Characterization and Selection of Applications for ENNCE WP1

Norwegian Computing Center
APPLIED RESEARCH AND DEVELOPMENT
Norwegian Computing Center / Applied Research and Development

NOTAT/NOTE

IMEDIA/03/99

Wolfgang Leister
Peter Holmes

Oslo
January 1999

Norwegian Computing Center
APPLIED RESEARCH AND DEVELOPMENT

NR-notat/NR Note

Tittel/Title:
Characterization and Selection of Applications for ENNCE WP1

Dato/Date: January
År/Year: 1999
Notat nr: IMEDIA/03/99
Note no:

Forfatter/Authors:
Wolfgang Leister, Peter Holmes

Sammendrag/Abstract:

The purpose of this document is to help support and justify the selection of applications suited for further investigation in regard to the QoS Negotiation Model described developed for the ENNCE project. We propose some criteria for selection of suitable applications. Several candidate applications and frameworks that are available for experimentation are classified. For the purpose of experimentation with QoS negotiation based upon the QoS Negotiation Model, a framework called MInT appears to be the most promising candidate.

Emneord/Keywords: Quality of Service, QoS negotiation, adaptive applications, architectural frameworks

Tilgjengelighet/Availability: Open

Prosjektnr./Project no.: ENNCE 801001

Satsningsfelt/Research field: quality of service, adaptive applications

Antall sider/No of pages: 15

Norsk Regnesentral / Norwegian Computing Center
Gaustadalléen 23, Postboks 114 Blindern, 0314 Oslo, Norway
Telefon 22 85 25 00, telefax 22 69 76 60, telex 76518 ncc n

Characterization and Selection of Applications for ENNCE WP1

Wolfgang Leister
Peter Holmes

Norsk Regnesentral
January 1999

Table of Contents

1. Purpose

The purpose of this document is to help support and justify the selection of applications suited for further investigation in regard to the QoS Negotiation Model described in [1]. Though it is beyond the scope of this document to elaborate upon the negotiation model or general QoS issues, some background information about these is provided in the next section. The application chosen in the WP1 is supposed to be used for experimentation on the project's network infrastructure [14].

2. Background

2.1 Quality of Service (QoS)

From the end user's perspective, the main issue of QoS is to provide reasonably well-defined application behavior and performance. From the application's perspective, the main issue of QoS (Quality of Service) is that the underlying platform, middleware and network provide a sufficient level of system resources which enable the application to deliver the behavior and performance desired by the end user. QoS represents the set of those quantitative and qualitative characteristics of a distributed multimedia system which are necessary to specify, monitor and control in order to achieve the required functionality [2]. In short, the system-specific factors which influence the "objective" aspects of QoS arise from the performance and characteristics of:

- the application(s) in use and the media formats they operate with,
- the various elements of the platform (e.g., terminal, devices, codecs, screen, middleware, operating system), and
- the underlying network infrastructure and basic network services.

There are also qualitative and subjective characteristics which can be used to assess the user's "degree-of-satisfaction" in regard to the collective effect of service performance. At the user-level, such characteristics include:

- sound quality (e.g., clarity, echo, noise suppression, clipping, fragmentation, etc.);
- image quality (e.g., image sharpness, image size, image update rate, partial vs. full frames, etc.);
- percentage of text / image which can be viewed without scrolling;
- one-way delay (e.g., delays during downloading);
- two-way delay (e.g., gaps during dialog); and,
- synchronization (e.g., lip-synchronization, (tele-)pointer synchronization).

Further, other factors can influence the user's subjective experience of QoS. These include:

- the user's abilities and eventual disabilities,
- the user's surrounding environment,
- the user's capacity and willingness to pay for "improved" services, and
- the user's previous experience with "improve-able" services.

It is a general observation that user-related factors are less well addressed in most recent research; instead, the focus is towards networking and other technical aspects [4]. The QoS Negotiation Model presented in ENNCE WP1 takes both the user's needs and the technical aspects into account [1].

