BRENTA - Supporting Mobility and Quality of Service for Adaptable Multimedia Communication

Andreas Kassler[1], Louise Burness[2], Piyush Khengar[3]; Ernö Kovacs[4], Davide Mandato[5], Jukka Manner[6], Georg Neureiter[7], Tomàs Robles[8], Hector Velayos[9]

[1] University of Ulm, Germany, email: kassler@informatik.uni-ulm.de
[2] BT, England, email: louise.burness@bt.com
[3] King's College, London, England, email: piyush.khengar@kcl.ac.uk
[4] Sony International (Europe) GmbH, Germany, email:kovacs@sony.de
[5] Sony International (Europe) GmbH, Germany, email: mandato@sony.de
[6] University of Helsinki, Finnland, email: jmanner@cs.Helsinki.fi
[7] T-Nova Innovationsgesellschaft GmbH, Germany, email: Georg.Neureiter@telekom.de
[8] University of Madrid, Spain, email: robles@dit.upm.es
[9] Agora Systems, Spain, email: hector_velayos@agora-systems.com

Abstract: *Wireless broadband networking provides new challenges and benefits for applications. As the wireless link is always unreliable and interworking with other networking technologies is required, adaptable applications are a must to support Quality of Service (QoS) towards the user. There are several ways to provide this adaptability. An ideal endsystem architecture has to be flexible enough to support all kinds of adaptation mechanisms. Within this paper we focus on the requirements that are imposed on such an endsystem architecture. Based on these we build a flexible, layered architecture that provides a QoS and mobility enabled transport interface for higher components/protocols. The component model we introduce is inherently designed to support adaptable multimedia services with built-in QoS and mobility support. We believe that such an architecture is the first step towards an integrated QoS and mobility management system that can be used for next generation multimedia applications.*

1. Introduction

The main objective of the BRAIN (Broadband Radio Access for IP Based Networks) project is to propose a system architecture that combines broadband radio access technology based on the HiperLan 2 standard, with other wireless access network technologies (like UMTS), to enable full coverage of seamless IP-based services for users in hot spot areas and on the move. High quality communication is achieved by providing different connection types, and using a suitable frequency band with an available bandwidth of up to 20 Mbit/s. Mobility for Mobile Terminals (MT) is supported by building a scalable architecture based on the concept of BRAIN Access Routers (BAR) attached to a BRAIN IP based access network (BAN) which is connected to the core network using a BRAIN Mobility Gateway (BMG).
Fixed network services will also be made available for mobile users with comparable Quality of Service (QoS). BRAIN will support not only current IP based services and applications. In addition to enabling broadband wireless and mobile access to the Internet, the BRAIN network will also support QoS enabled applications and services. The combination of broadband wireless access and the support of mobility and QoS will enable new kind of applications that benefit from this synergy. As an example, with BRAIN technology it will be possible to achieve high quality video and audio wireless transmission using IP services, with support for QoS in all BRAIN elements. In general, BRAIN emphasises the combination of high bandwidth, low delay and guaranteed QoS - one best suited for demanding multimedia applications.
Based on this technological development, new business opportunities will become available and improved for manufacturers, network operators and content providers. People on the move will be able

to remain connected to their office, the Internet and thereby to the market. New forms of mobile multimedia communication will be possible by providing premier, real-time capable multimedia services to mobile users. Within this paper we focus on design issues above transport layer and of the BRAIN End Terminal Architecture (BRENTA), which has to provide support for mobility, high speed networking and QoS for services, applications and the user. In addition, support for adaptable services has to be provided to support heterogeneity issues inherent in wireless communication systems. The architecture encompasses resource management specific mechanisms that are needed inside the end terminal to provide more predictable QoS. BRENTA has been designed to be a general architecture that can be used on top of a HiperLan based solution, but it is also envisioned to be used in combination with future access technologies by abstracting from the underlying wireless access technology. This paper is composed as follows. Within the next section we motivate and explain why it is necessary to provide adaptable services and support QoS for wireless data transmission. In section three we introduce a service and application classification that is used in designing the BRENTA. Section four presents an introduction into the BRENTA. Section five discusses how resource management issues at the end system and QoS orchestration both locally and remotely is integrated with BRENTA. Finally, section six presents our conclusion.

