Predictive Modeling of End-User Quality of Service for Network Based Education

Paritosh Dixit, Mladen A. Vouk and Donald L. Bitzer
Department of Computer Science
North Carolina State University, Raleigh NC 27695
Contact: lapi@galileo.csc.ncsu.edu

Abstract

An important class of network based applications are systems which provide distance education and interactive learning. To be effective they must provide adequate Quality of Service (QoS) support and guarantees. Traditional QoS management approaches tend to focus on network performance and transport layer mechanisms (such as switching, or TCP/IP flow control). However, in the context of network-based education (NBE) it is very important to also address QoS from an end-user perspective. In this paper we regard user-level QoS violations as events that can be described using traditional and rare-event reliability and availability models. We advocate use of such models to predict QoS over the duration of NBE sessions, and based on that initiate appropriate risk resolution mechanisms (including re-negotiation of network-level parameters, and selection of content delivery modes which comply with the predicted QoS). We use the data from an operational wide area NBE system to illustrate the modeling process, and discuss a possible pro-active QoS control mechanism.

keywords: Quality of Service, Distributed Systems, Reliability, Network Based Education.

1 Introduction

An important class of distributed multimedia applications which require service guarantees are distance learning and network based education (NBE) systems. Users of modern NBE services expect not only high quality educational and training content, but also effective delivery and support for these materials. NBE applications require high system reliability, low data loss rates, and low end-to-end response delays or latencies [1]. From an end-user perspective Quality of Service (QoS) that a NBE system can sustain for the length of a user session is of utmost importance [2], [3] and [4].

There is no universally accepted definition of QoS and we adapt a definition from [5] which states that *"Quality of Service represents the set of those quantitative and qualitative characteristics of a distributed multimedia system that are necessary to achieve the required functionality of an application"*. To address issues relating to NBE systems, we broaden this definition of QoS to include not only transport and network layer parameters but metrics which impact end-user perceptions more directly, such as system reliability, keystroke delays at end-user interface, and presentation quality metrics (such as image and audio quality, colors and media synchronization).

Effective large-scale wide-area NBE involves significant synchronous and asynchronous co-operation among multiple distributed applications, as well as platform and network-level tuning and negotiation. Architectures which unify QoS negotiation and evaluation at the application, platform and network level have been proposed by several researchers (e.g., [6] [7] and [8]). However, all appear to focus on short-term reactive process control instead of longer-term predictive control that may be needed in the case of NBE work where the user perceptions and learning is strongly influenced by the QoS delivered to the desktop.

In this paper we consider user level and application level QoS through a set of mechanisms that would allow long-term predictive QoS management. By *long-term*, we mean QoS failure forecasting for periods of several minutes to several hours, or for several days, in to the future. We consider QoS violations as failures that we model using conventional and rare event reliability and availability theory, and we embed intelligence in the decision process by predicting QoS failures and mitigating and resolving the risks they pose. This complements mechanisms such as the ones proposed in [6] [7] and [9].

Our experimental data derive from the operational behavior of NovaNET[1], a wide area computer based education system developed at the University of Illinois at Urbana-Champaign by it's Computer-Based Education Laboratory (CERL). A special characteristic of NovaNET is that it was designed to provide it's users with appropriate QoS. In most part the system operates over a country-wide switched network in order to provide bandwidth and throughput guarantees but it also provides access over the Internet. In our opinion, its switched network component is an excellent example of how Asynchronous Transfer Mode (ATM) based networks should use their QoS mechanisms in Switched Virtual Circuit (SVC) capabilities to provide similar service for NBE

[1]NovaNET is now owned and operated by University Communications Inc.

systems of the future.

Section 2 discusses end-user QoS issues and, section 3 describes an approach to QoS control based on failure modeling. In section 4 we discuss user perception of QoS expressed as reliability. Section 5 provides conclusions.

2 Some End-User QoS Issues

In a NBE system, the most important system entity, and the principal quality driver and constraining influence is, of course, the user. NBE users can be classified into a number of categories. Four exclusive general user categories are of prime importance: students, instructors, authors, and system developers. Examples of other important general categories of users are parents of the students, employers of continuing and adult education students, and educational administrators. Categories of special interest are K12 users, community college users, university users, and adult education users. Functional and usability requirements derive, in most part, directly from the NBE user profile.