2.2 Adaptation

Gecsei [5] describes adaptation in distributed multimedia systems (DMS). There, he emphasizes the goal of achieving user-acceptable performance of applications in the face of unexpected QoS variations:

> *In the context of DMS, adaptation is a complex process involving a number of system components. Despite the multitude of approaches, the overall objective of adaptation is the same: to extend the range of conditions over which a program performs acceptably.*
>
> (Gecsei [5], p. 59)

Furthermore, he describes the adaptation process in the following generalized manner:

> *Dynamically, adaptation in a DMS works through a set of mechanisms whose goal is to maintain the operating point within an acceptance region. When the operating point moves outside this region, a controller initiates some corrective action called adaptation, which brings the operating point back within the acceptance region. The controller does this by monitoring the value(s) of the observed parameters and by executing an adaptation algorithm.*
>
> (Gecsei [5], p. 60)

Since the focus in this document is upon applications, we can choose to create a working definition for the term ”adaptive application”. Here, an adaptive application is an application which:

- has an understanding of the end-user's current acceptance region (e.g., based upon the end-user's preferences and/or specification of this region);
- has an understanding of its current operating point; and
- adapts itself — according to a well-specified adaptation algorithm — whenever its current operating point moves outside the end-user's current acceptance region.

2.3 The QoS Negotiation Model

ENNCE WP1 is developing a generic model for QoS negotiations, the QoS Negotiation Model. In this model, negotiations are addressed and managed by a Service Agent (SA), an entity distinct from — yet acting on behalf of — the application.

For further development and investigation of the QoS Negotiation Model, one or more suitable applications must be integrated with the model; it is intended that this be achieved by supplying the applications with the appropriate interfaces to the SA. In this way, concrete experience with the model shall be gained, thereby providing a basis from which to judge the model's adequateness and suitability with regard to QoS negotiations.

These experiments will also provide some answers and input concerning certain unsolved questions and issues. These include an assessment as to how generic the SA can be, as well as how the interfaces between applications, user interface, and SA should be organized and most effectively designed. The user's role within the negotiation process remains still to be studied, especially when the services are tied to a cost model, and real money has to be paid for the services.

3. ENNCE's Application Requirements

This section begins with a general categorization of application types, as put forward by IETF. Thereafter, a list of applications domains suited for ENNCE WP1 is presented. Finally, a high-level description of ENNCE's WP1 requirements for application selection is presented.

3.1 Classes of Applications

The framework of the IETF (RFC 1633) [3] suggests three classes of applications: elastic, rigid, and adaptive applications. Elastic applications are often used for bulk file transfer and similar applications — applications which do not have to meet hard real time restrictions. This application class will enjoy accelerated performance when using higher bandwidth, but many other QoS parameters which could be negotiated are rather irrelevant in this case (e.g. jitter). Therefore elastic applications are not well suited as a case for ENNCE's WP1 effort. Still, we have to keep in mind that some protocols for data piping make use of an elastic protocol with no real time control. (In this case we can possibly consider the application to be in the class of rigid applications).

Both rigid and adaptive applications are suited for a investigation by ENNCE WP1. The difference is in how the responsibilities for controlling the data streams are distributed. Rigid applications leave all control to the network / middleware, with the consequence that the application will fail in case the preconditions cannot be met for some reason. For the purpose of ENNCE, rigid applications constitute the simpler case, since most of the restrictions are constant. These applications may still require end-to-end negotiation, however, in order to check and set the boundaries for the application's operating point. In addition, commands to the middleware may be involved.

The class of adaptive applications has been briefly defined above (see section 2.2). These kinds of applications are suitable candidates for further investigation in this project since — according to our definition — they may implicitly contain the information and mechanisms required in order to compare their own current operating point to the end-user's current acceptance region. An excellent candidate for ENNCE WP1 would be an adaptive application which provides access to the control of this comparison operation; the capacity to substitute adaptation algorithms, or at least change the policies within the adaptation algorithm, would also be of great value.

3.2 Application Domains

Applications suited for ENNCE WP1 can be in several applications domains, which may include:

- Streaming Video / VoD
- IP Telephony
- Digital TV
- Conferencing
- Tele-Teaching
- (Distributed) Games / VR Worlds
- Chat in multimedia mode
- GIS and navigation applications

Some frameworks exist, that combine some of these domains, e.g. the MInT project [6] where several of the MBONE tools [11] are combined to an application that is built for teleteaching applications.

