Quality of Service on a Distributed Virtual Reality System for Robots Tele-operation over the Internet

M. Patricia Martínez-Vargas[1], Maria E. Meda-Campaña[1], Victor M. Larios-Rosillo[1], and Francisco Ruíz-Sánchez[2]

[1] Research Center on Systems and Information Management, Information Systems Department, CUCEA University of Guadalajara, Periferico Nte. 799, Mod. L-308, 45100 Zapopan, Jalisco; Mexico
Phone: +52 (33) 37.70.33.52, Fax: +52 (33) 37.70.33.53,
{mmartinez,emeda,vmlarios}@cucea.udg.mx

[2] Sección Mecatrónica Departamento de Ingeniería Eléctrica CINVESTAV-IPN
A.P 14-740 07300 México D.F., Mexico
Phone: +52 (55) 51 60 37 89, Fax: +52 (55) 51 60 38 66
fruiz@mail.cinvestav.mx

Abstract. Virtual Reality applications for tele-operation are the base for nowadays development of virtual laboratories where researchers can share work at distance. Such systems must support a distributed framework to scale and a good Quality of Service (QoS) support to be effective. This paper proposes such a system, where the aim is the teleoperation of robots for complex cooperation tasks between peers. The system has multimedia components and virtual reality interfaces to control and monitor the real robots state. To ensure the QoS, good politics must be defined to manage the resources of the system to reduce bandwidth consumption over the internet. To apply the QoS policies, an online algorithm is adapted and compared against a greedy version. The online algorithm is taken from a financial set of algorithms used to invest in stock markets. The analogy between stock markets and the QoS control is shown as promising for further applications with QoS support over the internet.

1 Introduction

Multimedia and Virtual Reality applications [8] running over the Internet requires a high demand of connectivity among users communities. These applications are helping to increase the interaction necessary to share work at distance. Therefore, communities can work on complex team tasks for example, to manipulate and to teleoperate physical devices from remote environments with precision and effectiveness. However, one problem to overcome is when using the internet related in how to avoid the possible perturbations of the network produced by latency, jitters and data loss. Hence, the aim is to implement the adequate

T. Böhme et al. (Eds.): IICS 2004, LNCS 3473, pp. 69–79, 2006.

Quality of Service (QoS) management policies, resulting in good communication helping to users to coordinate complex tasks at distance. Moreover, if the users are seen like peers on the internet network, then some works in optimization can be done from the side of each peer to support the QoS.

This paper presents a strategy to dynamically support the QoS of a distributed peer to peer system with the goal to control robots tele-operation over the internet. Such strategy is related and built in from the software level. The organization of the paper is as follows: on section two we give the context of our specific application and the requirements to keep satisfactory conditions for robots tele-operation, on section three we describe the proposed architecture, the communication strategy and the QoS politics to implement; on section four we advance a distributed solution on QoS based in a financial on line-algorithm; on section five, we introduce the obtained results for this work and finally, on section six conclusions are shown as well as the perspectives where we propose future alternatives for QoS development.

2 The Problem Context

Universities and research centers in Mexico are joining efforts to support technologies helping to share resources from physical devices and computers. These resources can be integrated into the concept of Virtual Laboratories, where students and professors can tele-operate interconnected devices through networked virtual reality user interfaces. The University of Guadalajara (UdeG), the Centro de Investigación y de Estudios Avanzados IPN (CINVESTAV) and the Universidad Nacional Autonoma de Mexico (UNAM) are working together, in a collaborative project to support the development of virtual laboratories over the Internet. A first step, was to set up an environment allowing the tele-operation of real robots by means of virtual reality and multimedia applications over the internet.

To drive solutions to increase the coordination and communication among the peers teleoperating the robots, the system is aimed with a virtual reality interface and a video/audio channel on each peer. This configuration, result in an important bandwidth consumption because each peer on the network receives simultaneously the audio/video signals from all other peers and because the virtual reality interface acts like a shared memory where coherence must be kept. Furthermore, the virtual reality interface have one communication line for the robots control and other to feedback force information from robots sensors to be reflected in haptic interfaces. As a case of study, the system intents to control a pair of robots performing a simple writing task on a blackboard (each robot with tree degrees of freedom). While controlling the robots, the main constraint for the communication system is to prevent latency behaviors between peers. The communication bandwidth is used by separate channels related to the robot control data stream, the video/audio inputs and outputs, and the messages for peers management. The control of the system bandwidth consumption is done by deciding the granularity of information packages to send on each channel by means of the QoS online algorithms.

