Mapping User Level QoS from a Single Parameter

Antony Richards, Glynn Rogers
(arichard,grogers)@tip.csiro.au
CSIRO Telecommunications and Industrial Physics,
http://www.tip.csiro.au/~arichard

Mark Antoniades, Varuni Witana
(marka,varuni)@dstc.uts.edu.au
CRC for Distributed Systems Technology,
University of Technology, Sydney.

Corresponding Author:
Glynn Rogers
grogers@tip.csiro.au
CSIRO Telecommunications and Industrial Physics,
PO Box 76, Epping, 2121, NSW, Australia.

Abstract

An end to end network QoS architecture that presents a simple user interface is justified, designed and implemented. A user is required to control only a single parameter, the cost. The "satisfaction" concept has been introduced to quantify the QoS provided by the system. The transformations required to both map the cost into satisfaction and then configure the system are then developed.

Keywords: QoS, QoS Interfaces, User QoS, QoS Mapping/Translation, User Perception, Multimedia.

Submission Area: Quality of service management. (Related areas include: Multimedia network management models and architectures, Trials, case studies and experiences, and Resource management in broadband networks).

1 Introduction

The advent of distributed multimedia applications running from desktop PCs and workstations has given rise to the need for Quality of Service (QoS) support within computer networks. Current developments in network technology are aimed at providing a variety of facilities which will enable users to choose the level of QoS support appropriate for the application. From a technical perspective, the central QoS issue is then the provision of predictable and coordinated access to system resources, encompassing both the end systems and the network.

For an application to make use of system services, such as the facilities for negotiating QoS levels in ATM Networks, the application's resource requirements need to be specified. Because quality of service is ultimately a matter of perception on the part of the application user, this implies a need, at least in principle, for the user to specify the system resources required to support the desired level of QoS as perceived by the user. However, in practice, this places an impossible burden on the user who generally has no knowledge of the technical issues involved. Consequently, to minimise the complexity of the user's task, a system or application management resource is required which performs a mapping from this high level space of user perceptions to the low level space of system parameters.

Of course direct system access to this high level space is impossible and communicating the user's QoS requirements to the application requires some form of interactive GUI. The QoS control 'knobs' on this GUI should be kept as simple as possible.

The principal aim of the work reported here is to investigate the general characteristics of a method for systematically reducing the QoS controls that need to be exercised by the

user to just two - the level of user satisfaction with the delivered QoS and the price that the user will be required to pay for the service. We explore the nature of the relationships between these high level variables and the application level variables, such as frame rate and image resolution, as well as the lower level variables such as bandwidth. The ultimate aim is to enable the system to be able to configure itself by taking into account the relative importance to the user of variables such as frame rate and resolution based on the user's setting of the satisfaction or cost controls.

2 Related Work

There are three areas of research directly relevant to this paper in the sense that they create the context of the work reported here. These are - classification of video, network characteristics and QoS models. This section will briefly describe the related work in each area in order to provide an outline of this context.

2.1 Video Classification

Work by Apteker [2] has shown that video clips can be classified according to the importance of its content in terms of the following factors:-

Temporal Data Rate (T)**:** Does the clip change significantly over time? For example, a talk show would have a low temporal data rate.

Auditory (A)**:** Is the auditory component important for the understanding of the clip?

Visual (V)**:** Is the visual component important for the understanding of the clip?

As each factor can be either high or low, this scheme allows a clip to be classified into one of eight categories. Table 1 gives an example of a clip in each category.

	T_{lo}	T_{hi}
$A_{lo}V_{lo}$	Logo/test pattern	Station break
$A_{lo}V_{hi}$	Snooker	Sporting highlights
$A_{hi}V_{lo}$	Talk show	Advertisement
$A_{hi}V_{hi}$	Stand-up comedy	Music video

Table 1: Video Classification Examples.
Taken from [2].

The video perception trials performed in [2] allowed Apteker to conclude that the viewing satisfaction (the term watchability was used) of a clip is dependent on the category it is placed in. This knowledge can be used to configure the QoS parameters of a video on demand system.

