Revision of QoS Guarantees at the Application/Network Interface

MS-CIS-93-34
DISTRIBUTED SYSTEMS LAB 31

Klara Nahrstedt
Jonathan M. Smith
Abstract

Connection management based on Quality of Service (QoS) offers opportunities for better resource allocation in networks providing service classes. "Negotiation" describes the process of cooperatively configuring application and network resources for an application's use. Complex and long-running applications can reduce the inefficiencies of static allocations by splitting resource use into "eras" bounded by renegotiation of QoS parameters. Renegotiation can be driven by either the application or the network in order to best match application and network dynamics. A key element in this process is a translation between differing perspectives on QoS maintained by applications and network service provision. We model translation with an entity called a "broker".

1 Introduction

Much of the engineering of networks has been devoted to optimizing the network behavior under the traffic assumptions. New traffic, or new assumptions about the nature of traffic, can cause significant changes in the goals towards which we design and implement networks. Analysis of traffic is dependent on a model of distributed applications behavior. At this time, much traffic modeling for Broadband Integrated Service Data Network (B-ISDN) is speculative, as many of the applications are not yet operating.

In this paper, we propose a process through which better information exchange between applications and networks can take places. We call this process "negotiation", and provide an architecture which embeds negotiation at the call/connection boundary in the B-ISDN management hierarchy. To resolve differences in application and network perspectives on Quality of Service, we introduce a "broker" which translates information in both directions between application and network.

These ideas make the most sense in the context of complex, long-running distributed applications. We are exploring one such application, teleoperation, in order to refine our thinking. Teleoperation is the performance of work at a distance. For teleoperation there are a number of communication channels each of which has stringent requirements. But because of the dynamic change of physical information sensed at the end-points of the network, the requirements change over the lifetime of the application.

*Research support for this work came from Bellcore (through Project DAWN), and from the Corporation for National Research Initiatives (CNRI), which is funded by the National Science Foundation and the Defense Advanced Research Projects Agency under cooperative agreement # NCR-8919038.
1.1 Application/Network Interface

As new network capabilities and proposed applications appear, the boundary between the applications and network services has become a ripe area for exploration. A key research question is the nature of the application/network interface. Applications must react to network changes, and networks must manage complex application requirements. Perspective of requirements (from applications) and guarantees (from the network) are different. Our view is that it is a classic computer science problem, where two "languages" must be translated before any action can take places. This translation process differs from traditional compilers and interpreters in that the translation is bidirectional. While the application/network interface should be part of the architecture, current connection management models do not reflect this well.

Using the management hierarchy in [4], we examine the interface between call management and connection management (Figure 1.). The goal of our architecture is to provide a framework for the application with specified requirements (media the users want to use, relations between the media, quality of the media, etc.) between two remote application users. In addition one would like to allow the users to modify dynamically these requirements over the lifetime of the application. We describe the application user-user connection as a call. A call employs one or more transport connections as the connection support from network might differ for different media.

We differentiate between parameterized connections, where the requirements on the network connection are specified by the application and guaranteed by the network, and non-parameterized connections, where no specification of traffic behavior over the link is given to the network (e.g. UDP/IP, TCP/IP). The requirements and guarantees at the boundary are specified as "Quality of Service" (QoS) parameters. As we observed earlier, these parameters are different for applications and networks. At this point we want to emphasize that we are not concerned about details of a parameterization, i.e. exactly what parameters and what values should be used.

We observe that complex and long-lived applications can be divided into "eras" in which service requirements are constant. Eras are arbitrary, but divisions are convenient when a change in requirements/guarantees could benefit the application, network, or both. These changes are reflected at the application/network interface through a process of negotiation and renegotiation of QoS parameters. The architecture and mechanism we propose can be matched with any network architecture, and its associated service specification, e.g., Clark, et al. [3], or Lazar, et al. [13], or Ferrari, et al. [6], [11], [10] or Tokuda, et al. [5], or others.