System developers and operators are responsible for development, maintenance and operation of the system framework. They develop and integrate system interfaces, administration and management software, communications and scheduling algorithms, authoring tools, courseware generation, material access algorithms and software, and so on. They must be experts in specialized areas such as AI, education, software and computer engineering, and communications. They require specialized tools for NBE system framework development, maintenance, testing, performance evaluation, and operation. They also act as the "operational" staff", e.g. the equivalent of the production staff used in the current real time video classrooms..

Authors are courseware developers. They are responsible for development of individual lessons that are integrated into courses by instructors. It is essential that authors be both pedagogical and content experts. Some of the functionalities that a NBE framework must provide for them are various editors, compilers, interpreters, authoring languages, tools, and capabilities to gather information about the use of their lessons and about any problems encountered with them, as well as courseware security (including protection of copyrights, protection from system crashes and losses, etc.). It is extremely important to note that the authors, for the most part, will not be system experts, and thus the authoring tools and interfaces must be easy-to-learn and easy-to-use and must allow the authors to concentrate on the lesson development rather than struggle with the system intricacies.

Instructors are curriculum developers and material selectors. They sample and combine existing lessons, customize courses and projects, update existing projects and courses, and develop new projects and courses. They also teach and tutor, i.e., they deliver the course material, assist students and oversee student projects. They have to be knowledgeable in the course material area, and they have to be experts in student needs and curriculum construction. The NBE system framework needs to support them when they evaluate student knowledge and progress, grade and compare student work, register students, query student records, write reports, interact and tutor electronically (including shared screen, whiteboards, and voice-based interaction), and give advice. Of course, the system must facilitate curriculum generation, as well as access to different information sources, and it must handle student-related information with special care in that it must preserve the student's right to privacy and yet provide the instructor and other educators with appropriate and needed student and course evaluation information.

Students and **trainees** are the most important users of the system. They require appropriately reliable and timely lesson delivery, easy-to-use interfaces, collaborative support in local and remote joint projects, instructor's help, information about their grades and/or progress in the courses, and so on. The distribution of student support tasks, across the network and across resources, will depend on the task complexity, desired schedules and resource constraints. The solutions should not rule out use of any network type (wire, optical, wireless) or access mode (high-speed and low-speed). However, at any point in time, students' work must be secure and protected from data losses and unauthorized access. Furthermore, the adult, part-time learner is becoming an important customer of (higher education). To meet the needs of this population, we need to develop methods of educational delivery which effectively scale not only the barriers of space and time, but also of student diversity. The "class" of the future is likely to include students who are widely separated geographically, who are not able to "attend" lectures on a preset schedule, and who come with very different backgrounds and from very different walks of life. This presents new demands and challenges for the instructor, who must maintain the quality and integrity of the educational delivery given this diversity.

It is widely accepted [10] that interactive applications using real time support can provide significant learning benefits. For example, in some studies, the knowledge retention rate of trainees was observed to increase from 20% to 75% when using real time interactive support[11]. Another example is a recent study produced by Jerald Schutte, an applied statistics professor at the California State University at Northridge. This study claims that students learning in a virtual classroom (using text posted online, e-mail, newsgroups, chat, and electronic homework assignments) tested 20% better than their students who learned the material in a traditional classroom.

This is good but distance education involves interactions among its users and with the learning environment. McNabb [12] observes that although students may feel that the accessibility of distance learning outweighs the lack of dialogue, today's telecourses still have difficulty replacing face-to-face classes. So, to be effective such applications require more than just bi-directional communication. In studies by [13] and [14], it was found that the quality of the educational process depends upon <u>sustained</u> <u>two-way communication</u>. Without such support students may "drift-away" i.e., become autonomous and isolated, and thus be forced to refresh their short-term-memory buffers. This eventually results in reduced knowledge transfer, and may finally cause them to "time-out" of a session. Therefore, "consistency of performance" during the length of a session is an important guarantee.

