1 QUALITY OF SERVICE SEMANTICS FOR MULTIMEDIA DATABASE SYSTEMS

Jonathan Walpole, Charles Krasic, Ling Liu,
David Maier, Calton Pu, Dylan McNamee, and
David Steere

Department of Computer Science and Engineering
Oregon Graduate Institute *

***Abstract**: Quality of service (QoS) support has been a hot research topic in multimedia databases, and multimedia systems in general, for the past several years. However, there remains little consensus on how QoS support should be provided. At the resource-management level, systems designers are still debating the suitability of reservation-based versus adaptive QoS management. The design of higher system layers is less clearly understood, and the specification of QoS requirements in domain-specific terms is still an open research topic. To address these issues, we propose a QoS model for multimedia databases. The model covers the specification of user-level QoS preferences and their relationship to QoS control at the resource-management level, and is applicable to adaptive and reservation-based systems. In this paper we present the model, discuss the implications it has for multimedia database design, and describe a practical implementation of it.*

* This research was supported in part by DARPA contracts/grants N66001-97-C-8522, N66001-97-C-8523, and F19628-95-C-0193, and by Tektronix, Inc., and Intel Corporation.

1. INTRODUCTION

Interest in QoS management has grown with the arrival of multimedia systems whose resource consumption needs often exceed the available resource capacity of the systems on which they are deployed [1]. Rather than refusing to return a presentation in such situations, multimedia systems with QoS support have the capability of returning reduced-quality presentations using the resources that are currently available. Reduced-quality presentations exhibit inaccuracy compared to perfect-quality presentations. Such inaccuracy might take the form of dropped or delayed video frames or audio samples, reduced spatial resolution, or decreased peak signal to noise ratio (PSNR), in order to allow the presentation to be made using fewer system resources [2,3,4].

High-level control over QoS management can be provided by allowing applications or users to specify their tolerance for inaccuracy in the presentations returned, or by giving them control over how such inaccuracy is introduced when resources become scarce. Essentially, QoS control gives higher system layers the ability to specify what constitutes an *acceptable* presentation generated by the lower layers, and to define what *better* means between two alternative presentations.

For systems with resource reservation capability in their lower layers, the QoS specifications generated by higher layers can be translated, together with other information into the necessary resource reservation requests [5]. Adaptive systems take a different approach and use high-level QoS information to determine the best way to adapt presentation quality in response to uncontrolled changes in available resource capacity [6,7,8,9].

Although adaptive and reservation-based systems take quite different approaches at the resource management level, both classes of system require a QoS model that allows application-level QoS requirements to be specified and then related to resource usage plans. This paper presents such an application-level QoS model and discusses its use at the resource management level.

Our QoS model allows the definition of multiple quality dimensions for multimedia presentations. Users specify their QoS requirements by defining utility functions for each dimension. Utility functions map quality to utility, define thresholds for upper and lower bounds on useful quality, and can be weighted and combined to specify overall QoS requirements. This information is then used in the derivation of resource management requirements for either reservation-based or adaptive systems.

The proposed QoS model serves a number of purposes, including its uses as a guide to system architecture, a criterion for distinguishing among presentation plans, and a control input for feedback-based QoS management. There are several semantically relevant implications of the model. QoS specifications are one way of saying what the most important elements of the data are, so they can be emphasized in both capture and retrieval. The model

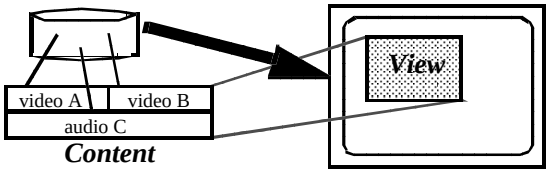
supports a distinction between semantics of data and semantics of query results. For example, the quality of stored data is distinct from the quality requirements for a particular instance of viewing the data. Our model also serves as a guide for deciding what meta-data should be attached to data and what information should be associated with uses of the data.

The rest of the paper is organized as follows. Section 2 presents our QoS model in more detail. Section 3 describes our use of the model in a real-world system that interactively delivers multimedia presentations from a remote storage server. We compare our model to other QoS models in section 4. Section 5 concludes the paper and discusses future work.

2. The Quasar QoS Model

The Quasar QoS model extends Staehli's QoS model [10,11,12]. Staehli defines the concepts of *content*, *view* and *quality*. Content is a composition of single-medium segments into a complete presentation, as specified by the creator of the presentation. For example, a content definition might specify 3 seconds of video source A followed immediately by 4 seconds of video source B, both proceeding in parallel with audio clip C. The view definition specifies an idealized mapping of content into a display space by a consumer of a presentation. A view might indicate that the video portion of the presentation should appear in a certain 6x8 cm rectangle on the screen or that the presentation should proceed at half normal speed.

The ability to specify view as separate from content is essential when the creator of a presentation can foresee neither all uses to which it will be put



Quality *= error in view*

Figure 1: Content, view, and quality

nor the environments where it will be played.

A quality definition in Staehli's model specifies the allowable divergence between the actual presentation delivered to the consumer and an ideal presentation as specified by content and view (see Figure 1).

Like Staehli's model, our model is based on the notion that the quality of a query result is a measure of the amount of error present in it. Our model takes the abstract view that queries return results that are approximations to real-world values, and that real-world values exist in a continuous space.