2.2 Network Models

There appear to be three mainstream approaches to the question of providing QoS support within networks.

1. The ATM Forum [3] is developing a set of specifications for a network technology which can enable each user to specify the detailed QoS characteristics of individual virtual connections associated an application. These characteristics are very low level; bounds on cell loss rate, end to end delay and delay jitter. In addition the user has a choice of four Service Categories which provide quite different network behaviours ranging from the circuit emulation Constant Bit Rate (CBR) category through the feedback control Available Bit Rate (ABR) to the best effort like Unspecified Bit Rate (UBR). The result is a very tightly managed, connection oriented network which can provide rigidly defined types of service.

2. A broadly similar approach is being adopted by the Integrated Services Working Group of the Internet Engineering Task Force (IETF) [11] although here the emphasis is exclusively on the provision of a small number of service classes. Three have so far been defined - the familiar Best Effort service, the Controlled Load service which provides the performance of a lightly loaded network, and the gilt edged Guaranteed service. The last provides hard bounds on performance - zero queueing loss (but not zero overall loss) and an end to end queueing delay bound. However, as distinct from the ATM Forum's approach, the characteristics of an individual connection are not directly specifiable by the user but instead are determined by the network with the user able to realise the bound by suitably shaping it's traffic. Nevertheless both the Integrated Service and ATM Forum approaches have in common the concept of end to end individually managed flows with reserved resources and a signalling mechanism.

3. This concept is specifically rejected by the Differentiated Services Working Group of the IETF which is developing a more decentralised model [4] in which the required network performance is specified in terms of the relative priority of groups of aggregated flows, Behaviour Aggregates, rather than the requirements of individual flows. Resource allocation is determined on the individual router scale, based on individual administrative domain policy (eg individual ISP's) using the concept of Per Hop Behaviours (PHB's). End-to-end services are envisaged as being constructed from PHB's via a set of bilateral agreements between domains in addition to the traffic management policies of the individual domains. The policies are expressed in terms of Behaviour Aggregates so that there is no notion of defining flows or virtual

connections with precommitted resources through the network.

It is clear that there is no unified model for the provision of Quality of Service at the network level. Indeed it is conceivable that there never will be. An application running over a corporate or ISP Intranet might, for example, have access to the full level of QoS support specified by the ATM Forum. Then, in order to access information over the Internet it may encounter either a Differentiated Services or Integrated services model. There is even a proposal, within the IETF, for Integrated Services over Differentiated Services!

The central point here is that our QoS Management system must be capable of interfacing with a variety of network QoS support mechanisms without this being explicitly evident to the user. Consequently our model is layered in order to clearly differentiate between those attributes that are related directly to the performance of the application itself, for example frame rate and image resolution, and those which indirectly influence the performance of the application such as network bandwidth and latency.

2.3 QoS Models

QoS models must operate end to end. That is, it is not sufficient to have QoS bounds between say just the network adapter cards, as the scheduling from the adapter card to the application may break the QoS requirements. Thus, QoS within a system must encompass all components of an application, from the network to the CPU and its scheduling.

Several models have been proposed to address QoS in such an environment. The QoS-A architecture from Lancaster University [5] specifies QoS in terms of flow-specs which contain low level QoS parameters such as burst durations and peak rates as well as delay

and loss.

The OMEGA model presented by Nahrstedt et. al. [8] has a layered QoS structure. These layers are split across the user/system boundary so that part of the QoS model is specific to the application (eg how to configure the video), while some of the QoS model is applicable to the whole system (eg bandwidth allocation). Parameter mapping (or translation as used by Nahrstedt) is required between the various layers of the model although Nahrstedt does not specify how to automate the mapping.

Fischer [7] has identified the need for QoS to be layered in a similar way to the OSI protocol stack with the user level QoS kept as simple as possible. While the generic parameters suggested include cost and quality, application specific parameters, such as the frame rate, were also included at the user layer, making the model non-generic. To simplify the user interface, the user QoS parameters have been put into arbitrary bands. For example the frame rate is expressed in terms such as very fast, fast, normal, slow, and very slow.