The QoS can be defined as the capacity to provide better network services for application users requirements, providing satisfaction and understandability of all subjective or objective parameter values [5]. The QoS goal is to provide a resource management strategy by priority including policies considering issues like bandwidth, jitter, loss, and latency [4]. This paper is related with the solutions that optimize the use of resources on each peer to deal with the QoS.

One of the main tasks for the virtual laboratory system, is to monitor and to supervise the communication lines over the internet. Thus, the aim is to get a stable communication among peers depending on available information such as bandwidth, latency, jitter, and data loss. To get the best control over these parameters and to ensure the safe tele-operation of robots, is necessary to establish the adequate QoS policies.

Our contribution is to offer an scalable architecture that preserves the QoS and to provide security (by means of stable signals) for the interconnected devices that must be controlled in real time. The system implements the QoS resource control based in policies. Policies are divided in tasks, actions and rules to provide the security and QoS issues.

3 The Architecture of the System

The architecture of the system is shown in figure 1 illustrating collaborative peers tele-operating the robots. Each peer is supported in different communication ports and protocols like the RTP (Real Time Protocol) for video, audio and Virtual Reality updates and like the TCP/IP protocol for the system administration messages. There are different kinds of peers states that can be settled as: *device tele-operation state*, *device master control state* and *observer state*. Tele-operation state is only possible when a haptic device is present to control at distance one robot through the Virtual Reality interface. Video and audio are available to look other peers participating in the tele-operation session. The device master state is the peer where the real robots are interconnected and controlled. This master peer, get control instructions from the tele-operation peers and has a virtual reality interface also with audio-video channels. The video channels are implemented for security to check that the real state of robots is coherent against the virtual reality interface. The robots sensors send force feedback information to the haptic devices. Finally, the observer state belongs to each additional peer that is not going to tele-operate any device and just only wants to observe via the virtual reality interface the cooperation tasks.

The system is OS independent via the Java Virtual Machine with a dependency only from the C++ programs to support the data acquisition cards for sensors and the interconnected haptic devices. Such program interfaces are supported by the Java Native Interface. However, for these devices the most frequent OS systems are linux and windows for the available drivers and libraries in C++ or C, on the external hardware devices used.

Video confering is supported by the java media framework library and an implementaton of the RTP protocol that is used both for video and virtual reality updates. The virtual reality interface is built in the top of the Java 3D library

with support for sterographic devices. An administration module is implemented to manage the security on the system, validate the users and have a general overview of the other peers via the TCP/IP protocol. Finally, there is a QoS module that implements the QoS policies and that is able to connect/disconnect or reduce the bandwith consumption of all the networked peers in order to preserve the QoS. This module can load different kinds of algorithms to improve the performance of the QoS policies decisions to take. In this paper, we are presenting two algorithms, one online and another determinist called greedy.

Next section will shown the policies to apply on the system before start explaining the QoS algortihms implementation.

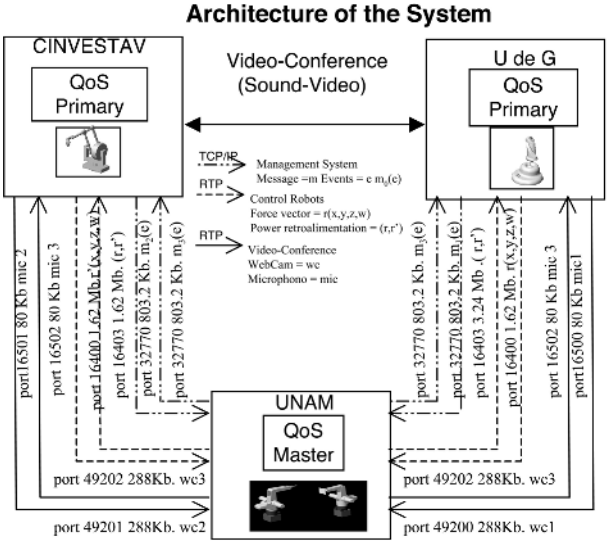

Fig. 1. The VE is partitioned in regions managed by a cluster

3.1 QoS Adaptation Policy

To manipulate efficiently the system resources for the QoS application support; in general terms we have defined the following policies on each pear:

- A robots control stream always has a higher priority to use network bandwidth, CPU and other related resources than a video stream.
- An Audio stream always has higher priority to use network bandwidth, compared with the video stream.
- To avoid the QoS oscillation, the QoS system self-adaptation only performs quality-degrading activity.
- The system may perform quality-upgrading adaptation under receivers requests. The receiver request will be made using the administration channel.