Under this model, a query returning a perfect result would return an error-free replica of this continuous space. However, computer-based storage, manipulation, and presentation requires that (a) real-world values be captured using equipment of limited accuracy, (b) they be represented digitally using a finite number of bits, and (c) computer and network resources be used to deliver the digital representation of the result to the user.

We use the term *capture error* to describe the class of errors that result from the use of inaccurate capture equipment. These errors are incidental in the sense that they depend on the characteristics of the specific capture equipment used, and may not be present when different equipment is used.

Other errors are inherent in the digital representation of continuous data. We use the terms *quantization error* and *sampling error* to describe the classes of inherent errors that result from the use of a finite number of samples and a finite number of bits per sample, respectively, to represent time-varying values from a continuous space.

We use the term *delivery error* to describe the class of errors introduced by resource management decisions that influence the delivery of query results. Delivery errors in multimedia presentations include shift, rate and jitter errors caused by, for example, page and packet-oriented data transfer, buffering delays and resource scheduling policies.

Capture, sampling, quantization and delivery errors account for the difference between the perfect continuous representation of a real-world value and the value returned by a query result that is intended to represent it. As technology advances, computers become faster and have higher precision, enabling continual improvements in the quality of the query results returned by systems. However, for a particular system, even though a query result that satisfies a user's quality requirements may be considered *as good as perfect*, from the perspective of that user, according to our model it is not possible for a system to return a truly perfect result in general, since that would require infinite resources. This concept of perfect quality provides a reference point that allows quality improvements, as well as quality degradations, to be described, and user QoS requirements, data QoS characteristics, and system QoS capabilities to be specified independently of each other.

Our model separates the QoS characteristics of delivered presentations from those of the underlying, stored, digital representation of the data. We refer to the QoS characteristics of a delivered presentation as *apparent quality*, and the QoS characteristics of the stored digital content as *latent quality*.

2.1 QoS as a Distance Measure

Since presentations are often used to represent reality, we model the space of possible states for a presentation as a continuous, metric space.

Definition 1 (*presentation state space*)

Let *o* denote a target object of type presentation. The space of possible states for *o*, *Sp*(*o*), is a continuous metric space with the following properties:

Distance: a distance function *distance*(*u*,*v*) is defined over every pair of states *u*,*v in Sp* . The distance function describes the absolute value of the difference between two states of a presentation.

Symmetry: for every *u*,*v in Sp, distance*(*u*,*v*) = *distance*(*v*,*u*).

Triangle inequality:
for every *u*,*v*,*w in Sp, distance*(*u*,*v*) + *distance*(*v*,*w*) >= *distance*(*u*,*w*).

The above definition of the presentation state space as a metric space is useful because it allows further definitions of quality in terms of distance measures.

Definition 2 (*relative quality*)

For two possible presentation states, *u* and *v*, in *Sp(o)*, their relative quality is defined by *distance*(*u*,*v*).

Definition 3 (perfect quality)

The unique, error-free state in *Sp(o)* defines perfect quality for *o*.

This definition of perfect quality provides a common reference point from which to measure the relative qualities of different presentation states.

Definition 4 (*quality-loss*)

Let *p* be the perfect quality state for object *o* in *Sp*(*o*), and *q* be another presentation state of *o*. Quality-loss is a normalization function, *quality-loss*(*q*), that maps the *distance* (*q*,*p*) to a real number in the range [0,1].

Given the properties of the presentation space, it is now possible to define a quality-loss based ordering on presentation states. The normalization of quality-loss is useful because it enables a uniform distribution of resource consumption levels across the range of quality-loss values.

Definition 5 (*latent quality*)

Let *l* be the presentation state of object *o* in *Sp*(*o*) originally captured in digital format. The latent quality of *o* is defined by *quality-loss* (*l*).

Definition 6 (apparent quality)

Let *a* be the presentation state of object *o* in *Sp* (*o*) experienced by the viewer. The apparent quality of object *o* is defined by *quality- loss*(*a*).

The latent and apparent qualities of a presentation are determined by the capture, sampling, quantization and delivery errors present in its stored and delivered representations respectively.

Definition 7 (*quality dimension*)

A quality dimension is a dimension of the presentation state space *Sp(o)*.

We use the term quality dimension to represent each distinct type of information in a presentation over which QoS control is possible. By "type of information" we mean aspects of the presentation such as its color depth, spatial or temporal resolution, which may be made available as QoS adaptation parameters.

Definition 8 (*dimensional quality-loss*)

Given the perfect quality presentation state *p* and another presentation state *s* of object *o* in *Sp(o)*, and a quality dimension *d*, the dimensional quality-loss of *s* is *distance (s,p)* along dimension d.

The use of quality dimensions and dimensional quality-loss are important for simplifying the specification of QoS requirements and the implementation of QoS control mechanisms. They allow aspects of a presentation with different degrees of importance to be distinguished from one another and insulate users and system builders from the complexity of the entire presentation state space.

Definition 9 (*quality dimension type*)

A quality dimension type is a grouping of quality dimensions with common characteristics.