Negotiation is done during call establishment and it begins an era in which the negotiated parameters for the call are guaranteed. Renegotiation is done when either application requirements
or network capabilities change. Renegotiation specifies a new era in the call lifetime as the quality of the call is modified.

1.2 Outline of the Paper

The motivation for negotiation and renegotiation is given in Section 2. The negotiation and renegotiation employ a QoS “matcher” (for translation) between the application and network QoS parameters, which is implemented by the QoS broker. We describe the process of “matching” and give an example parameterization in Section 3. Negotiation and renegotiation based on QoS are explored in a telerobotics application. This application has strict requirements on the communication during an era, but changes requirements over time. The telerobotics system we are implementing, as well as expected results are presented in Section 4. The conclusion in Section 5 summarizes what we know now and what remains to be understood.

2 Motivation

“Tempora mutantur nos et mutamur in illis”
(The times are changed, and we are changed with them)

New I/O devices and software technology developments in workstations can support complex networked applications with long lifetimes, such as multimedia communication systems. Over longer periods, changes in requirements become more significant. In faster networks, we observe relatively more dynamic changes over time due to a proportional scaling of the magnitude of randomness (e.g., congestion, delays), which can result in changed guarantees for the application users.

The question is how to communicate the dynamics between application and network. The goal is to provide services which would adapt to changes without closing down the communication between the application users.

2.1 Dealing with Dynamics

There are essentially two ways in which varying demands can be accommodated.

First, the application can completely specify its demands when the call set up is being carried out (“the static approach”). The demands can be specified either as deterministic bounds or bounds in the form of a range (minimum and maximum values) as specified in Tokuda et al. [5]. In the case of assigning a range of parameters, the network can dynamically adjust resource allocations within the range. The network can make an “admission” decision based on demands and if the call is accepted, resources are allocated by the network.

Second, the application can dynamically specify requirements to the network, as well as react to changes in network capabilities (“the dynamic approach”). This reaction to dynamics takes the form of negotiation and renegotiation between the network and the application and between the application entities.

The advantage of the first approach is the simplicity (and certainty) of the resource allocation and service provision model. While the call admission criteria may be complex internal to the network, the simple yes/no model for the connection configuration is attractive. Better admission decisions can be made as the level of detail with which the application specifies its behavior at setup time increases - this provides more information to the call admission process. Unfortunately, complex applications must often specify their aggregate behaviors, and this typically takes the form of statistical specifications such as average and peak bandwidths. These aggregates have the
difficulty that time-varying resource demands are hard to specify, and thus overallocation often results.

The advantage of the second approach comes from recognition that time-varying demands are a fact of life for complex applications, and as well that there are significant dynamics in network resource availability. It is our belief that these dynamics can be exploited in resource allocation decisions and lead to better performance. A key issue is the tradeoffs possible between the simple "yes/no" model desirable for applications and exploiting the dynamics of a long-running application.

2.2 Performance Potential

Figure 2 illustrates a scenario for resource allocation behavior over time. The curve labeled $f_1$ represents the maximum resource allocations the network is able to accommodate. The curve labeled $f_2$ represents the minimum acceptable resource allocation with which the application can operate. The crucial observation is that both of these vary with time, and this variation can be exploited. Consider the optimum system behavior, where, for all $t$, $f_1(t) \geq f_2(t)$ (correctness condition). Now, from the network’s perspective, it can make optimal allocation decisions when $f_1 = f_2$. From the application perspective, it can operate where $f_1 \geq f_2$. If we look at the system behavior over some time interval $[0,t_A]$, aggregate throughput for the system will be optimized when

$$\text{Network Perspective} = \int_0^{t_A} (f_1(t) - f_2(t)) \, dt$$

is minimized. Any static bounds must specify $\min_{[0,t_A]}(f_2(t))$ as a lower bound. While values less then $\max_{[0,t_A]}(f_1(t))$ may be specified as an upper bound to increase the probability of cell
admission, this value is greater then \( \min_{[0,t_A]}(f_1(t)) \). Static bounding, then results in a resource waste proportional to (dashed rectangle in Figure 2):

\[
\int_0^{t_A} (f_1(t) - f_2(t)) dt \approx (\max_{[0,t_A]}(f_1(t)) - \min_{[0,t_A]}(f_2(t))) * t_A
\]