Similarly, Alfano [1] performed user perception experiments to manually classify the frame rate and resolution of video configurations into five quality bands. Audio was similarly classified into five bands. A mapping into the system levels was performed through measurement of each possible configuration.

Further work by Fischer [6] has led to a framework called *Cooperative QoS Management* which concentrates on multicast applications. The user interface presented is similar to the one developed in this paper in that it has sliders to set the minimum and ideal values of the application specific parameters such as frame rate, colour and resolution. The Importance parameter needs three values to be set per application specific parameter.

However, the work does not present a method of converting from a user parameter into the system configuration, or the parameters required to reserve the resources.

Thus, although work exists that shows that QoS should be layered, and that the QoS parameters must be mapped between the layers, automatic techniques have not been developed to perform this mapping and configure the system. The work presented in this paper addresses this issue.

3 QoS Mapping

Figure 1 shows the three layer QoS architecture that has been developed. The Resource layer represents the components of the computer system, such as the network and operating system, as well as the QoS parameters that are used to configure the system for a particular purpose (eg bandwidth, delay bounds, CPU priority, etc.). Above this, the Application layer represents the application specific parameters that determine the operation of the application as observed by the user, eg frame rate, image resolution, coding distortion etc. At the top, the User Level represents the perceptual domain of the application user, a full characterisation of which clearly must await future developments in psychophysics. For present purposes we characterise this layer by a single, empirically determined parameter, User Satisfaction. The translation function between the Application and User layers provides the many-to-one mapping between the application specific parameters and the user satisfaction parameter. Likewise, a translation function is required between the Application layer and the Resource layer.

Clearly the satisfaction of the user is constrained in practice by limited access to system resources at the Resource layer. While these limitations will be imposed by a

variety of mechanisms we assume for the purpose of this paper that they are represented to the user by a single parameter, cost. The mathematically under-determined problem of mapping a specified satisfaction to a number of application parameters is then solved by the standard technique of regularisation using this cost functional [10].

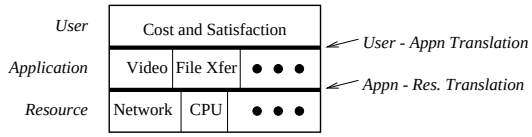


Figure 1: The Layered QoS Model Used

4 The User–Application Translation Function

As described above the purpose of this translation function is to enable the user to choose a level of satisfaction and have that level of satisfaction delivered by the system at minimum cost. However it is clear that the many to one mapping from the application parameters to satisfaction is a functional of the same form as the cost functional. Consequently the operation of the translation layer is symmetrical with respect to cost and satisfaction. In other words a user can specify the price (s)he is prepared to pay, and the translation layer will give the best satisfaction that can be supported or, alternatively, if the satisfaction is specified, the translation layer will give this at minimal cost.

In order to implement this concept a systematic scheme is required to translate the application specific parameters to the satisfaction level and to the cost. This section will describe the application specific parameters that need to be set, and the mathematical form of the functionals that we have chosen to perform the translation. While specifying the precise nature of these functions must await further research, we have chosen mathematically convenient forms which we believe sufficiently capture the general nature of these

functionals to provide an adequate basis for investigating the software structures required. The approach we have adopted is firstly to relate each application specific parameter to an individual component satisfaction, and then to combine the component satisfactions in a functional to give the final, single, satisfaction level. An identical approach is adopted for cost.

4.1 Translating a Single Application Parameter to its Component Satisfaction

The component satisfaction s_i is a function of the single application specific parameter x_i ie

$$s_i = g_i(x_i)$$

While determining the specific nature of this function requires either a carefully designed program of subjective testing or some body of psychophysical theory (yet to be developed) it is possible to make some observations about the general form of this function on the basis of common experience. As illustrated in Figure 2:

1. there is a lower bound on x_i. Below this value the satisfaction is negligible and can be approximated as 0,

2. there is a upper bound on x_i. Above this value, the increase in satisfaction is negligible, and s_i can be approximated as 1,

3. between these two bounds, s_i is a monotonically increasing function of x_i.