Quality dimension types are defined to enable reuse and help enforce consistency among exposed quality dimensions that are similar. Quality dimensions can be categorized as being of the base types *capture*, *sampling*, *quantization* or *delivery*, or combinations thereof, according to the class of errors they introduce. To be useful in real systems, however, quality dimensions must eventually be defined in terms that are meaningful in the application domain. In our model, we allow application-specific quality dimensions to be defined as sub-types of the basic types outlined above. For example, a query result whose type is a single video stream might be described using four quality dimensions: frame rate, color-depth, horizontal and vertical resolution (see Figure 2)

Quality adaptations in the frame rate quality dimension can be implemented via frame dropping which reduces the temporal resolution of the video and maps to an adjustment of sampling frequency, and hence sampling error. Similarly, quality adaptations in the color-depth quality dimension can be implemented by changing the number of bits per pixel, which can be mapped to an adjustment of quantization error. Quality adaptations in the horizontal and vertical resolution dimensions can be implemented by changing the number of pixels used to represent the image,

hence adjusting the spatial res olution of the image. Dropping pixels can be viewed as an adjust ment of quantization in the image, or, taking the analog

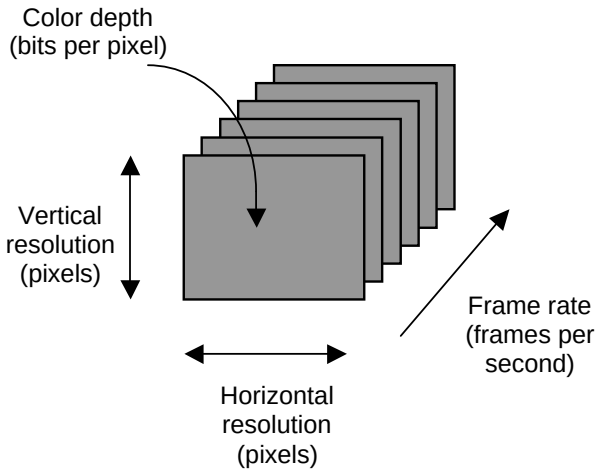


Figure 2: Example quality dimensions for video

video display view, as a reduction in the sampling frequency along scan lines.

Specific delivery quality dimensions include the *shift*, *rate* and *jitter* subtypes. Shift quantifies the amount of time a presentation is behind or ahead of schedule, rate quantifies the proximity of the actual play rate to the intended play rate of the presentation, and jitter quantifies the variation in rate of the presentation.

Some systems export quality dimensions that allow compound errors, from more than one class, to be introduced during QoS adaptation. An example in the video streaming domain includes an MPEG-1 to H.263 transcoding step, which is a lo ssy conversion of the video from one compressed format to another, perhaps with a new frame rate and spatial resolution. In this case, the quality dimension exported by the system combines cap ture, quantization and sampling errors.

2.2 Mapping Utility to Quality

The above definitions of the presentation state space and quality dimensions allow the quality-loss in a particular presentation state to be described either in quality dimension -specific terms, via the *dimensional quality -loss* function, or in abs olute terms via the *quality-loss* function. Dimensional

quality-loss functions impose a partial order on the presentation state space. This partial order can be translated into a total order by specifying the relative importance, or weighting, of different dimensions and different values within a dimension. This assignment can be viewed abstractly as a distortion of the presentation state space. The quality-loss function, as described above, imposes a total ordering on presentation states by assuming one particular distortion of the space. In a real system, this distortion could be based, for example, on the resource consumption requirements of the various presentation states. However, different distortions are likely to be appropriate to match the requirements of different users performing different tasks. We support the specification of user QoS requirements in our model by allowing a mapping of *utility* to dimensional quality-loss in each of the available quality dimensions.

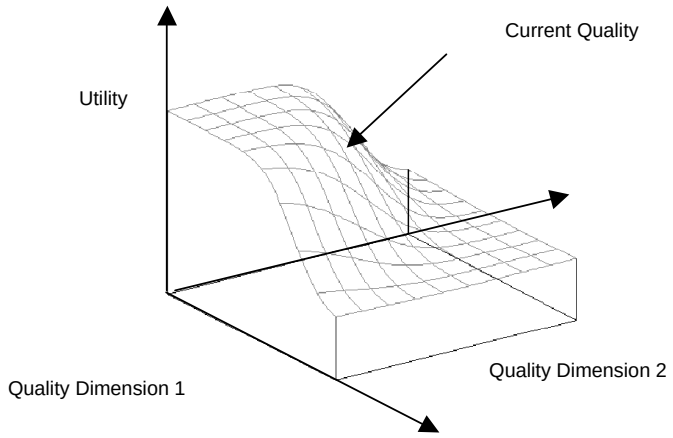


Figure 3: A two-dimensional quality surface

Definition 10 (*utility value*)

A utility value is a measure of usefulness, represented by real numbers in the range [0,1]. 0 represents useless, and 1 represents as good as perfect

Conceptually, the mapping of utility to presentation states defines a multidimensional *utility surface* that describes the usefulness of all possible states in the presentation state space (see Figure 3).

At any instant the quality achieved by the system defines a point on the utility surface. Given such a surface, the goal of a quality adaptation strategy would be to attain the highest possible position on the surface using the currently available resources and quality adaptation options.

In practice, however, it is difficult for users to specify their requirements in terms of a multidimensional surface, therefore we use the simpler, and more restrictive, approach of defining a separate *utility function* per quality dimension. These utility functions relate particular points in that quality dimension with their utility to the user.

Definition 11 (utility function)

A utility function, U_d, is a function that maps dimensional quality-loss distances in a single quality dimension to utility values.