Consider now the "boxes" outlined in Figure 2. In our scheme, the interval over which the k-th box is defined is \( t_{k+1} - t_k \). Each adjustment in a box's height represents the result of a renegotiation. The unused bandwidth is approximated using these boxes to calculate the definite integral by numerical integration

\[
\int_0^{t_A} (f_1(t) - f_2(t)) dt \approx \sum_{k=0}^{A} (t_{k+1} - t_k) * (f_1(t_k) - f_2(t_k))
\]

In our approach the approximation of the dynamic changes may be better than in [5] because it matches more closely the network perspective of the QoS parameter changes (Figure 2.) and therefore unused resources can be provided to other connections (users). The dynamic approach provides the advantage of flexible information to the network which can result in better dynamic resource allocation. The disadvantage is the overhead of renegotiation. However, this overhead can be limited by enforcing minimum era size.

### 2.3 Negotiation and Renegotiation based on QoS

The first questions in discussing negotiation are who the parties are, and how the parties negotiate. There are really two parties to any QoS negotiation in networked multimedia applications - other application elements and the network infrastructure, as Figure 1 illustrates. There are peer-to-peer negotiations between the application elements and application-to-network negotiations. The peer-to-peer negotiations settle the multimedia requirements between the end-points. The application-to-network negotiations communicate the performance requirements for the multimedia connections between the application and the network. This split between the types of negotiation is detailed in Figure 3.

This conceptual split between types of negotiation reflects the observation that applications and network elements may have different perspectives on what Quality of Service means.
Application QoS is "quality" in terms meaningful to application services, i.e., how well the application can present data to satisfy the expectations of end users. Specification is in terms of application characteristics. The application characteristics parameters include information on the multimedia stream description and the media relations, such as communications topologies and entity roles. The stream description maintains media quality parameters. Some parameters for quantized continuous media include sample rate, sample size, compression algorithms and sample loss rate. A set of relevant parameters are given in the Table 1 and Table 2. Table 1 reflects the high-level description of the system, while Table 2 gives parameters for various media.

<table>
<thead>
<tr>
<th>Application Characteristics</th>
<th>Application Type</th>
<th>Multimedia Conferencing</th>
<th>Remote Seminar (Education)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synchronization of Media</td>
<td>Robots Data, CD-Audio, High Motion Video</td>
<td>High Motion Video, CD-Audio, Text</td>
<td>Telepointing, CD-Audio, Text, Slides, Motion Video</td>
</tr>
<tr>
<td>Conversion of Media</td>
<td>All Data</td>
<td>Video, Audio</td>
<td>All Data</td>
</tr>
<tr>
<td>Integration of Media</td>
<td>Robot &gt; Graphics</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Communication Relation</td>
<td>Textile Data from one or more Robot Hands</td>
<td>Audio, Text</td>
<td>Telepointing, Text, Audio</td>
</tr>
<tr>
<td></td>
<td>Unicast or Multicast</td>
<td>Multicast</td>
<td>Broadcast</td>
</tr>
</tbody>
</table>