Some of these domains are rather specialized, as Digital TV, Video on Demand, and IP Telephony. For the purposes of the ENNCE project we prefer more general applications.

These domains have very different requirements to the network and operating system QoS. Therefore we cannot include all aspects into the choice of one single applications, and therefore the choice of a framework is advantageous.

3.3 Requirements

In this section, we present high-level description of ENNCE's WP1 requirements for application selection.

The application must have at least a degree of communication which makes it necessary to introduce QoS methods. These might be real time, high bandwidth, or other QoS demands. In the best case, candidate applications might be adaptive ones, including well-defined interfaces for controlling specific QoS parameters. Otherwise, candidate applications should be configurable in ways relevant to experimentation with QoS negotiation.

In the client/server paradigm, configurability may be possible on either or both of the client side and the server side. On the server side for example, encoding format, encoding parameters, frame-rate and different other parameters might be possible to control. Examples on the client side include possibilities where the application can be configured to drop certain frame types, decode mp3-audio only to a certain frequency, use other algorithms, etc. Preferably, candidate applications will include well-defined APIs for such configuration.

An application which offers only limited possibilities/APIs for configuration may still be of interest, assuming that it provides very special possibilities for experimentation with QoS. In such cases, access to the application's source code is a must.

Briefly summarized, the requirements for application selection are:

- ***R1*** the application enables the implementation of *code-passing interfaces* as a basis for experimentation with QoS negotiation within the QoS Negotiation Model (mandatory);
- ***R2*** during operation, the application must be able to sufficiently exercise the network (i.e., continuous flow or bursts of ≈ 0.3 Mb/sec or more) (mandatory)
- ***R3*** source code for the application is available and/or the application offers well-defined interfaces for controlling specific QoS parameters (mandatory)
- ***R4*** the application offers well-defined APIs for configuration (highly preferable)
- ***R5*** the application provides *unique* possibilities for experimentation with QoS (possible)

4. Application Characterization

Among the possible applications we investigate the following for their suitability: The Berkeley Continuous Media Toolkit (CMT) [7], Vic [8] and Vat [9]. Additionally we take a look at LAVA [16] and a student's project in ENNCE WP2 [13]. Other candidate

applications can be found in the MBONE Software Archive [11], which is a collection of applications used to evaluate the Multicast Backbone infrastructure. These applications are widely used, and the choice of one or several applications from this collection would make it possible to compare own results with others' efforts.

We investigate also Mediate [12] for suitability. Mediate is an application framework from which a number of applications can be created. The framework includes a few basic applications of its own, which can be naturally extended.

The MInT framework [6] employs an overall system topology and application constellation which is similar to that outlined in the ENNCE WP1 work. MInT employs a framework approach into which several applications are integrated, rather than using an approach based upon a single, monolithic application. For this reason, most of the characteristics outlined in the MInT project documentation are also valid for ENNCE WP1. The fact that MInT has these characteristics strongly influences its candidacy for use within ENNCE WP1's further work.

4.1 Organization of the characterization

In the following we characterize some of the chosen applications to give a hint on suitability for the project. The criteria used are given in the following table.

	Description
Application Domain	Describe the application domain as stated in section 3.2. Additional characteristics, e.g. one-way/two-way communication should be mentioned additionally.
QoS parameters	Consider QoS- and implementation-specific aspects which may affect candidacy. Mention any other relevant specifics (see also R3).
Bandwidth	State whether application meets requirement R2 (i.e., sufficient traffic volume).
Codecs	Describe media types that are used within the application
Protocols	Which application/transport protocols are used (e.g. MPEG, RTP, TCP, UDP)
API issues	
User Interface	Does the application already have a user interface for certain QoS aspects ?
Negotiation API	Is there an API available for negotiation (see also R1, R4) ?
Configurability (server)	Describe configuration possibilities at startup / while running
Configurability (client)	Describe configuration possibilities at startup / while running
Adaptivity	Does the application act adaptively ?
Implementation issues	Describe issues on the implementation, that could have an influence on practical testing; see also R1.
Code access	How is the code available. This may have impact in being able to include a new API for negotiation purposes.