2. QoS and Adaptable Services

Distributed multimedia applications impose stringent timing requirements. While it is possible to negotiate and maintain a common quality for data representation and transfer in a point-to-point scenario, point-to multipoint communications must be aware of a potential heterogeneity [PA93]. If mobile devices participate in such multipoint scenarios the level of heterogeneity increases even further due to different capabilities, compression support (hard/software) or network interfaces and bandwidth availability. When the participants request different quality levels and when the receivers have different processing capabilities and network connections, multipoint communications suffer indeed. Moreover, wireless or mobile systems should gradually adapt to QoS changes in the mobile channel [Ka94], [Sr97]. The scale of dynamic QoS variation is a crucial difference between mobile systems and communication in fixed high-speed network technology. Therefore mobile systems must provide some components to cope with the variability inherent to wireless access networks and provide adaptable services and QoS support. The major challenge is to guarantee adequate high-level end-to-end QoS in spite of the unreliable radio channel and the mobility of the terminals.
The BRAIN project will develop an end system architecture (BRENTA) that will be able to deal with the extreme QoS violations, which are likely to occur during a running session that is exposed to the radio access environment. BRENTA has to co-operate with BRAIN access network components in order to provide soft, end-to-end QoS, i.e. predictable and controllable services to applications and users. The core of BRENTA inherits and develops from the traditional Internet approach, but also incorporates aspects of a flexible QoS middleware solution.

3. Application and Service Classification

Basically, different types of distributed applications have to be supported by BRENTA. *Legacy applications (or type A applications)* use standard TCP/IP (+UDP) stacks and do not directly address QoS issues or mobility specific tasks. As an example, consider Web-Browsing or FTP-applications.
Recently, certain IP based applications have evolved to use newly developed IP QoS mechanisms (based on IntServ [IS]/RSVP [RSVP] or DiffServ [DS]), to provide more predictable services. These *Self-contained QoS-aware* applications (or type B applications) will also be able to operate on a BRAIN MT with a performance similar to fixed network connections. In addition, they may be able to manage mobility related functionality without any external support. These applications can use various session layer protocols (e.g. SIP, H.323), use standard IntServ and/or DiffServ mechanisms and typically include RTP/RTCP/RTSP functionality. However, we consider these applications to be specialized, written by skilled application programmers, who know how to directly deal with issues like e.g. QoS violations due to hand-over mechanisms. As an example consider here enhanced VoIP clients. However, these applications (unless otherwise designed for a specific platform), will not be

able to interact with local resources (e.g. CPU or memory) in a co-ordinated manner, and will have to monitor the QoS delivered by the system and provide mechanisms to react to QoS violations.
In addition to nowadays distributed applications we see a need for providing QoS, mobility and adaptation mechanisms for emerging component based applications. We denote them as *QoS-aware applications based on a component model (or Type C applications).* These will be able to adapt to QoS violations and perform mobility specific tasks by themselves, but rely on basic lower level functionality offered by pre-fabricated components like video grabbers, data compressors, packetizers, renderer, etc.. This will lead to easier development of distributed QoS aware applications, thereby considerably reducing time to market and fostering third-party basic components (like Java Beans) market.
Finally, type D applications may only subscribe to adaptable services that provide support for QoS and mobility by themselves. Type D applications are typically not designed to directly deal with *QoS violation* issues and therefore need some form of *intelligence,* provided by external components that hide QoS and the handling of mobility. Furthermore, application programmers can even use high level QoS-languages for provisioning QoS. Not yet implemented, but this paradigm will significantly ease development of distributed multimedia applications. Application programmers will just have to subscribe to adaptable media streaming services, that manage QoS and mobility transparently to the application.