Associated with each utility function are two thresholds, q_{min} and q_{max}, that describe the upper and lower bounds on useful quality, respectively (see

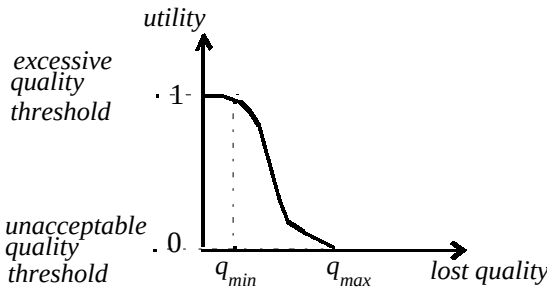


Figure 4: A utility function with thresholds

Figure 4).

q_{max} defines the point at which dimensional quality-loss has grown so large that utility equals zero (i.e., its is the lower bound on useful quality). q_{min} defines the point at which further decreases in dimensional quality-loss yield no further increase in utility because utility equals 1 (i.e., it is the upper bound on useful quality). Acceptable quality adaptations should ensure that quality remains between these thresholds in each dimension.

An approximation to the true overall utility value for a presentation state can be calculated by performing a weighted combination of the values returned by the utility functions for each of the state's quality dimensions.

Definition 12 (dimensional utility)

Given a presentation state, *s*, of object *o*, in *Sp(o)*, a quality dimension, *d*, and a utility function, U_d over that quality dimension, the value returned by $U_d(s)$ is the dimensional utility of *s* in dimension *d*.

Definition 13 (overall utility function)

Given a presentation state, s, a set U of dimensional utilities of s, and a set W of weights, we define the overall utility function, denoted by U_{all}(u,w), as follows:

U_{all}: U∗W ->$_C$ R

U_{all} takes a dimensional utility vector u in U and a weight vector w *in* W and returns a real number in R, where:

- $u = (u_1,u_2,...u_i,...,u_n)$, $1<=i<=n$, each u_i in U represents a dimensional utility,
- C is the $R =_{def} [0,1]$, and represents the weighted combination of dimensional utilities,
- constraint that if $\exists u_I$ such that $u_i = 0$ then U_{all} $(u,w) = 0$,
- $w = (w_1,w_2,...w_i,...,w_n)$, $1<=i<=n$, each w_i *in* [0,1] represents the weight that the dimensional utility u_i takes in the computation of the overall utility function.

The overall utility function described above can be used to reconstruct an approximation to the real utility surface.

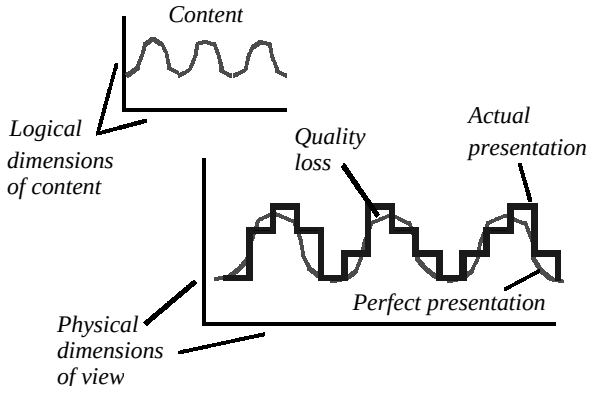


Figure 5: Content, quality and view concepts

2.3 The Quality-View Relationship

An important characteristic of the quality specification approach described above is that a user's quality requirements are not defined relative to the

quality of the stored content. Instead, they are defined in absolute terms, and hence the model supports independence among viewing requirements and stored content characteristics. However, as described so far, quality requirement specifications are still somewhat ambiguous. Consider the following example.

Example 1: A user defines a utility function for the frame-rate quality dimension of a video presentation. The x axis of the function is in units of 1/(frames per second), and the upper bound on useful quality is defined to be equivalent to 30 frames per second.

The ambiguity of this specification is rooted in the use of time in the x axis of the utility function. When the user refers to 30 frames per second, whose seconds are they referring to? Do they mean seconds of playout time (viewer's time), or do they mean seconds of content time (author's time)? If the video is being viewed at normal speed, i.e., the speed the author intended it to be viewed, both interpretations are equivalent. However, in real systems viewers generally have control over play speed, through controls for fast forward, slow motion, and reverse play, etc. When such controls are used, what effect, if any, should they have on quality? Specifically, in Example 1, when a user doubles the play speed, should more than 30 frames per second ever be received? If the specification of quality requirements is based on viewer's time they should not, whereas, if it is based on author's time they should.

One reason for interpreting QoS specifications relative to viewer's time is to prevent quality parameters, such as frame rate, from surpassing the viewer's perception level and wasting resources as view parameters, such as play speed, are increased.

Conversely, a reason for interpreting QoS specifications relative to the author's time is to preserve quality when query results are stored. Consider the case of a viewer storing the result of a query instead of, or in addition to, viewing it during retrieval. The latent quality of the stored result should be independent ofthe speed with which it was delivered.

Because of the issues highlighted in the discussion above, our QoS model, like Staehli's, distinguishes among *view* specification and *quality* specification. View specification is concerned with mapping the logical dimensions of the content, which were specified by the author, to real world dimensions specified by the viewer (see Figure 5).

Example 1 only discussed the time dimension, however, view specifications can also refer to other dimensions such as window size. We refer to the default mapping for these dimensions as the *identity view*, but expect viewers to have controls to over-ride the identity view in order to define *actual views* that match their specific viewing requirements.