Table 1: Application Characteristics for different Application Types

<table>
<thead>
<tr>
<th>Media Type (Quality)</th>
<th>Robotics Siney Data</th>
<th>Satellite Image</th>
<th>Slow Motion Video Quality</th>
<th>Telephone Audio Quality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample Size (x2)</td>
<td>64 Bytes</td>
<td>640x480 pixels</td>
<td>640x480 pixels</td>
<td>160 Bytes</td>
</tr>
<tr>
<td>Sample Rate (x2)</td>
<td>500 samples/sec</td>
<td></td>
<td>5 samples/sec</td>
<td>50 samples/sec</td>
</tr>
<tr>
<td>Compression joint</td>
<td>JPEG (5:1)</td>
<td>MPEG (5:1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>End-to-End Delay Time (s)</td>
<td>10 ms</td>
<td>1 sec</td>
<td>100 ms</td>
<td></td>
</tr>
<tr>
<td>Sample Loss Rate (x2)</td>
<td>max 1 sample/min no loss of 2 consecutive samples</td>
<td>0</td>
<td>1 sample/min</td>
<td>1 sample/min</td>
</tr>
<tr>
<td>Cost</td>
<td>$ xx/min</td>
<td>$yy/image</td>
<td>$zz/min</td>
<td>$ww/min</td>
</tr>
<tr>
<td>Input Device</td>
<td>Robot Hand</td>
<td>Image File</td>
<td>Camera</td>
<td>Microphone</td>
</tr>
<tr>
<td>Output Device</td>
<td>Screen</td>
<td>Screen</td>
<td>Screen</td>
<td>Speaker</td>
</tr>
</tbody>
</table>

Table 2: Media Quality for different Media Types

Network QoS is "quality" in terms of network service, such as bandwidth and delay guarantees. The set of parameters depends on the chosen network, its connection establishment protocol, call admission protocol, and real-time services models. An example network QoS parameterization is studied in Section 3.2.

2.3.1 Negotiation/Renegotiation of Application QoS Parameters

Negotiation (renegotiation) establishes an agreement between the parties with respect to the application QoS parameters. Using some connection the application QoS are exchanged. The receiving party checks the incoming multimedia quality and service requirements for feasibility (e.g. re-
sources, service existence, device support). The result is either "accept", or "modify". In the case of "modify" answer, a suggested quality is returned to the sender. The request-sending party has the option: it can either change the quality or leave it unchanged. If unchanged, the receiver side must adjust the incoming media quality to its own quality (i.e. drop the information).

2.3.2 Negotiation/Renegotiation of Network QoS Parameters

Negotiation/renegotiation establishes agreement between connection management and network management on network QoS parameters.

The focus of our work is the interface not the network architecture. We can draw on an extensive body of work on real-time transport protocols, and architectures [3] for achieving real-time goals such as low jitter [7] and low delay [10], [11]. Several architectural techniques for service of this traffic class are discussed by Lazar, et al. [13], [9].

Negotiation/renegotiation of network QoS happens on a per-connection basis. The connections are unidirectional connections. We assume the network management uses a distributed admission policy. Thus, the connection set up is tied to negotiation of QoS parameters. The admission protocol performs actions to guarantee them (admission-reservation, admission-allocation). A general connection set up protocol is shown in Figure 4. The result of the QoS negotiation during the connection set up is either "accept" and successful connection set up, or "reject" with the possible QoS parameters are reported to the initiator of the connection. This gives the end user an opportunity to dynamically adjust the media quality.

Renegotiation can take two forms. First, when the existing connection quality is to be altered, renegotiation is carried on as a background activity by the network management entity, while data transmission remains the foreground task. The background nature of the task limits its impact on data traffic. For "accept", the resources are allocated and the new era begins. For "reject" the previous quality is maintained. Second, if a new connection must be established, the renegotiation with the network starts in order to accommodate new request.

For removing a connection, no renegotiation is necessary. A connection release request is sufficient.

![Figure 4: Connection Set Up Protocol](image-url)
3 Brokerage

The split perspectives on QoS (brought out in the previous section) require translation between the two "languages" used by the application and network to characterize quality.

Translation is implemented by a QoS broker. The QoS broker is invoked after negotiation/renegotiation of application QoS parameters, and after negotiation/renegotiation of network QoS results in a "reject" result (see Figure 3). QoS broker is also invoked when the network/application management signals changes in quality of guarantees/requirements (see Figure 5).