Code policy	What policy is used? Can code be accessed, changed, etc.? See also R3.
Implementation language	The implementation language(s) has an impact on practical testing, availability of tools, compilers, etc.
Operating system(s)	Restrictions in choice of the operating system may have an impact on using the application within the project.

Each of the applications will be characterized with regard to the table presented above.

4.2 CMT

The Berkeley Continuous Media Toolkit (CMT) [7] is a toolkit to support continuous media players for playing MPEG-I video, MJPEG video, 8 KHz ulaw audio, 8 bit linear audio, and 16 bit linear audio. The CMPlayer can be used to play files on local file-systems as well as files on a remote CM video file server. The CMPlayer is written in TCL and is provided with CMT as a set of tcl scripts.

CMT	**Description**
Application Domain	Streaming video, one way media stream
QoS parameters	
Bandwidth	Does suffice R2.
Codecs	MPEG-1 video, au audio
Protocols	MPEG, RTP
API issues	
User Interface	Yes
Negotiation API	At startup
Configurability (server)	Configurable at startup
Configurability (client)	Configurable at startup
Adaptivity	No
Implementation issues	
Code access	Free
Code policy	BSD
Implementation language	C, Tcl/Tk
Operating system(s)	Digital Unix, FreeBSD, HP-UX, Solaris, (SGI, Linux)

4.3 VIC/VAT

Vic [8] is a tool for video conferencing over the Internet, that was designed with an extensible architecture to support heterogeneous environments and configurations. For example, in high bandwidth settings, multi-megabit full-motion JPEG streams can be sourced using hardware assisted compression, while over lower bandwidth environments like the Internet, aggressive low bit-rate coding can be carried out in software. Vic is based on version 2 of the Real-time Transport Protocol (RTP), which provides basic real-time media communication support. Although vic can be run point-to-point using standard unicast IP addresses, it is primarily intended as a multiparty conferencing application.

VIC	**Description**
Application Domain	Video Conferencing
QoS parameters	
Bandwidth	Does suffice R2.
Codecs	H.261, MJPEG, NV, CellB, SCR, MPEG
Protocols	RTP
API issues	
User Interface	Yes
Negotiation API	No
Configurability (server)	N/A (No server.)
Configurability (client)	Interactive, rate control.
Adaptivity	Yes
Implementation issues	
Code access	Free
Code policy	Open
Implementation language	C
Operating system(s)	Digital Unix, FreeBSD, HP-UX, Solaris, SGI, Linux, Windows

Vat [9] the Audio Tool, allows users to conduct host-to-host or multihost audio teleconferences over an internet (multihost conferences require that the kernel support IP multicast).

VAT	**Description**
Application Domain	Audio Conferencing
QoS parameters	
Bandwidth	May suffice R2, if high-quality audio coding is used.
Codecs	PCM, GSM, DVI
Protocols	RTP
API issues	
User Interface	Yes
Negotiation API	No
Configurability (server)	N/A (No server.)
Configurability (client)	Interactive
Adaptivity	No
Implementation issues	
Code access	Free
Code policy	Open
Implementation language	C, Tcl/Tk
Operating system(s)	Digital Unix, FreeBSD, HP-UX, Solaris, SGI, Linux, Windows

4.4 ENNCE WP2

The student's project in ENNCE WP2 [13] uses an application for transporting video over an ATM network. The project is implemented in DaCaPo, XIL, and employs several video codecs. While it would be interesting to experiment whether the negotiation model of ENNCE WP1 would fit for this implementation, it is not yet available (it has yet to be finished).

ENNCE WP2	**Description**
Application Domain	Video Streaming
QoS parameters	
Bandwidth	Does suffice R2.
Codecs	CellB, JPEG, MPEG-1, H.261
Protocols	MPEG, H.261, UDP
API issues	
User Interface	Unknown
Negotiation API	Unknown
Configurability (server)	Unknown
Configurability (client)	Unknown
Adaptivity	Unknown
Implementation issues	
Code access	Free
Code policy	Unknown
Implementation language	Unknown
Operating system(s)	Solaris

4.5 LAVA

The LAVA project [16] developed a streaming video system for delivery of video streams based on MPEG. Though the code is available and does fulfill most of the requirements, the maintenance of the software is discontinued at present.