The correct selection of the required QoS policy will depend on the latency of the system and also on the current system state. A system state involves the

state variables of each one of the peer to peer connections of the application. In our case these state variables are Video (V) and Audio (A), the associated values of these states variables are the following:

Normal Video	NV
Degraded Video	DV
Without Video	WV
Normal Audio	NA
Degraded Video	DA
Without Audio	WA

Notice that these states variables describe the performance of each one of the peer to peer connections of the application, for instance we are considering three peer to peer connections: 1) UNAM-CINVESTAV, 2) UNAM-UdeG and 3) UdeG-CINVESTAV, if we are monitoring the UNAM-UdeG connection then it is possible that the latency reaches high values, in this case the state variable V takes the DV value. However, the situation in another peer to peer connection could be different and the state variable V in this connection could take another value.

Given the architecture of the application, the QoS policies will be implemented by peer to peer connection considering its latency in the video-conference channel and its state variables. This is illustrated on figure 2.

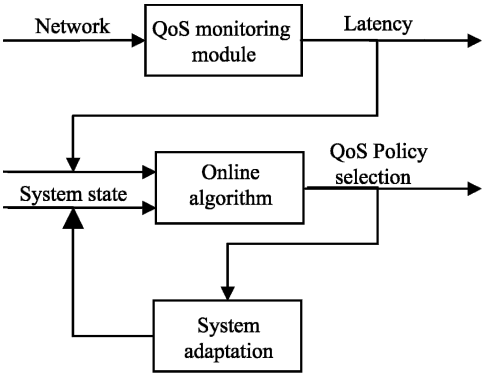

Fig. 2. QoS policies process

Following are described each one of the QoS policies.

a) QoS policies when the latency value is less than 15 ms.

1. NV Continuing in same state;
2. DV Upgrade video under receiver request;
3. WV Open video session under receiver request;
4. NA Continuing in the same state;
5. WA Open audio session under receiver request;

b) QoS policies when the latency value is in the interval $15 \leq L \prec 17ms$.
6. NV Degrading video;
7. DV Continuing in same state;
8. WV Open video session under receiver request in a degrading mode;
9. NA Continuing in the same state;
10. WA Open audio session under receiver request;

c) QoS policies when the latency value is in the interval $17 \leq L \prec 19ms$.
11. NV Close video session;
12. DV Close video session;
13. WV Continuing in the same state;
14. NA Close audio session;
15. WA Continuing in the same state;

d) QoS Policies when the latency value is in the interval $19 \leq L \prec 20ms$ Send warning messages over the administration channel

e) QoS policies when the latency value is greater or equal than 20 ms.
16. NV Close video session;
17. DV Close video session;
18. WV Continuing in the same state;
19. NA Close audio session;
20. WA Continuing in the same state;

The automatas representing the change of values of the states variables video and audio given a latency value are depicted on figures 3 and 4 respectively.

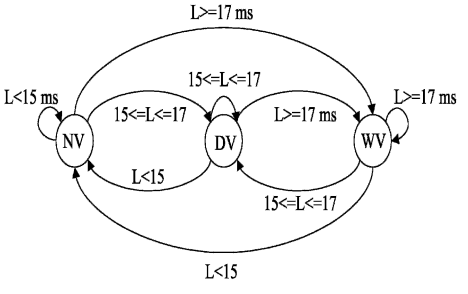

Fig. 3. Automata representing the values taken by state variable video (V) given any latency value

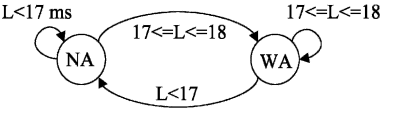

Fig. 4. Automata representing the values taken by state variable audio (A) given any latency value

4 The Distributed Solution in QoS

An important component to control the QoS is the network monitor which provides as a return value the latency between peers. Latency is estimated as the time for one package to travel from one peer to another. To avoid transients on the latency lectures, the monitor also implements statistical functions to get

some form of average latency providing a final value to be used by an online algorithm which is the responsible of taking the optimal decisions to apply the QoS policies.