4. BRAIN End Terminal Layered Architecture

Applications will interface with a QoS- and mobility-aware protocol stack through set of interfaces, each addressing one of the application types described below. Legacy applications (type A) access IP services by directly interacting with the classical (neither QoS- nor mobility-aware) transport layer (API 0). Legacy applications (type A) may eventually use the services provided by a new, QoS- and Mobility-Enabled Transport Interface. This interface (API A), also denoted as *Service Interface*, allows separating BRENTA from a specific network implementation. For instance, one might specify monitor thresholds (e.g. low/high watermarks) and entry-points where events are triggered whenever QoS violations occur. Since most legacy applications do not feature any QoS support, an external configuration tool for API A should allow users to configure/setup and monitor the QoS parameters provided. As an example, such a tool might instruct the network stack to associate a certain DiffServ Codepoint (DSCP) with a dedicated flow (i.e. associate high priority and low packet dropping for all IP packets for port 80). The lower layers would recognize the request for premium service for the dedicated flow and associate a certain HiperLan channel for these packets. By such mechanisms, standard legacy applications (like Web-Browser) could be enhanced to provide better predictable service without changing the application code. Such a tool might be accessed through a set of GUIs (each addressing a given level of user expertise). If no QoS is provisioned, the given legacy application will still be able to operate as usual, but without QoS support.
Type B applications may use various session layer protocols (e.g. SIP, H.323) across API B. These protocols may be even partly embedded into the applications. Since type B applications directly have to manage QoS and mobility related issues by themselves, they only use standard IETF protocols enhanced by some mobility related functionality.
Type C applications incorporate the functionality offered by a component level API (denoted as API C). It has to provide specific multimedia components like frame grabbers, codecs, packetizers, renderers that can be chained together on a per flow basis, based on applications requirements. IETF protocols are encapsulated in components to provide compatibility and flexibility. That is, type C applications would use a session manager that could be based on SIP or H.323 transparently to the applications. In addition, API C provides components that deal with local resource management issues like soft real-time scheduling of media processing threads or local memory management in order to provide better control over the local system and more reliable QoS. All components provide functionality for monitoring and adaptation. However, the intelligence that reacts to QoS violations still has to be implemented inside the application.

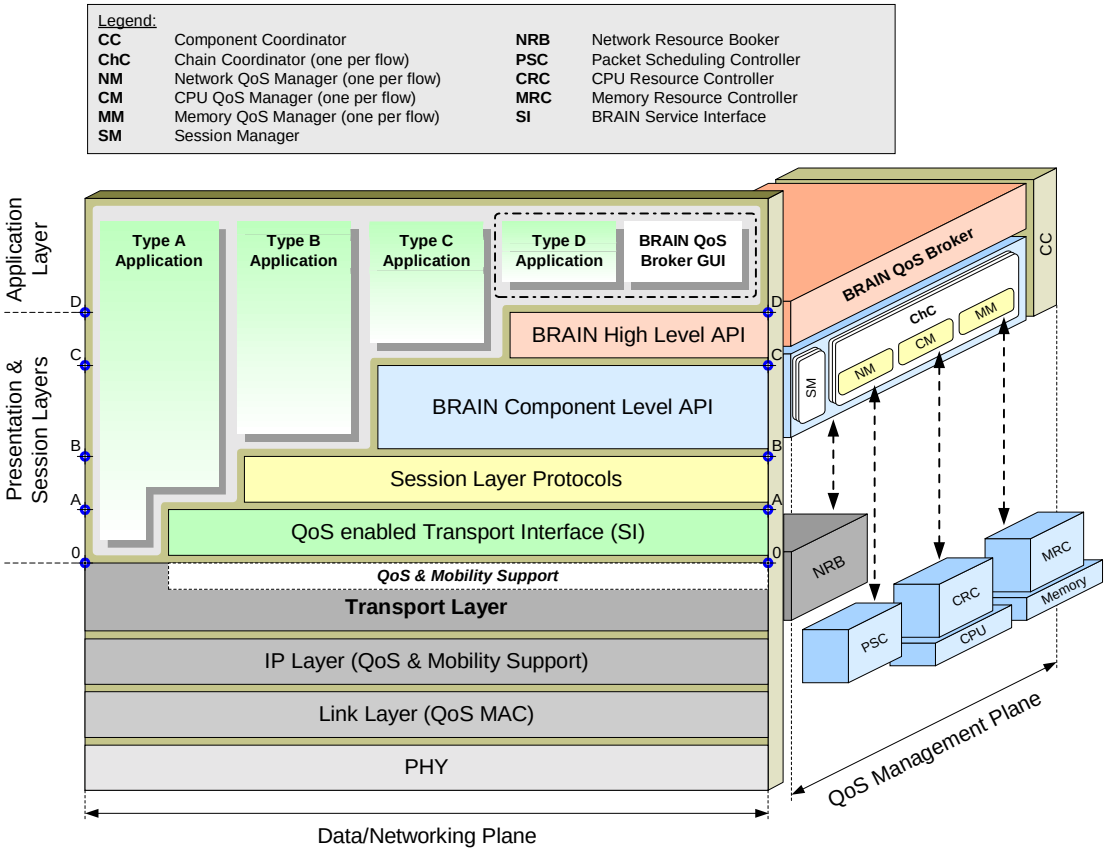

Figure 1: The BRAIN End-System layered Terminal Architecture (BRENTA)