LAVA	**Description**
Application Domain	Video Streaming
QoS parameters	
Bandwidth	Does suffice R2.
Codecs	MPEG-1, MPEG-2
Protocols	MPEG, M-JPEG-Transport, UDP
API issues	
User Interface	Yes
Negotiation API	Partially
Configurability (server)	Partially
Configurability (client)	Partially
Adaptivity	Partially
Implementation issues	
Code access	Free
Code policy	Open
Implementation language	C
Operating system(s)	Solaris (Server, Client), SGI (Server, Client), Windows (Client)

4.6 Mediate

Mediate [12] provides a robust, common development framework for collaborative multimedia applications. MEDIATE was originally designed for supporting office-based co-operative work, but it is now being tested for new domains such as mobile work and games. It addresses the problem of weakly integrated multimedia support in CSCW applications leading to inflexible and redundant interaction management. MEDIATE offers a toolkit architecture that comprises a distributed multimedia switching system, a protocol for distributed processing, a grammar for persistence and communication, and a factory for remote object generation. MEDIATE is also an *object-oriented development framework* with support for several integrated multimedia types, most notably digital video, as well as the necessary functionality to build domain-specific applications.

MEDIATE	**Description**
Application Domain	Multimedia Framework.
QoS parameters	Uses applications with very different QoS requirements.
Bandwidth	Does suffice R2.
Codecs	Several
Protocols	Several
API issues	
User Interface	Java-based
Negotiation API	No
Configurability (server)	Partially
Configurability (client)	Partially
Adaptivity	No
Implementation issues	
Code access	Available
code policy	Open
Implementation language	Java
operating system(s)	Solaris, FreeBSD, Windows NT

4.7 MInT

MInT [6] is a flexible multimedia tool set that allows the establishment and control of multimedia sessions across the Internet. The system architecture is fully distributed, with no central components. MInT offers the following features:

- Audio transmission and reception for qualities ranging from GSM up to CD based on the NEVOT and VAT tools
- Video transmission and reception based on VIC
- An integrated conference control instance
- floor control
- voting based on MPOLL
- interface to SDR
- RSVP reservation capabilities for all used applications
- Adaptive video transmission
- Joint viewing and remote control of postscript documents
- Invitation of remote users based on the Session initiation protocol (SIP-version 3.0) with name resolution capabilities

MInT	**Description**
Application Domain	Multimedia Framework, Teleteaching / Conferencing
QoS parameters	Uses applications with very different QoS requirements.
Bandwidth	Does suffice R2.
Codecs	Several (application dependent)
Protocols	Several (application dependent)
API issues	
User Interface	Yes
Negotiation API	Yes
Configurability (server)	Yes, in some cases (application dependent)
Configurability (client)	Yes, in some cases (application dependent)
Adaptivity	Yes, in some cases (application dependent)
Implementation issues	
Code access	Free
Code policy	Open
Implementation language	Application dependent
Operating system(s)	Digital Unix, FreeBSD, HP-UX, Solaris, SGI, Linux, Windows

5. Conclusions Regarding Application Selection

5.1 Project status

At the outset of the project, ENNCE had the goal of working towards the selection, experimentation and development of applications which — in the final phases of the project — could be easily ported between the WP1 and WP2 system models and architectures. After the course of the project's first year, it can be observed that the overall topologies of WP1's QoS Negotiation Model and WP2's system model liken one another to a certain extent.

It has been judged, however, that it would be wasteful to try and redirect the momentum of WP2's software implementation progress to accord precisely with WP1's model; it is not reasonable to sacrifice precious time and good work merely to attain complete consistency within the project. Likewise, it is also judged to be more valuable to allow WP1 the possibility of exploring the advantages and disadvantages of code-passing interfaces for QoS negotiation in a more free development context, rather than forcing WP1 to try and implement the features of their Negotiation Model within WP2's system architecture.

Quite simply, there are too few resources within the project to try and attain the fine-grained coordination necessary in order to guarantee application-portability by the conclusion of the project's time-frame; the project's resources can better be used by allowing WP1 and WP2 to address — in a loosely-coupled manner — the specific QoS issues targeted by each of their respective work-packages and architectures.