![Diagram of QoS Brokerage](image.png)

Figure 5: Signaling of QoS Change

3.1 Model

Invocation of the QoS broker from an application begins with an analysis of media interrelations. If the media are integrated [8] (as when multiplexing two or more media streams in the application subsystem) the broker has to determine media requirements for the aggregate on the requested connection. Translation between media quality and network quality parameters must then be performed for each connection. A breakdown of the subtasks involved is shown in Figure 6.

The process of translation and integration/disintegration is bidirectional. In translation from application QoS parameters to network QoS parameters, the mapping is from media quality for the particular connection to network QoS parameters. The invocation of the QoS broker from the network side translates network QoS parameters into media quality parameters for a connection. If the media quality parameters are for an aggregate, the translator must decompose the parametrization appropriately.

3.2 Parameters and Classes

QoS parameters from a fixed parameter space map into equivalence classes. In networks, these classes are traffic classes. Application requirements are mapped into application classes. The class concept is useful for control and scheduling during data transmission when guarantees are required. Translation can occur between application classes and traffic classes. Classes can be compared to language constructs, which applications and networks can use for their own computation (e.g.
The QoS parameters can be compared to language elements. We will illustrate possible class translations with an example.

Consider the media classification categorized by delay and loss in Table 3. Each class can be further refined with other parameters such as sample size.

At the network side, we can use the traffic classes introduced in [13]. Then $\text{Class}^I \sim \text{Class}00$, $\text{Class}^{II} \sim \text{Class}01$, and $\text{Class}^{III} \sim \text{Class}10$, and $\text{Class}11$.

Classifications in the application subsystem translate into both classification of the traffic at the switches (network, data link layer), and the functionality of the transport layer. In the transport layer, classes 10 and 11 would require different loss-recovery strategies (in class 10 we may need retransmission, but in 11 we may not, as Turner and Peterson [2] argue).

Translation between the equivalence classes appropriate for the application and the service

---

**Figure 6: Model of QoS Broker**

---
classes used by the network has the potential problem of information loss. This is a consequence of the many-to-one nature of the mappings from applications characteristics to equivalence classes; such mappings lose information about the application, as when an application’s memory referencing is viewed as a page fault rate. We believe that application/network communication through detailed translation (using QoS parameters) can do better.

### 3.3 Translation and Integration/Disintegration

For this process to operate, we must fix the parameter space of both application and network. We listed application QoS parameters in Table 1 and Table 2. For the network QoS parameter set, we use the parameters of the Tenet protocol suite [6], [11], [10]. The QoS parameters are:

- “throughput pledge” (minimal interarrival time for the cells $x_{\text{min}}$, minimal value of the average cell interarrival time $x_{\text{ave}}$ and time interval $I$ over which these values have been computed),
- performance requirements (end-to-end transmission delay $D_{\text{max}}$ and the probabilistic bound $W_{\text{max}}$ on the losses of cells in the network).

Using these application/network parameters enabled translation in one direction, from media quality parameters to network quality parameters, is shown in Figure 7. The equations are used to convert between parameters; parameters are abbreviated as indicated by parentheses.

#### Translation

<table>
<thead>
<tr>
<th>Media Quality Parameters</th>
<th>Network Quality Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample Size (sz)</td>
<td>Throughput Pledge</td>
</tr>
<tr>
<td></td>
<td>- minimal interarrival time</td>
</tr>
<tr>
<td></td>
<td>$x_{\text{min}}$</td>
</tr>
<tr>
<td></td>
<td>- average packet interarrival</td>
</tr>
<tr>
<td></td>
<td>$x_{\text{ave}}$</td>
</tr>
<tr>
<td></td>
<td>- time interval $I$</td>
</tr>
<tr>
<td>Sample Rate (sr)</td>
<td>Performance Requirements</td>
</tr>
<tr>
<td></td>
<td>- end-to-end transmission</td>
</tr>
<tr>
<td></td>
<td>delay $D_{\text{max}}$</td>
</tr>
<tr>
<td>End-to-End Delay (t)</td>
<td>- probabilistic bound on</td>
</tr>
<tr>
<td></td>
<td>the losses of cells in the</td>
</tr>
<tr>
<td></td>
<td>ATM network $W_{\text{max}}$</td>
</tr>
<tr>
<td>Sample Loss Rate (ls)</td>
<td></td>
</tr>
<tr>
<td>Compression Ratio (cr)</td>
<td></td>
</tr>
</tbody>
</table>