We define the lower bound on x_i as the minimum acceptable level, M, and the upper bound as the ideal level, I.

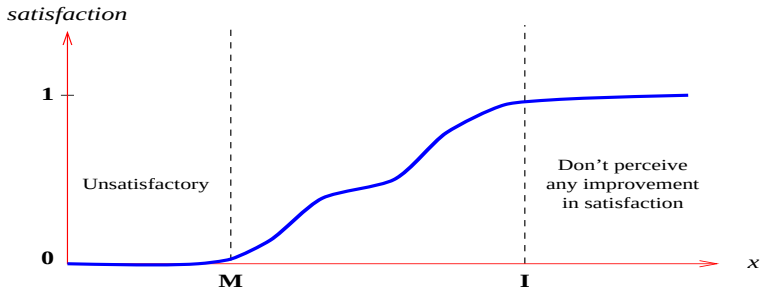


Figure 2: The Generic Translation from an Application Specific Parameter to its Satisfaction.

In the absence of specific empirical or theoretical information on the form of the functions g_i, our approach is to select a standard mathematical function which has the above characteristics and which can be easily parameterised to provide a family of specific functions. The selection of particular parameter values can be left to a subsequent program of empirical testing or incorporated in the software as a user setup function. While this approach does not capture the full richness of the relationship between the application parameters and the perceptions of the user, we believe that it provides a sound vehicle for exploring system concepts and for initial experimental testing.

The logarithmic function is attractive because it not only satisfies the criteria described above but it also reflects the intuitive notion that a fixed increase in x will have a greater impact on s when x is small than when x is large. For example, the increase in satisfaction when changing from a frame rate of 1 fps to 2 fps is greater than that when changing from 24 fps to 25 fps. In other words the basic nature of the log function is that multiplying x by some factor increases s by an additive amount.

While other functions would undoubtedly produce satisfactory results we have adopted the logarithmic function as the basis of the work reported here. We believe it to be

consistent in general with the subjective experiments that have been performed by others ([2, 9]).

The parameterised form of the logarithmic function used here is:

$$s(x) = a.ln(bx + c)$$

where:

$$a = \frac{1}{p - 10},$$

$$b = \frac{e^{\frac{1}{a}} - 1}{I - M},$$

and

$$c = \frac{I - Me^{\frac{1}{a}}}{I - M}.$$

This has a single parameter p which can be interpreted as a sensitivity parameter in the sense that p controls the sensitivity of the component satisfaction to variations in x for low values of satisfaction. We anticipate that it is in this region of low satisfaction levels (but low cost) that the greatest user demands on system response will occur. Figure 3 shows this family of logarithmic curves for a range of sensitivity values. Note the common boundary conditions $s(M) = 0$ and $s(I) = 1$.

Note also that for $p = 10$ there is a linear relationship between satisfaction and application level parameter value. This case needs some special care because here $a = \infty$ and $b = c = 0$. By writing the argument of the logarithm as;

$$\frac{b(x - M)}{I - M} + 1,$$

using the series expansions for $\log(x+1)$ and e^{p-10}, and then taking the limit, it is readily shown that:

$$s(x) = \frac{x}{I-M} - \frac{M}{I-M}.$$

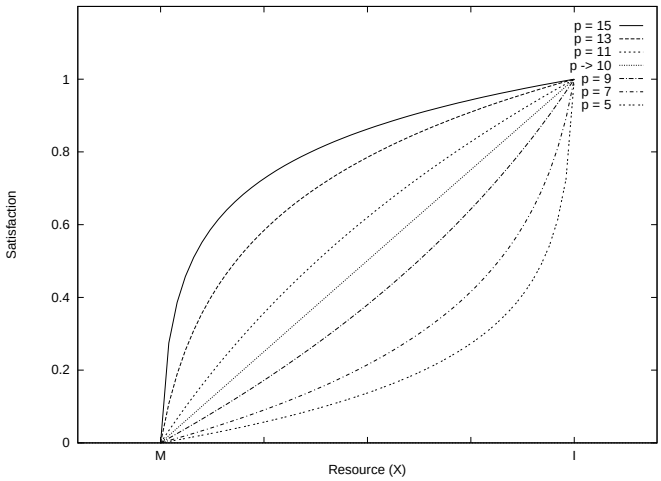


Figure 3: The Logarithmic Family of Curves

4.2 Combining Component Satisfaction Variables

The overall satisfaction of a user is expressed as a function of the individual component satisfactions s_i. That is,

$$S_{tot} = f(s_1, s_2, \ldots, s_n) = f(g_1(x_1), g_2(x_2), \ldots, g_n(x_n)).$$