The discussion above illustrates that quality specifications can be interpreted relative to the identity view or the actual view. The quality requirements of query results intended for immediate viewing only should normally be interpreted relative to the actual view, whereas the quality

requirements of query results intended for storage should be interpreted relative to the identity view.

2.4 Advantages of the Model

From a data semantics viewpoint the QoS model described above has several advantages. It identifies the data components that are important for storage and retrieval and separates quality of presentation from quality of stored representation.

The model also offers several degrees of independence that are useful in multimedia database systems. First, it supports independence among authoring and viewing concerns. At content creation time the author doesn't have to anticipate the viewer's eventual use of the presentation, or the capabilities of the system on which it will be viewed. Authors are concerned primarily with the specification of content, although they may specify default parameters for view and quality. Viewers can choose to use these defaults or over-ride them by specifying their own view and quality requirements.

Secondly, the model supports independence among viewing concerns and system capabilities. The key to providing this form of independence is the ability to specify degraded quality in order to make efficient use of scarce resources. Hence, the type of end-system the viewer has need not limit either the content or the view of the presentation. However, given a specific content and view, the capabilities of the system impose a limit on quality.

Third, since quality requirements can be specified with respect to either the identity or actual view, the model supports queries that retrieve presentations for immediate viewing as well as for storage.

Fourth, because quality is defined relative to a hypothetical perfect presentation with quality dimensions that are continuous, the model supports the quantification of the latent quality of the content. As technology advances and new capture devices, with higher precision and throughput, become available, the resultant quality improvements can also be quantified using this model. In contrast, an approach that defines quality requirements relative to the quality of the actual stored content requires a separate notion of quality to quantify the latent quality of the content. Furthermore, such an approach lacks independence between the specification of the viewer's quality requirements, the quality of the content, and the characteristics of the technology used to capture it.

The model also has the advantage of providing a basis for both hard guarantee and adaptation-based systems. Since we model requests for QoS adaptation using utility functions with upper and lower bounds on acceptable quality, a request for a hard QoS guarantee can be made simply by specifying both upper and lower bounds at the same quality level. In this case, utility functions are simple step functions.

Finally, the model illustrates a distinction between information that should be associated with data and information that should be associated with uses

of data. Information describing the latent quality of the data should clearly be attached to the data itself, as meta-data. Similarly, the quality dimensions that are made directly accessible by the representation of the data (for example, a layered video encoding) should be stored as meta data associated with the data itself. QoS specifications, comprised of utility functions and parameters for combining them, should clearly be associated with uses of data, rather than associated directly with the data itself, as should view specifications.

3. Practical Implementation of the Model

3.1 Architecture Overview

We have implemented a prototype multimedia system based on our model

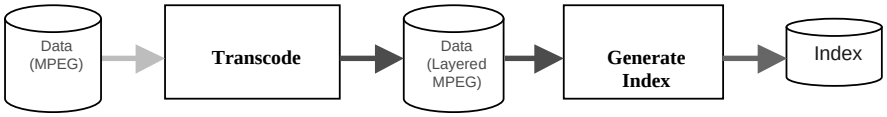


Figure 6-a: Off-line components of QoS adaptation prototype

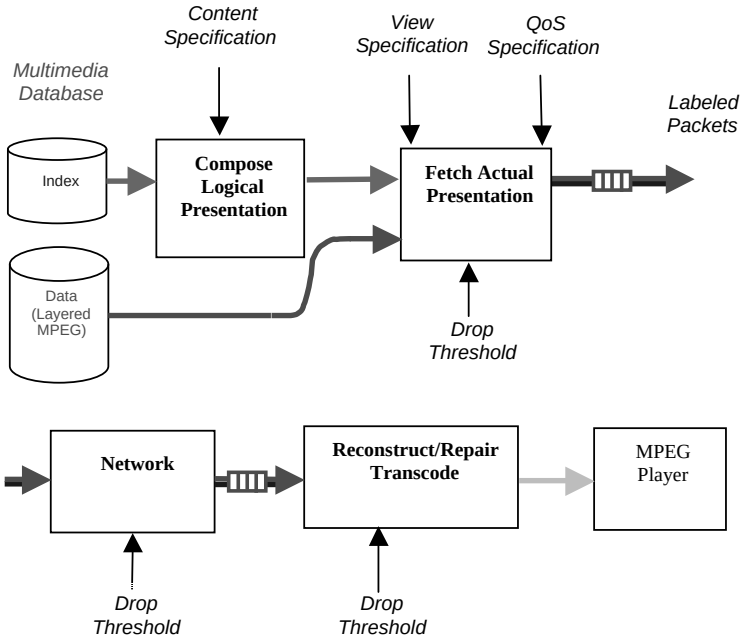


Figure 6-b: On-line components of QoS adaptation prototype

(see Figure 6).

The basic components of the implementation are a remote storage server, a network, and a multimedia presentation client. The remote storage server supports preparation of a layered video encoding format suitable for network transport; the video format is derived from MPEG-1.

The video preparation process is divided into off-line and on-line components (see Figures 6-a and 6-b). Network filters perform prioritized data dropping to scale resource consumption. The client side includes components to reconstruct and repair data as necessary to form a standard MPEG video stream. The user is able to specify Quality of Service requirements via the client, which are used to control adaptation decisions throughout the system.