5.2 Selection of MInT

In light of this assessment of project status, WP1 has selected the MInT framework in order to implement and explore the viability and efficiency of code-passing interfaces for QoS negotiation within the QoS Negotiation Model.

MInT offers a near-identical architectural topology when compared with WP1's current model. Presently, MInT employs certain IETF-standards (e.g., SIP, RSVP) as the basis of own message-passing interfaces. Since the source code for MInT is available (requirement R3), it will be possible to redesign and re-implement MInT's internal interfaces to be code-passing interfaces, instead (requirement R1).

In addition, the MInT software package includes versions of applications such as VIC, vat, NEVOT [10], etc., which have been modified to include to internal interfaces to MInT. Together, these applications are capable of generating a sufficiently large quantity of network traffic — traffic which is necessary in order to try and assess the real value of dynamic QoS negotiation and re-negotiation (requirement R2). Some of these applications also include well-defined APIs for configuration (requirement R4). It remains to be seen whether MInT offers truly *unique* possibilities for experimentation with QoS negotiation (requirement R5).

In addition to the work with MInT, an associated student will look at some aspects of the roles of the user in QoS negotiation. He has chosen to use CMT as a platform for verifying his results. His work will be conducted within the ENNCE framework. For his purposes most of the requirements mentioned are fulfilled. Moreover, his work shall be directed at employing CMT as a single, media streaming application; a level of effort which is appropriate within the frame of a diploma thesis.

6. References

[1] W. Leister and P. Spilling, “A Service Agent for Connection and QoS Management in Multimedia Systems”, NR Technical Note IMEDIA/05/98, Norsk Regnesentral, Oslo, 1998.

[2] A. Vogel, B. Kerherve, G. v. Bochmann, and J. Gecsei, “Distributed Multimedia Applications and Quality of Service - A Survey”, IEEE Multimedia, Vol. 2, No. 2, pp. 10-19, 1995.

[3] R. Braden, D. Clark, and S. Shenker, “Integrated Services in the Internet Architecture: an Overview”, IETF, RFC 1633, 1994.

[4] Aurrecoechea, C., Campbell, A., Hauw, L., “A Survey of Quality of Service Architectures”, Multimedia Systems Journal, May 1998 (special issue on QoS Architecture), 1998. See also: Technical Report, Lancaster University, ftp://ftp.comp.lancs.ac.uk/mpg/MPG-95-18.ps.Z, 1995.

[5] Gecsei, J., “Adaptation in Distributed Multimedia Systems”, IEEE Multimedia, April-June 1997, Vol. 4, No. 2 , pp. 58-66.

[6] Sisalem, D., User Handbook of the Multimedia Internet Terminal (MInT), available on http://www.fokus.gmd.de/research/cc/glone/products/MInT/

[7] CMT, The Berkeley Continuous Media Toolkit, Version 4.0 http://bmrc.berkeley.edu/projects/cmt/versions/4.0/, University of California, 1998.

[8] VIC, http://mice.ed.ac.uk/mice/archive/vic.html

[9] VAT, http://mice.ed.ac.uk/mice/archive/vat.html

[10] NEVOT, http://www.cs.columbia.edu/~hgs/nevot

[11] Dave Thaler, MBone Software Archives, http://nic.merit.edu/~mbone/output.html.. 1996.

[12] Steinar Kristoffersen, Developing Collaborative Multimedia.The MEDIATE Toolkit, PhD-thesis, Lancaster University, 1998. Also: NR Report 930, Norsk Regnesentral, Oslo, July 1998.

[13] Ole Sunde: Video Transmission over Wide Area ATM Networks, Project description, http://www.unik.no/~olesu/hfag/project.html, 1999

[14] Lars Aarhus, Jannicke Riisnæs, Tore Solvar Karlsen: An Experimental Network Infrastructure Supporting QoS, NR Technical Note IMEDIA/02/99, Oslo 1999.

[15] Thomas Plagemann, Knut A. Sæthre, Vera Goebel: Application Requirements and QoS Negotiation in Multimedia Systems, Proc. 2nd Workshop on Protocols for Multimedia Systems, Salzburg, Austria, October 1995, pp. 411-426.

[16] Sørensen, Tryggve E: «LAVA Player design and implementation». NR Technical Note IMEDIA/02/1997, January, 1997.