Desirable properties of $f(s_1, s_2, \ldots, s_n)$ include:-

1. If s_1 is much smaller than each of the $\{s_2, \ldots, s_n\}$, then S_{tot} must be dominated by s_1. For example, if a sports video has a very high resolution, but plays back at 1 fps, then the overall satisfaction is low.

2. $f(s, s, \ldots, s)$ must be equal to s which allows $f(s_1, \ldots, s_n)$ to be scaled.

It is also desirable that the calculation of S_{tot} scales with respect to n. That is, the complexity of calculating S_{tot} must be less than, or equal to $O(n)$.

One relationship that satisfies these requirements is

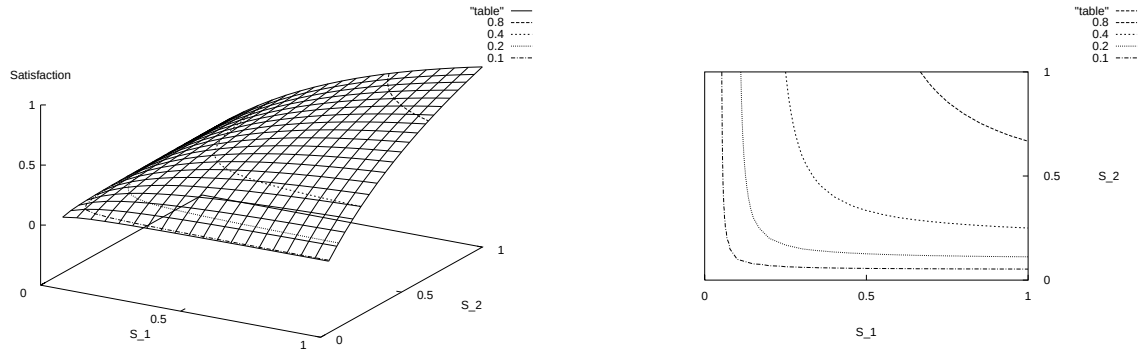
$$\frac{1}{S_{tot}} = \frac{1}{n} \times \sum_{i=1}^{n} \frac{1}{s_i}.$$

That is,

$$S_{tot} = \frac{n}{\sum_{i=1}^{n} \frac{1}{s_i}}.$$

Note that this formula is similar to the formula for the equivalent resistance of parallel resistors, except that the result is multiplied by n so that requirement 2 is satisfied.

S_{tot} as a function of s_1 and s_2 $(n = 2)$ is shown graphically in figure 4 as a two dimensional surface and in contour form. Note the symmetry inherent in the function.



(a) (b)

Figure 4: Combining Satisfactions.
(a) gives a three-d view of the graph, while (b) gives a contour map. (Lines on the contour map represent point of equal S_{tot}.) The x and y axis are the component satisfactions.

However when S_{tot} is a function of the application specific variables x_i the shape of the surface is determined by the setting of the sensitivity parameters p_i. For example, figure 5 illustrates the asymmetry in the satisfaction contour graph as a function of x_1

and x_2 when $p_1 = 3$ and $p_2 = 9$. The acceptable resource level (M) was zero for both x_1 and x_2, while the ideal resource level (I) was 1.