3.2 QoS Specification

The system allows a user to specify QoS adaptation requirements via a micro-language. The language provides constructs for expressing utility functions in the controllable quality dimensions of the multimedia system. The language also provides a construct to specify weighted combination of

```
value temporal_utility =
   let frame_rate_low =   5.0 in
   let frame_rate_high = 30.0 in
   let frames_to_lost_temporal_qos = fun fr -> 1.0 / (0.5 * fr) in
   let max_lost_temporal_qos =
                     frames_to_lost_temporal_qos(frame_rate_low) in
   let min_lost_temporal_qos =
                     frames_to_lost_temporal_qos(frame_rate_high) in
   let range = max_lost_temporal_qos - min_lost_temporal_qos in
   let user_utility = fun lq ->
                let offset = lq - min_lost_temporal_qos in
                let lq_norm = offset / range in
                let util = 1.0 - lq_norm in
                util * util
   in
                {low_thresh=min_lost_temporal_qos;
                high_thresh=max_lost_temporal_qos;
                utility_fn=user_utility}

value temporal_weight=1.0;;
value spatial_utility= ...
value spatial_weight=0.5
```

Figure 7: An example quality specification written in our QoS microlanguage

utility functions.

Utility functions express a mapping between lost quality levels and a normalized scale of user utility. A general utility function for a single quality dimension was shown earlier in Figure 4. The two thresholds indicate the points where quality becomes either excessive or inadequate, and we are primarily concerned with the shape of the function within the range defined

via these thresholds. The function's shape guides the adaptation process in the multimedia system.

Figure 7 gives an example of a quality specification. The specification is

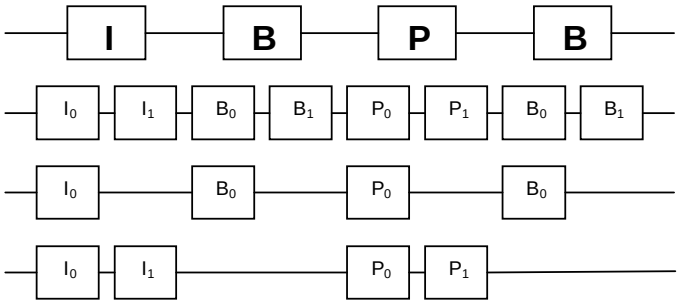


Figure 8: MPEG-based layered video encoding

in the form of a micro-program which, in practice, can be produced either directly by the user, perhaps via a graphical editor, or selected from a list of prespecified defaults.

The language has the declarative style of a functional programming language, although, this style is not a hard requirement of our approach.

We model utility as it relates to quality-loss. For sampling quality dimensions we model quality-loss by 1/f, where f=frequency response. Note that this value tends to zero for perfect temporal resolution and to infinity for no frequency response. Since multimedia video involves displaying a sequence of still frames, the temporal signals are represented by discrete subsampling of the true signal. The Nyquist theorem states that maximum frequency response under discrete sampling is half the sampling rate (frame rate). In other words, lost quality is proportional to half of the sampling interval, hence the substitution of 0.5 * frame rate in the 1/f lost quality formula in Figure 7. The components of spatial resolution could be handled in a similar manner, but in our system we use peak signal to noise ratio (PSNR)[1] as the quality dimension for lost spatial quality.

3.3 Resource Management

The system delivers video as a stream of data packets, and adapts quality by selectively dropping packets. To enable this approach, the layered video encoding separates into distinct packets, data corresponding to the quality dimensions in which adaptation is possible. Figure 8 illustrates the selective data dropping approach. The top line depicts a sequence of four packets, each

[1]PSNR is a popular image error measurement based on the sum of the squared differences between corresponding pixels of two images.

corresponding to one MPEG picture. The second line represents a transcoding of the MPEG format where each picture has been divided into two components, denoted by the subscripts. If both level zero and level one components are decoded, the result is the same as the original MPEG picture. If level one is dropped then the result exhibits degraded PSNR compared to the original. The third and fourth lines depict different adaptation policies. In the third line, PSNR is reduced while temporal resolution is maintained. The fourth line depicts preservation of PSNR over temporal resolution.

User-provided QoS specifications are translated into a priority labeling scheme for packets in the video stream. This association of priorities with packets fixes the order in which packets will be dropped should adaptation become necessary due to resource shortages. At run time, the system adjusts the resource requirements of video delivery to match the available resources simply by adjusting the priority threshold below which packets are dropped.

In a reservation-based approach, the priority threshold would be fixed at the level corresponding to the user's maximum quality loss threshold; and admission control would ensure that adequate resources are available to process all packets with priority higher than the threshold.

3.4 Priority Labeling

The packet priority labeling algorithm uses utility functions to compute the cumulative utility loss of dropping each component of the layered video encoding. In our current encoding, these components are packets associated with I, B, and P pictures[2], at four PSNR levels. The computed loss is cumulative in that it accounts for the loss in the component in question and its dependencies. Dependencies are either hard, as they are implied by the structure of MPEG, or soft, as they reflect good policies for adaptation. An example of a hard dependency is that dropping an I picture implies that certain P and B frames may be dropped too. An example of a soft dependency is that dropped frames be spaced as uniformly as possible.