Finally, type D applications will use a QoS Broker (either as a component provided by API C or transparently via API D) as intelligence, which manages and orchestrates local and remote resources. The broker re-acts to QoS violations and mobility specific events on behalf of the application based on a QoS trading policy that may even be downloaded from a QoS trading provider. Since type D applications just have to subscribe to controllable, adaptable multimedia services, components manage media processing transparently on behalf of applications. The application only has to provide the policy that describes how the system should react in the case of local and/or remote resource variation or QoS violations. However, this policy may also be provided by a GUI through API D so that a user may specify its preferences and local QoS policies independently from applications. The GUI might provide different level of user expertise. Persons with no expertise can thus be able to select from a high level view the QoS profile they desire. The QoS Broker will manage the component chains and feature QoS- and mobility-awareness, e.g. by taking high-level decisions, as for what adaptation to QoS violations concerns.

The APIs C and D provide typical middleware solutions. These concepts naturally lead to a layered architecture depicted in Figure 1. In order to support all different application types described above, the proposed architecture needs to be (i) *modular,* (ii) *open and (iii) configurable/flexible*. *Modularity* guarantees that existing applications can be immediately used *as is*, whereas more complex middleware solutions can be gracefully introduced later, as soon as they become available. *Openness* further broadens the scope of the proposed architecture, taking into account interoperability issues with other architectural solutions (e.g. active networks). *Flexibility* is needed to dynamically cope with different media types by, e.g. supporting downloadable codecs or policies. In addition, the interfaces have to be well defined and standardized so that QoS enabling components may enhance the system by being downloaded from a QoS component provider even during runtime.

The BRAIN end-system architecture has to provide seamless services over QoS enhanced IP networks with mobility support and wireless access sub-networks. Premium services will be available in hot spot areas with maximum quality, depending on user preferences. When moving out of those areas, the user expects a controlled and predictable degradation of the QoS, if he is operating a dual-mode

terminal. However the user may control the degradation by specifying a set of high-level QoS parameters and policies for each service she/he subscribed. These parameters are the main input for tuning the BRAIN QoS architecture. The architecture specifies how these parameters are mapped to application and network specific QoS parameters, providing end-to-end QoS. To achieve user expectations, co-operation is needed between the application providing the services, the operating system managing local resources and the network elements including the BRAIN mobile terminal.
For supporting all type of applications described above (i.e. type A, B, C and D), different sets of application programmer interfaces (APIs) have to be provided for different application types. BRENTA encompasses a variety of components, end-system mechanisms and protocols to cope with the dynamic variation in mobility management and QoS. It will provide applications with more predictable services and allow applications to react in a pre-determined way to QoS violations. One reason for building upon a component based architecture was to even allow dynamic downloading of key components for e.g. updating codecs or protocol objects. BRENTA is designed by using IETF protocols and custom components as building blocks. Due to the open architecture and the component model used, future emerging IETF protocols providing support for mobility and QoS may easily be integrated.

5. Resource Management and QoS Orchestration

In order to provide applications with QoS, mobility and adaptation support (according to the various types of applications described above) a set of entities has been identified. We split our architecture in several planes: a QoS and mobility enhanced protocol plane, a resource management plane that manages local resources and a QoS management plane, that co-ordinates all components and manages end-to-end QoS especially for distributed multimedia applications.
Besides QoS enhanced transport interfaces/protocols, mechanisms for managing resources are necessary in order to provide more stringent QoS guarantees. As an example consider a video player that receives frames from a video on demand server over the air interface, decodes and displays them. It is not sufficient to just reserve resources on the air interface and in the network (e.g. 1Mbit/s constant rate via mobility enhanced RSVP interworking with HiperLan2) in order to provide predictable QoS. If the MT is not able to manage its local CPU and memory resources in order to decode and display 25 fps it is also not necessary to reserve expensive resources on the air interface. In other words, all entities and components involved in the overall transmission process have to work together to provide predictable end-to-end QoS in a reliable manner.
The basic building blocks for providing QoS are based on a QoS enabled transport interface and session protocols. We identify the need for pre-fabricated, customizable software entities (components) that provide meaningful services through a published service interface (e.g. Java-Beans, DCOM-objects, or CORBA-objects that may even be downloaded from a QoS component provider). Each component has to monitor its most important QoS parameters, and implement means for its adaptation. A Component Coordinator (CC) manages component lifecycle and "chains" of multimedia components on behalf of the application (Type C application) or of an external QoS Broker (Type D application). Each "chain" is associated with a flow and is controlled by the CC through a Chain Controller (ChC). The ChC modifies "chains" based on applications/QoS Broker (see below) commands, in response to variations in QoS and mobility conditions.
The resource management plane is split into resource controllers (RC) and resource managers (RM). Tasks of the resource managers are to provide usage information for a given resource, implement mechanisms for adaptation for a specific local resource and hide the complexity of QoS negotiation from the specific local resource. Each RM instance is typically associated with one flow and a specific type of resources (namely, CPU, Memory, and Network). The RM can be a component, manageable through the CC. Resource Controllers control admission for the specific resource, manage reservation of the resource, allow dynamic negotiation for the resource and perform adaptation. Typically, one instance per end terminal and resource is available (i.e. one CPU Controller). RM request, negotiate, and release resources via the RC. For shared resources (like CPU, memory or network) resource schedulers assert that resources are granted to specific resource consumers according to their requirements that were specified using the RM.