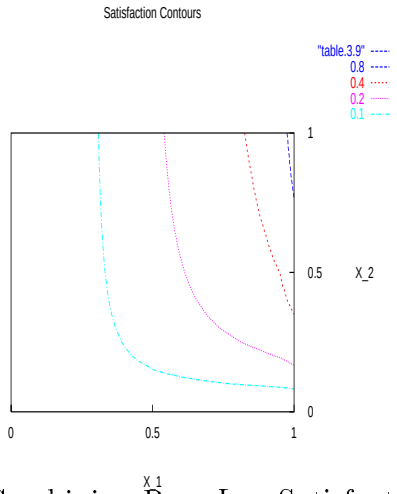


Figure 5: Combining Base Log Satisfactions.
The sensitivity of x_1 is 3, while the sensitivity of x_2 is 9.

4.2.1 Choosing an Application Level Operating Point - the Cost Functional

As noted above, the function $S_{tot} = f_x(x_1, x_2)$ is a many to one mapping and the problem of choosing a specific set of values for the Application Specific variables to realize a particular satisfaction S is under-determined. In other words all of the points (x_1, x_2) lying along the contour $S_{tot} = S$ represent feasible solutions to the problem.

The standard mathematical approach to such problems is to introduce a second functional, a regularising functional, which enables a single solution to be obtained. This is achieved by choosing the form of the regularising functional so that its specific numeric values along any contour of the first functional go through a single minimum (or maximum). The problem of determining an operating point is then reduced to a standard minimisation (or maximisation) problem.

For this case it is very natural to introduce a regularising functional in the form of a cost. Each of the application specific entities consumes system resources and consequently

incurs a cost. For example maintaining a particular frame rate costs CPU cycles and network bandwidth and, in a user pays environment, these costs will ultimately appear in monetary terms.

Now while we do not presume to anticipate the specifics of this costing process, it is possible to anticipate the general form of the costing relationship. Indeed the considerations described above for determining the form of the satisfaction functions apply here as well.

There is, however, a further constraint on the form of the cost functional. As illustrated in figure 6, in order for there to be a unique minimum the constant cost contours must intersect the constant satisfaction contours at two points except at the minimum where the two contours share a common tangent. Furthermore as is clear from the figure (and as can be demonstrated mathematically) that for this point to be a minimum (as distinct from a maximum) the second derivative of the satisfaction contour must be greater than that of the cost contour at this point.

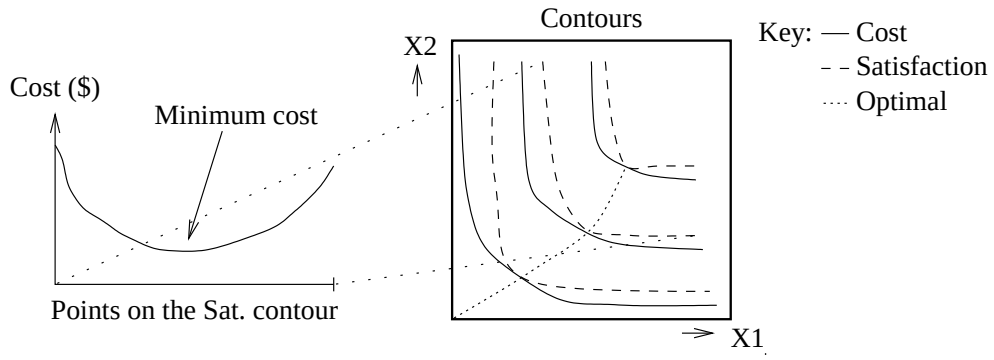


Figure 6: Optimising the Cost for a Given Satisfaction.

If these conditions are satisfied the resulting functionals can be used to perform one of two separate optimisations - either the cost is minimised for a specified level of satisfaction *or* the satisfaction is *maximised* for a specified cost. In either case the locus of optimum

points (the dotted line in figure 6) provides the required one to one mapping between the satisfaction (or cost) and the application specific variables.

Note that if the contours are symmetric about the $x_1 = x_2$ line, then the locus of optimal operating points degenerates to the line $x_1 = x_2$.

5 Implementation Details

In order to validate the QoS Framework, the translation mechanism described above has been incorporated in an experimental Video on Demand (VoD) system called SuperNOVA implemented in Java.

5.1 The Main Menu

Figure 7 shows the main menu of the program. Once a clip has been selected, a user decides how much (s)he is prepared to pay for it by moving the cost slider (the User layer of figure 1). Through the algorithms described in this paper, the satisfaction LED bargraph is then calculated.