An online algorithm is defined as one that must produce a sequence of decisions on incoming inputs based only in past with out knowledge about the future [6]. Controlling the QoS of an application is an intrinsically online problem. The study of online algorithms include concepts of scheduling, optimization, data structures, game theory, combinatorial problems and other computational topics. The framework of online algorithms includes the approach of competitive analysis where the quality of an online algorithm is measured against an optimal offline algorithm which is an unrealizable algorithm that has the knowledge of all the future. Competitive analysis falls in the framework of worst case complexity.

One way to analyze the algorithm is to view the online problem as a game between an online player and a malicious adversary. The online player runs the online algorithm over an input created by the adversary. The adversary, based on the knowledge of the algorithm used by the online player tries to put the worst input to make the decision task costly for the online algorithm and to minimize its competitive ratio.

Our original idea is to relate the problem of QoS to an online financial problem where the online player seek to invest its wealth among a number of investment opportunities in the most profitable way [3]. In our problem, the online player look for the network state and must decide which QoS policy to apply based on the online network latency measure. The network can be seen as a market of s securities that can be stocks, bonds, currencies or commodities. Each security has a price that fluctuates with the time and depending on the price (related for us as the online latency measure), the online player must decide to buy or sell a QoS policy in order to get a trading strategy that keeps a wealth in our system or in other words, that keeps the QoS. If the latency start to increase its value, the system must sell its services applying a policy that means to decrease the use of bandwidth in communications. Moreover, if the latency goes down on its value means that the price is good to buy and is related to apply again a QoS policy to connect or increase the use of communication components to increase the QoS to the optimal level. If the algorithm have not an optimal performance on decisions, the system may operate with a negative profit called loss but in the contrary keep a good QoS means a profit.

The adapted algorithm for the QoS control is based on a Money Making online trading strategy from [2]. On such strategy, we must define market sequences by relative prices as $X = x_1, x_2, ..., x_{n-1}$. This market sequence is related in the system to a set of lectures from the network monitor. Fixing $n \geq 2$, assume that each feasible sequence is of length n and impose the restriction that the optimal offline return associated with an online sequence is at least ϕ where $\phi \geq 1$. Is not difficult to see that ϕ can be obtained as $\phi = \prod_{1 \leq i \leq n-1} \max\{1, x_i\}$ Consider the following algorithm wich is defined recursively in terms of (n, ϕ). Use $R_n(\phi)$ to denote the return of the algorithm for a n period sequence. The algorithm is decribed as follows:

Algorithm: if n = 2, invest the entire wealth and otherwise invest b units in QoS policies where

$$R_2(\phi) = \phi$$

$$R_n = \max_{0 \leq b \leq 1} \inf_{x \leq \phi} \{(b_x + 1 - b) R_{n-1}(\phi_{n-1})\}$$

The b units is a probability number meaning the trustability on a market and in the case of our QoS application, is related to the network stability. If b is higher, it means that the network is in bad state and that is necessary to invest in more QoS policies to optimize the use of communication lines. Depending on the value of R_n we have a set of ranges meaning the policy by priority to invest. So, to lower values of R_n the policy is less aggressive in terms of resources to reduce in use.

On next section we will compare this algorithm with a Greedy one, which structure is very simple taking also its input online. Then, Greedy gets an average from a set of lectures (equivalent to the market sequence for the Money Making algorithm) and directly maps to a QoS policy.

5 Experimental Results

To test the proposed algorithm a set of 50000 inputs taken from the internet and representing time of data traveling in milliseconds between two peers. The market sequences were organized in packages of 10 inputs where both Greedy and Money Making algorithms started. The Greedy makes an average of the 10 lectures and then maps the value to a table pointing to the QoS policy to apply. In the case of the Money Making algorithm the result from R_n is scaled and adapted to map it to the QoS policy table. Also, for Money Making the value $b = 0.4$ was set to have a conservative output. All the algorithms and programs where implemented in Java using the UDP protocol to manage the data packages. The obtained result in cost (added and accumulated value when a policy must be applied) is shown in figure 5.

It can be seen that the cost of the Money Making algorithm is over 2000 compared against Greedy that grows faster with the time.