$W_{\text{max}} = \frac{ls}{sr}$

$D_{\text{max}} = t - 2\times \text{service time in upper layers}$

Figure 7: Translation from Media Quality Parameters to Network Parameters

The translation from network QoS parameters to media quality parameters is performed if change of end-to-end delay for the connection, and/or change of interarrival time, and/or change of loss probability are reported from the network. In this case, the equations shown in Figure 7 are inverted as appropriate.

When the application subsystem wants to multiplex two or more media streams itself, the media quality of the composite is more complex. The QoS broker has to calculate the resulting media quality parameters before translation is complete. Figure 8 shows integration of two media quality parameter sets. Two kinds of equations are used for each parameter computation, so that
we can multiplex both homogeneous (upper equations), and heterogeneous (lower equations) media streams.

As translation is bidirectional it follows that if integration was performed, to reverse the mapping, disintegration is required. Disintegration requires demultiplexing and translation dependent on the particular changes signaled by the network.

We elaborate on integration/disintegration and the translation process in the telerobotics scenario in Section 4.1.

4 Application to Telerobotics

We are exploring the negotiation/renegotiation architecture in the context of an actual application, that of telerobotics/ distributed digital teleoperation [1]. Our test system configuration is shown in Figure 9.

Teleoperation allows an operator to exert forces or to impart motion to a slave manipulator. The operator can also experience the forces and resulting motion of the slave manipulator, known as “kinesthetic feedback”. An operator is also provided with visual feedback, and possibly audio feedback as well.

Visual information requires at least megabit bandwidth with frame rates in excess of ten frames per second. Normally, teleoperation makes use of two to three video channels. The kinesthetic communications channel is required in both directions for each manipulator. There are normally two
manipulators. Kinesthetic channels require transmission of some hundreds of bits at the kilohertz rate. There are strict timing requirements on manipulator channels (robotics data) and irregular, or missing data can result in physical damage. Along with these channels might be channels for audio, and video information.

4.1 Example Scenario

We assume the robot subsystem is equipped with two robot hands, a video camera and a microphone. The application specifies that tactile data, video and audio must be synchronized. The synchronization will be achieved through integration of tactile data, video and audio data [8], where data collected together travel together in a bundle through the communication system. We use the media quality values, specified in Table 2. The QoS broker analyzes the media relations, which results in two integration steps and two translation steps and further negotiation for two connections.

After integration of two handed robot sensory data using the upper equations in Figure 8 we get the values shown in the tactile data row of Table 4. The integration of audio and video data gives, using the lower equations in Figure 8, the values shown in Audio/Video row of Table 4. Translation from robot sensory media parameters to network parameters calculates (Figure 7) interarrival time,
Table 4: Values after Integration of Media Qualities

<table>
<thead>
<tr>
<th>Type</th>
<th>Sample Size</th>
<th>Sample Rate</th>
<th>Response Time</th>
<th>Sample Loss Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tactile Data</td>
<td>128 bytes</td>
<td>500 samples/s</td>
<td>10 ms</td>
<td>1 sample/min</td>
</tr>
<tr>
<td>Audio/Video</td>
<td>[150, 6304] bytes</td>
<td>50 samples/s</td>
<td>100 ms</td>
<td>(1, 38.4) audio samples/min</td>
</tr>
</tbody>
</table>

Time interval, end-to-end delay and probabilistic loss. The deterministic bounds are shown in the tactile data row of Table 5. Translation from audio/video media parameters to network parameters results in values shown in Audio/Video row of Table 5. The interarrival and sample loss rate are specified in range form: [min value, max value].