Hard and soft dependencies are used to define an ordering on packets within each quality dimension, which may be total or partial depending on the quality dimension in question. We then compute, for each packet and each dimension, the cumulative quality-loss when the packets and all lower-order packets in that dimension are dropped. The utility functions in the user's quality specification provide quality dimension-specific mappings from cumulative quality-loss to cumulative lost dimensional utility. The final prioritization of packets in the stream, based on lost overall utility, is then derived as follows:

---

[2]The MPEG video compression standard defines three frame types: Intra-coded (I), predictive-coded (P), and bidirectionally predictive-coded (B).

1. If in all quality dimensions the cumulative lost dimensional utility is zero, assign minimum priority.
2. If in any quality dimension the cumulative lost dimensional utility is one, assign maximum priority.
3. Otherwise, scale the weighted combination of the cumulative lost dimensional utilities into a priority in the range [minimum + 1, maximum- 1].

Minimum priority is reserved for packets that should never pass, because the cumulative lost dimensional utility of the packet in all quality dimensions does not cause quality to drop below the q_{min} threshold of any of the utility functions in the users QoS specification. Similarly, the maximum priority is reserved for packets that should always pass since in at least one of the quality dimensions, to drop the packet would cause quality to drop below the q_{max} threshold.

4. Related Work

Sabata, et al.[13] define QoS in terms of Timeliness, Precision, and Accuracy. Roughly speaking, timeliness relates to the responsiveness of the system - how much time expires between the receipt by the system of a request, and its production of a result. Precision refers to the number of bits used to represent the result. Accuracy refers to the distance between an infinitely precise result and the real - world value it is intended to represent. These primitive QoS concepts are similar to the error classes defined in our model. Precision and accuracy error components can be applied within the quality dimensions of our model. For example, the number of frames used to represent a video presentation could be viewed as defining the precision error component of presentation states in the frame rate quality dimension. Similarly, the number of pixels per frame can be viewed as defining the precision error component of presentation states in the x and y spatial quality dimensions, and the number of bits per pixel can be viewed as determining the precision error component of presentation states in the color-depth quality dimension.

Sabata et al's accuracy component is related to our capture error class, since the capture process for obtaining video content not only imposes limits on the precision error component, but also on the accuracy due to the characteristics of the capture device.

Our utility functions are similar to Sabata et al's benefit functions. They define utility as a function of lost quality for each dimension of the result. We don't distinguish between quality lost due to imprecision or inaccuracy, since errors due to lack of precision are indistinguishable from errors due to inaccuracy as far as the user is concerned. In other words, when declaring their quality requirements, users care simply about the level of error in the result, not the source of the error. Users are capable of distinguishing among

errors in different quality dimensions however. For example, they typically know the difference between low frame rate and low spatial resolution. In the lower levels of the system, however, it is useful to model sources of error. Stored data has precision and accuracy error components that define its *latent quality*. Information about these error components could be stored as meta data alongside the data itself.

Our quality adaptation machinery (the data-dropping virtual machine) introduces further error into the presentation by dropping data in various quality dimensions. The exact effect of this dropping depends on the policy for labeling/prioritizing components of the data stream, but tends to introduce precision errors, i.e., a frame dropper reduces the temporal resolution of the stream and hence is affecting the precision error component of the temporal dimension. Data droppers that drop color information are similarly affecting the precision error component in the color dimension.

Its not clear whether Sabata's notion of timeliness fits as an error component in our model. The accuracy and precision error components seem to refer to the latent quality of stored data, and could be stored as meta data. The timeliness notion seems more applicable to the apparent quality of the retrieved data. This is purely a viewing concern, not an authoring concern, and in our case this is analogous to the specification of a utility function for a new delivery quality dimension.

QoS specification, at the resource level, has received significant attention in the literature [14,15,16,17]. The token bucket model is often used to describe network traffic flows in terms of average and peak bandwidth and burstiness. The approach can be used to describe both stream characteristics and reservation requirements [14,17]. This approach to QoS specification is at a lower level of abstraction than our QoS model, but can serve as a target to map our QoS specifications into. For example, a set of utility functions that describe thresholds for video frame rate and resolution can be used, together with other video stream meta-data, to generate a token-bucket description of the video stream's resource requirements.

Thimm describes techniques for QoS adaptation in multimedia databases [8,9]. His notion of stream presentation parameter is similar to our quality dimension notion. He describes a method for varying presentation QoS using lookup tables and describes normalization functions, similar to our weighting of utility functions, for combining multiple presentation parameters. Layered above these mechanisms is an embedded QoS control system for managing QoS adaptations globally across concurrent presentations.

Rajkumar et al introduce a resource allocation model for QoS management that includes the concepts of QoS dimensions and utility surfaces [18]. The model described in [18] is somewhat more general than ours in the sense that it relates arbitrary application processes and system resources. However, the definition of content and view specifications makes our model more applicable to multimedia databases. The model described by Rajkumar et al also maps utilities directly to resource allocations, whereas

our model maps utilities to dimension-specific quality measures. As a consequence, our model allows a separation between viewing concerns and system capabilities. Finally, our model introduces the concepts of latent and apparent quality, not present in the model described by Rajkumar et al.

Other researchers have proposed models for QoS contract negotiation in environments with multiple resources [19,20].

5. Conclusion

We have outlined a model for QoS control in multimedia databases. The principle concepts of the model include a separation of content, view and quality specification, the definition of quality as a distance measure in multiple quality dimensions, and the use of utility functions to capture user QoS preferences in each dimension. We demonstrated that the model can cover practically useful adaptation strategies by describing a prototype implementation in which user QoS preferences drive prioritization of an underlying data stream, enabling data to be dropped in the correct order when resources become scarce.

