262, June 2002.

[24] Ruiz, P.-M., Brown, G., Groves, I., *Scalable Communications for Ad hoc Extensions connected to Mobile IP Networks.* In: Proc. PIMRC'2002. Lisbon, September, 2002. Vol. 3, pp. 1053–1057.

[25] Ruiz, P.-M., Gomez-Skarmeta, A., Groves, I., *Multicast Routing for MANET Extensions to IP Access Networks: The MMARP Protocol.* In: Proc. Int. Workshop on Mobile IP-based Network Developments. London, October, 2002. pp. 75–81.

[26] Ruiz, P.-M., Garcia, E.-J., *Improving User-perceived QoS in Mobile and Wireless IP Networks Using Real-Time Adaptive Multimedia Applications.* In: Proc. PIMRC'2002. Lisbon, Sept., 2002. Vol. 3, pp. 1467–1471.

[27] Sisalem, D., *End-to-end Quality of Service Control using Adaptive Applications.* In: Proc. IFIP International Workshop on QoS, 1997.

[28] Xiaohui Gu, K. Nahrstedt et. al.,*An XML-based Quality of Service Enabling Language for the Web.* Project Report - NSF, 2001.