Table 5: Values after Translation from Media Quality to Network Quality

<table>
<thead>
<tr>
<th>Type</th>
<th>Interarrival</th>
<th>Time Interval</th>
<th>End-to-End Delay (service time in upper layer 1 ms)</th>
<th>ATM Cell Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tactile Data</td>
<td>0.75 ms</td>
<td>2 ms</td>
<td>8 ms</td>
<td>0.0000333</td>
</tr>
<tr>
<td>Audio/Video</td>
<td>[0.152 ms, 6 ms]</td>
<td>20 ms</td>
<td>98 ms</td>
<td>[0.00033, 0.0128]</td>
</tr>
</tbody>
</table>

In order to demonstrate reverse translation we assume negotiation for the audio/video connection wasn’t successful and the lower bound on interarrival cell time (0.152 ms) can’t be guaranteed. Suggested lower bound is 1 ms.

Translation from the network parameter to media parameters results in the upper bound of sample size going down to 960 bytes from 6304 bytes (sample rate unchanged). This means we must smooth the traffic (fragmentation of the video samples), which may affect end-to-end delay due to increased service time in upper layers, or we must lower the resolution of the video frame. The resulting value of sample size after reverse translation and disintegration from the resulting Audio/Video sample size to audio and video sample sizes are shown in Table 6.

Table 6: Values after Reverse Translation and Disintegration

<table>
<thead>
<tr>
<th>Changes</th>
<th>Audio/Video Sample Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interarrival</td>
<td>960 bytes</td>
</tr>
</tbody>
</table>

4.2 Expected Results and Implementation Status

This application provides dynamic changes over its execution because the physical information changes as the robot hands are moving. The changes of the physical information may result in renegotiation of requirements among the remote sites as well as changes in network guarantees. We will test the “era” concept. The important evaluation parameters will be the time required to change eras and reasonable era length. These parameters will provide more insight into which approach (static or dynamic) is suitable for this real-time class of applications.
The complex timing requirements of the telerobotics application give us a platform to study parameterized call/connection management and negotiation services. Telerobotics employs distributed control and execution mechanisms which force some real-time requirements on the network. As we implement the lower layer protocols (currently based on Tenet protocol suite), we expect to do performance measurement and evaluation of the underlying ATM network from the application performance point of view.

The communication software and hardware support for video, audio and ATM host interface have been implemented on IBM RISC System/6000 workstations using AIX. To obtain robotics sensory data over the ATM network we are connecting the SUN and RS/6000 stations with a S bus-Microchannel bus interconnection card. The hardware and device drivers on the RS/6000 are functional. The specification and design of the QoS broker as well as negotiation/ renegotiation of application QoS are implemented as part of the telerobotics project. Currently we are working as on the implementation of network guaranteed services using ideas described in [6],[10], [11] and [12], as well as the extended connection and call management, including negotiation/renegotiation of network QoS, as described in section 2 and 3.

5 Conclusion

The main contribution of this work was to flesh out an architecture with which complex, long-lived applications could adapt to variations both in their requirements and in the capability of the network to service their requirements.

- “Eras” are used to describe and discretize variations in quality of service parameters for complex, long-lived applications.
- Negotiation and renegotiation provide a mechanism to signal variation in QoS parameters at the application/network interface. They are invoked at era boundaries, and can aid resource allocation.
- Application requirements and network resource allocations are expressed in fundamentally different terms and languages. A translation process, modeled as a QoS broker, bridges this gap.
- Teleoperation is a complex application with possibly long usage intervals. We are using teleoperation both to gauge the dynamics and traffic characteristics of a real application and to experimentally validate our architecture.

There are many open questions related to QoS and its use in managing applications and networks. For example, it remains unclear how to choose a “good” parameter space, and whether what is “good” for the network is “good” for the application writer. Many different parameterizations and service classifications exist. We must move towards understanding and exploiting their domains of applicability.

6 Acknowledgments

Dave Feldmeier, Ruzena Bajcsy, Dave Farber and Lou Paul have made significant contributions to this work, either through suggestions or constructive criticism, all of which is appreciated.
References