The model supports several degrees of independence that we believe are important in multimedia databases. In particular, it supports independence among authoring concerns, viewing concerns and system capabilities, and allows the quantification of the latent quality of content as well as the degradations in quality that result from retrieving content for viewing or storage.

In the future we plan to explore the implications of more complex content and new object-based video encoding schemes. Our definition of multiple quality dimensions, and our use of a layered stream format to implement them can be viewed as a crude form of complex content already. Our QoS model allows users to identify how important these various components of the presentation are relative to each other. This view also offers a glimpse into the future when even single video streams will have complex structure due to the independent encoding of the various objects contained in the stream. Whether to describe different objects in the video using different quality dimensions, or to define them as different points along a single quality dimension is just one of the open research questions we are interested in addressing in the future.

6. References

[1] "Report on the 5th IFIP International Workshop on Quality of Service (IWQoS'97)", Oguz Angin, Andrew Campbell, Lai-Tee Cheok, Raymond R-F Liao, Koon-Seng Lim, Klara Nahrstedt, ACM SIGCOMM Computer Communication Review, July 1997.

[2] "Adaptive Methods for Distributed Video Presentation", Crispin Cowan, Shanwei Cen, Jonathan Walpole, and Calton Pu, Computing Surveys Symposium on Multimedia, December 1995, Volume 27, Number 4, pages 580-583.

[3] "Design of a Multimedia Player with Advanced QoS Control", Rainer Kos ter, MS thesis, OGI, January 1997.

[4] "Meeting arbitrary QoS constraints using dynamic rate shaping of coded digital video," Eleftheriadis, A., and Anastassiou, D., In NOSSDAV 95, Lecture Notes in Computer Science, Springer-Verlag, vol. 1018 pp. 95-- 106, April 1995.

[5] "Resource Management in Networked Multimedia Sys tems," Klara Nahrstedt and Ralf Steinmetz, Computer, IEEE, pages 52-63, May 1995.

[6] "System Support for Mobile Multimedia Applications", Jon Inouye, Shanwei Cen, Calton Pu, and Jonathan Walpole, Proceedings of the 7th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 97), St. Louis, Missouri, May 1997.

[7] "Flow and Congestion Control for Internet Streaming Applications," Shanwei Cen, Calton Pu, Jonathan Wal pole, Proceedings Multimedia Computing and Network ing 1998 (MMCN98), pages 220-264, San Jose, California, January 24-28, 1998.

[8] "Optimal Quality of Service under Dynamic Resource Constraints in Distributed Multimedia Database Systems," Heiko Thimm, Ph.D. Thesis, Universitat Darmstadt, June 1998.

[9] "Managing Adaptive Presentation Executions in Distributed Multimedia Database Systems", Heiko Thimm, Wolfgang Klas, Jonathan Walpole, Calton Pu, and Crispin Cowan. Proceedings IEEE International Workshop on Multimedia Database Management Systems (IW-MM- DBMS'96), Blue Mountain Lake, NY, IEEE Computer Society Press, pp. 152-159, August 1996.

[10] "Device and Data Independence for Multimedia Presentations", Richard Staehli, Jonathan Walpole and David Maier, Computing Surveys Symposium on Multimedia, December 1995, Volume 27, Number 4, pages 640-643.

[11] "Quality of Service Specification for Multimedia Presen tations", Richard Staehli, Jonathan Walpole and David Maier, Multimedia Systems, November, 1995, volume 3, number 5/6.

[12] "Quality of Service Specification for Resource Management in Multimedia Systems", Richard Staehli, Ph.D. thesis, OGI, January 1996.

[13] "Taxonomy for QoS Specifications," B. Sabata, S. Chatterjee, M. Davis, J. Sydir, T. Lawrence, In the Proceedings of the IEEE Computer Society 3rd International Work shop on Object-oriented Real-time Dependable Systems (WORDS '97), Newport Beach, CA, Feb 1997.

[14] "RSVP: A new resource reservation protocol," Zhang , L., Deering, S., Estrin, D., Shenker, S., and Zappala, D., IEEE Network 7(5), pp. 8--18, September 1993.

[15] "Integrated Services in the Internet Architecture: an Overview," R. Braden, D. Clark, S. Shenker, ISI, MIT, and Xerox PARC Internet Request for Comments 1633, June 1994.

[16] "Supporting Real -Time Applications in an Integrated Ser vices Packet Network: Architecture and Mechanisms", Clark, D., Shenker, S., and L. Zhang, Proc. SIGCOMM '92, Baltimore, MD, August 1992.

[17] "Specification of Guaranteed Quality of Service," S. Shenker, C. Partridge, R. Guerin, Xerox/BBN/IBM, IETF Integrated Services WG, INTERNET-DRAFT, 13 August 1996.

[18] "A Resource Allocation Model for QoS Management" Raj Rajkumar, Chen Lee, John Lehoczky and Dan Siewiorek, In Proc eedings of the IEEE Real -Time Sys tems Symposium, December 1997.

[19] "Architectural Support for Quality of Service for CORBA Objects," Zinky JA, Bakken DE, Schantz R., Theory and Practice of Object Systems, Jan 1997.

[20] "The QoS Broker," Klara Nahrstedt, Jonathan M. Smith,IEEE Multimedia, Spring 1995, Vol.2, No.1, pp. 53-67.