Behind the scenes, the cost slider selects the constant cost contour and a satisfaction optimisation is performed along this contour. These calculations give the satisfaction that will be achieved, and the configuration of the transmitted video (frame rate and resolution).

Alternatively this operation can be performed to minimise cost for a specified satisfaction.

The user–application translation calculations are performed offline, and the results

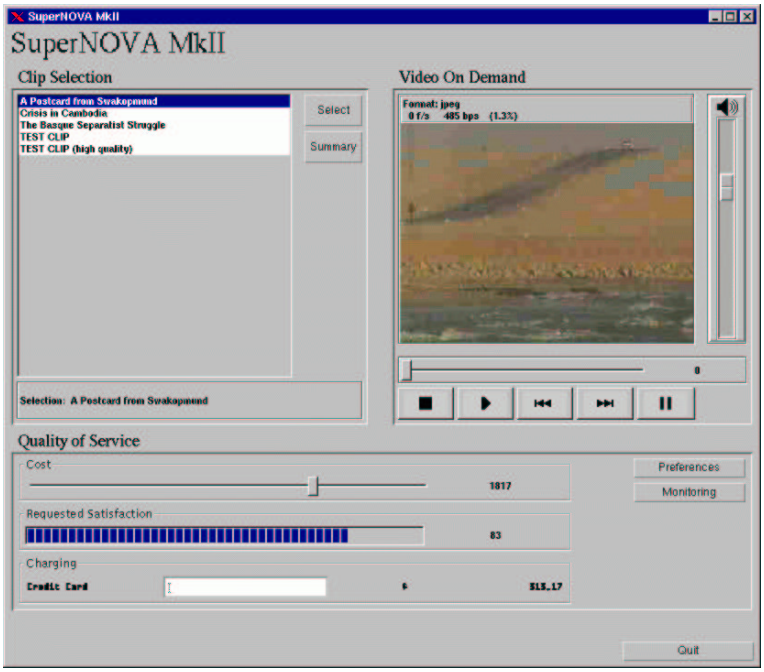


Figure 7: Main Menu

are accessed through a lookup table that is dependent on the clip, and the defined range and sensitivity of the frame rate and resolution. This table contains the locus of optimal points of figure 6.

For this experimental implementation, the application–resource translation has been estimated by assuming that network bandwidth (and hence cost) is proportional to both frame rate and resolution. A table is generated using the bandwidth of the clip's highest resolution and frame rate to determine the constant of proportionality. This table contains the bandwidth and satisfaction for every possible frame rate and quantisation level (noting that frame rate and quantisation level are both quantised variables).

The table entries are then sorted according to increasing satisfaction so that points (on the contour) of equal satisfaction are adjacent. When following the sorted list, the algorithm in table 2 is applied recursively to remove non-optimal operating points. The remaining points constitute the final lookup table.

Condition	Response	Reason
$b_1 \geq b_2$	Discard Operating Point 1	Operating point 1 has a higher bandwidth than 2, yet it produces the same, or less satisfaction. Thus point 2 is a better operating point.
$b_1 < b_2$	Keep Operating Point 1	Operating point 1 has a lower bandwidth than point 2, but its satisfaction is also lower.

Table 2: The Algorithm to Cull the Non-Optimal Operating Points.
The satisfaction of entry 1 (s_1) is less than or equal to the satisfaction of entry 2 (s_2). b_i is the estimated bandwidth of entry i ($i = 1, 2$).

5.2 Setting the Sensitivity

The sensitivity and range of the video attributes (ie the frame rate and resolution) are given default values when the clip is digitised, based upon the clip type (see section 2.1).

The *Preference* button on figure 7 pops up another menu, shown in figure 8. This menu, which operates at the Translation layer of figure 1, allows the setting of acceptable and ideal levels for each of the video attributes as well as setting the sensitivity levels for the different attributes.

These values are then used as parameters for the algorithm described in section 4.