The NovaNET experience indicates that an upper bound on keystroke response times should be under 250 milliseconds (ms.) regardless of applications, transport layer mechanisms, or the learning mode (audio,video etc.). In the context of end-user QoS modeling, we stress the need to use *keystroke response times* rather than *network round trip delay* since they represent the round trip delay incurred not only due to network elements but also due to all intermediate and end user processing and scheduling elements (e.g. operating system, scheduling). Traditional approaches have usually concentrated on network delay. Keystroke response times are therefore more relevant as indicators of end-user expectations than end-to-end delays observed over the communication subnet. Standards from ITU [15] indicate that for most applications one way audio delays should not exceed about 200 ms., and that for uncompressed video transmissions the maximum allowable end-to-end delay should not exceed about 250 ms.[2]. NovaNET system measurements also indicate that, once a user starts one hour of work (e.g., a lesson), to maintain reasonable user satisfaction, the probability of getting through that hour without any problems should be above 0.95 [2].

We expect that a good NBE environment would have reliability and other characteristics that at least match above figures. For example if an instructor wishes to use a network based solution in a classroom then that instructor needs to have accepted guarantees that the system will perform adequately when it is needed. Otherwise, network assisted solution will hinder the teaching workflow not aid it. Similarly if a student wishes to rely on an NBE solution to do homework or study, there must be guarantees that the system will be available, and will operate correctly, when the student's workflow requests such assistance.

It is obvious that end-user participation in specification, monitoring and negotiation of the QoS is important either directly or through the agency of the application they are using. There are a number of end-user QoS models, and offer corresponding experimental implementations ([17], [9], [7]), but to the best of our knowledge, none have been widely accepted.

3 Predictive QoS Control

The fact that QoS, at least in the NBE context, should be managed, negotiated and addressed using methods which involve active end-user participation [18] implies a set of predictive application layer services similar to the those proposed in [19]. In general, QoS management at the application level would involve (a) user (or application) derived specification of the desired quality followed by an accurate mapping of these specifications to the lower communication layers, and (b) provision of an intelligent decision interface mechanism that uses that information to facilitate negotiation-renegotiation for it's clients (end-users).

We believe that such a decision interface should rely on short-term and long-term predictive QoS modeling at the level closest to the user. Essential elements of an architecture that would support this approach are (a) model(s) that can predict QoS behavior of the underlying information infrastructure, (b) model(s) that can predict QoS needs of the user/application subsystem, and (c) an appropriately flexible and adaptive mechanism that can respond to these predictions either

[2]MPEG codecs as well as H.120 and H.261 should respect a delay less than 150ms [16]

through dynamic negotiations with the underlying infrastructure or through dynamic adaptation of the application's own QoS needs (e.g., use of delivery modes that are less demanding of the quality or resource that cannot be met).

Of course, there are difficulties associated with the delivery of CM information that are inherent to any distributed communication system. For example, applications using audio and video streams (e.g. real time interactive learning applications using multimedia capabilities) usually involve a variety of resources, architectures and devices distributed across distances. This tends to impose severe constraints on performance and achievable service guarantees. While an integrated architecture that guarantees performance for such applications still remains a challenge, we strongly believe that it will incorporate a "QoS layer". This "layer" could manifest as client resident agents, as broker resident agents, or both [5], [7],[9], [20]. The QoS agents would use appropriate signaling