In the QoS management plane a QoS broker serves as the centralized intelligent entity governing at the highest level all the QoS and mobility mechanisms on behalf of applications. This entity ensures that sufficient resources are available to accommodate a given application requirement at connection establishment time, both locally and remotely. This implies a negotiation process among peer QoS Broker entities. Fallback mechanisms (broker-enabled applications communicating with other types of applications - like type A, B, or C) shall be taken into account as well, which could be achieved via proxy agents. The QoS Broker maps QoS parameters across the various components/protocol layers. Afterwards, the QoS Broker monitors the components (as an example the broker could retrieve connection quality by monitoring RTP/RTCP components), and reacts to any mutated condition (e.g. QoS violations) by fine-tuning or re-arranging multimedia component "chains" which might require re-negotiation with peer broker entities. The QoS Broker can be a component by itself and can influence the chain controller. The broker uses the resource managers to request, reserve and release local resources. In addition, the broker controls a media flow via its associated chain controller. Similarly, usage information from multimedia components and other statistical data like compression rate or time to compress a frame can be retrieved from the components via the chain controller. Also, RTCP sender and receiver reports can be made available to the broker via the chain controller. The monitoring information and the user supplied local QoS profile and preferences are used in the QoS trading process to manage local and remote resources. QoS profiles may be specified by each user for different situations (e.g. at office / at home) separately and dynamically updated with BRENTA.

6. Conclusion

In this paper we have presented an overview of the design of BRENTA, an end-system terminal architecture for the BRAIN system that provides adaptable services, support for QoS, and mobility. We think that the modular, flexible and open architecture provides many benefits, e.g. the transparent enhancement of legacy applications by QoS mechanisms, and also the ability to take advantage of new IETF protocols as they develop. In addition, BRENTA clearly separates local resource management, QoS management and network QoS mechanisms. BRENTA gives a clear separation for the actual network QoS implementation so that it can be used with over provisioned VPNs, as well as with DiffServ or IntServ enabled networks. As a new feature, several APIs will be provided that allow the rapid development of distributed multimedia applications based on adaptable services, incorporating QoS and mobility handling.

References

[DS] The Differentiated Services working group home page, ttp://www.ietf.org/html.charters/diffserv-charter.html

[IS] The Integrated services working group home page, http://www.ietf.org/html.charters/intserv-charter.html

[KA94] Katz, R., Adaptation and mobility in Wireless Information Systems, *IEEE Pers. Comm.*, I/1994, no. 1, pp. 6-17.

[PA93] Pasquale J., et al, Filter Propagation in Dissemination Trees: Trading Off Bandwidth and Processing in Continuous Media Networks, *Proceedings of the 4th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV 93)*, Lancaster University, Lancaster, UK, 1993, pp. 269,278.

[RSVP] The resource reservation protocol working group home page; http://www.ietf.org/html.charters/rsvp-charter.html

[SR97] Shrivastava M., Mishra P.P., On QoS in Mobile Wireless Networks, *Proc. IEEE 7th Int. Workshop on Network and Operating System Support for Digital Audio and Video*, pp. 147-158, 1997.

Acknowledgement

This work has been performed in the framework of the IST project IST-1999-10050 BRAIN, which is partly funded by the European Union. The authors would like to acknowledge the contributions of their colleagues.