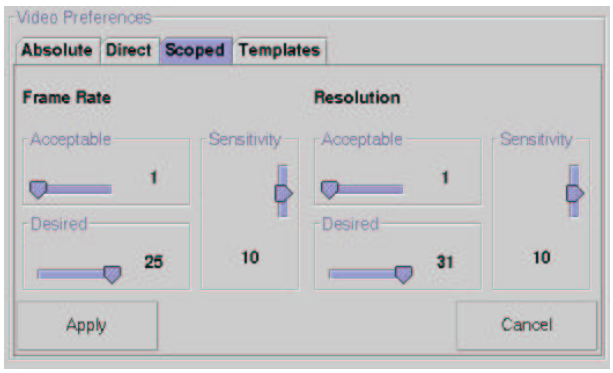


Figure 8: Preference Menu

5.3 Projected Use

The User interface described above can be viewed at different levels. The naive user would only need to use the main menu and adjust the Cost or Satisfaction sliders depending on how much he/she was prepared to pay (figure 7).

It is expected that only an expert user will actually use the interface in Figure 8. These parameters would be pre-set by the content provider (by a trained user) depending on the clip type.

6 Further Work

This paper has been using theoretical curves for satisfaction (ie the log curves). Empirical experiments will allow this curve to be replaced with ones that better match the human visual system.

The system then needs to be extended to work with two resources, network bandwidth and CPU capacity. Finally, a diversity of applications need to be developed.

7 Conclusions

The need for a QoS Framework that requires only very simple manipulations from a user was outlined. A user interface based on a cost or satisfaction control is proposed. Mathematical relationships have been developed to allow a transformation from this into the configuration of the system and the resources required to support it.

Acknowledgements

The authors would like to thank Diet Ostry for helpful discussions. The work reported in this paper has been funded by CSIRO and the Co-operative Research Centre Program through the Department of Industry, Science & Tourism of the Commonwealth Government of Australia.

References

[1] M. Alfano and R. Sigle. "Controlling QoS in a Collaborative Multimedia Environment". In *"Proc. 5th IEEE International Symposium on High-Performance Distributed Computing"*, Aug. 1996.

[2] R. T. Apteker, J. A. Fisher, V. S. Kisimov, and H. Neishlos. "Video Acceptability and Frame Rate". *"IEEE Multimedia"*, 2(3), Fall 1995.

[3] The ATM Forum Technical Committee. *Traffic Management Specification Version 4.0, af-tm-0056.000*, Apr. 1996.

[4] D. Black, S. Blake, M.Carlson, E. Davies, Z. Wang, and W. Weiss. An architecture for differentiated services. INTERNET-DRAFT, May 1998. draft-ietf-diffserv-arch-00.txt.

[5] A. Campbell, G. Coulson, F. Garcia, D. Hutchinson, and H. Leopold. "Integrated Quality of Service for Multimedia Communications". In *"Proc. IEEE INFOCOM'93"*, San Francisco, Mar. 1993.

[6] S. Ficher, M.-V. Salem, and G. v. Bochmann. "Application Design for Cooperative QoS Management". Technical Report #1067, Université de Montréal, May 1997.

[7] S. Fischer and R. Keller. "Quality of Service Mapping in Distributed Multimedia Systems". In *IEEE International Conference on Multimedia Networking*, pages 132–141, Aizu-Wakamatsu, Japan, Sept. 1995.

[8] K. Nahrstedt and J. Smith. "A Service Kernel for Multimedia Endpoints". In *"Proc. IWACA '94"*, Sept. 1994.

[9] M. Ott, G. Michelitsch, D. Reininger, and G. Welling. An architecture for adaptive QoS and its application to multimedia system design. *Computer Communications Special Issue on Guiding Quality of Service into Distributed Systems*, 1997. http://ccrlwww.nj.nec.com/key/97-R-005/publication/abstract2.html.

[10] A. N. Tikhonov and V. Y. Arsenin. *Solution of Ill–Posed Problems.* New York: Winston, Wiley, 1977.

[11] P. P. White and J. Crowcroft. The inegrated services in the internet: State of the art. *Proceedings of the IEEE*, 1997.