6 Concluding Remarks and Perspectives

This paper presented a strategy to control the QoS of a peer to peer application with the aim to tele-operate robots. One of the main problems is related to preserve the stability of the communication lines by priority where the proposed solution deals with resource management and optimization by reducing bandwidth consumption of the peers in the system. This issue drives to identify and build the QoS policies for the system and then to find an algorithm to have an optimal use of policies to get the QoS. The optimal algorithm is taken from a financial online algorithm where the paper shows how it was related and applied

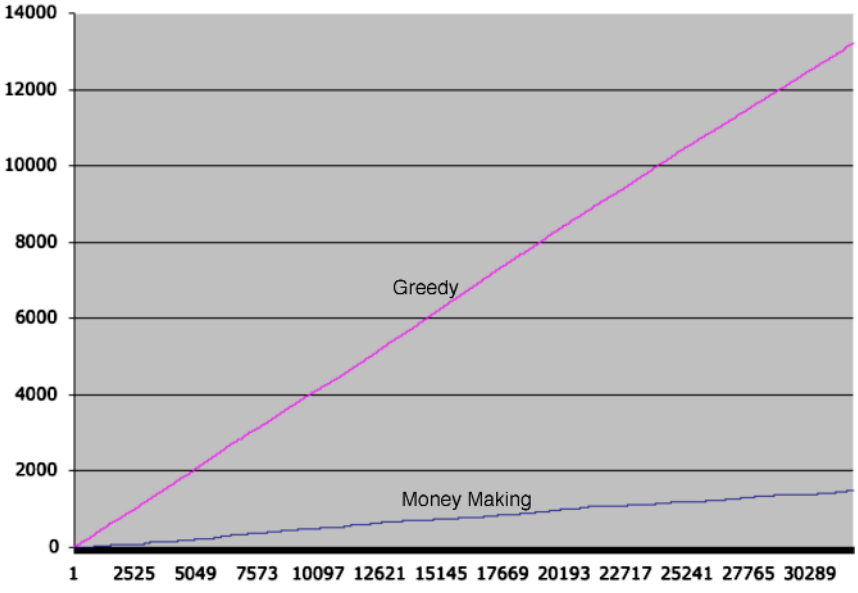

Fig. 5. Accumulated cost between a Greedy and a Money Making for the QoS system where horizontal axis represents the inputs of network latency and vertical axis the accumulated cost per choosing a policy

to the QoS system. The use of the financial recursive algorithm Money Making in QoS open a framework for future works to improve the QoS under different conditions of the network where some parameters can be better adapted as the b factor for the reliability of the network that was fixed as a constant. Other algorithm can detect the network state and adapt a better value for b. Another work to continue is to compare not only greedy but different families of online algorithms (deterministic and random) to improve the system and study how can help the optimization of resources among communities of hundred of users.

References

1. Borodin, A., El-Yaniv, R.: Online Computation and Competitive Analysis. Cambridge University Press (1998)
2. A. Chou, J. R. Cooperstock, R. El-Yaniv, M. Klugerman, and T. Leighton The statistical adversary allows optimal money-making trading strategies In proceedings of the 6th annual ACM/SIAM Symposium on discrete algorithms, 1995.
3. Competitive solutions for online financial problems ACM Computing surveys, Vol30 No.1 March 1998
4. Quality of Service Networking www.cisco.com/univercd/cc/td/doc/cisintwk/itodoc/qos.html Cisco, December 2003.
5. Kalkbrenner, Gerrit., Pirkmayer, Teodor., Dornik, Arnd, Van., Hofmann, Peter.: Quality of Service (QoS) in Distributed Hypermedia-System Technical University of Berlin, IEEE, (1995)

6. Manasse, M. S., McGeoch, L.A., Sleator, D.D.: Competitive Algorithms for On-Line Problems. Proceedings of the 20th Annual ACM Symposium On Theory of Computing (1988) 322–333
7. Park, K.S., Kenyon, R.V.: Effects of Network Characteristics on Human Performance in a Collaborative Virtual Environment. Proceedings of the IEEE VRAIS 99 Houston Texas, USA, March (1999) 104–110
8. Singhal, S., Zyda, M.: Networked Virtual Environments: Design and Implementation. ACM Press SIGGRAPH Series - Addison Wesley, New York (1999)