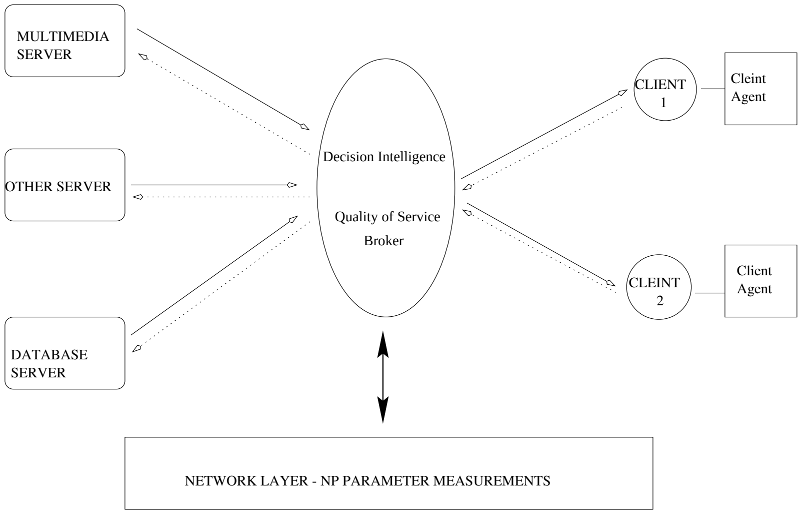

Figure 1: Application Layer Decision Intelligence

and violation detection mechanisms, to prevent violations of pre-agreed QoS guarantees or at least mitigate ill effects. Figure 1 illustrates this.

Existing mechanisms approach QoS management only through real time measurements and control of resource availability and capacity, and therefore they have the drawback of being reactive in nature. In reactive schemes, QoS violations are usually detected after the event has occurred, or at best, concurrently. While such mechanisms may be adequate for static media and bulk transfer applications (such as email, database processing, etc.), real time interactive applications require QoS guarantees for which a more *predictive* and *preventive* approach seems appropriate. A predictive scheme is more attractive than reactive schemes because it allows for a probabilistic evaluation of the QoS *likely* to be available in the future. An example of very short term prediction approach is [17]. In the presence of accurate predictions of future QoS, a client may for example, be able to negotiate for a QoS which would decrease the probability of violation (e.g. choose a media delivery mode which is compatible with the bandwidth forecasted to be available or negotiate a

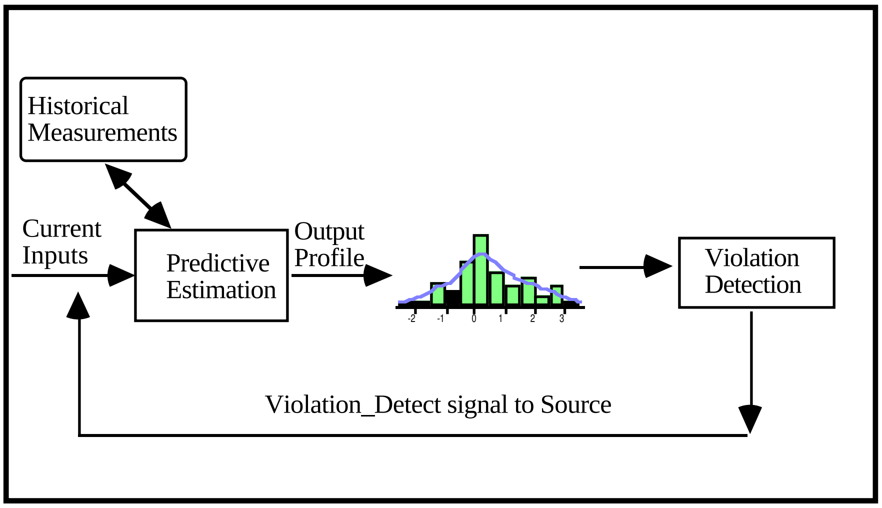

Figure 2: A Predictive and Adaptive Negotiation Architecture

session time which does not violate the requested QoS based upon the forecast.

While forecasting is a common technique used in the analysis of time correlated data it's use in end-user QoS modeling is less common. We feel that stochasticity in the communicating entities (network, servers and clients) provides an added incentive to explore forecasting techniques [21] in the context of *predictive* QoS.

Our framework differs from a real time approach in that we do not wait for a degradation to actually occur, instead we perform predictive maneuvers recursively, and we forecast behavior within specified future windows in time. In this context, we define a *QoS failure event* as any event where a QoS contract is violated. Examples of such events are audio and video quality degradation (for e.g., due to delayed or lost packets), excessive end-to-end response delays at user interface, unavailability of user information, etc. Our approach is meant to provide *reliable* prediction of violations long before they occur. The logic for this predictive approach is illustrated in Figure 2. The feedback process uses historical information to generate a probabilistic profile of anticipated QoS and provides estimates of QoS sustainable in the future. As new observations become available, older data is discarded and the estimates may be updated.

There are many QoS drivers at different levels in a distributed environment. For instance, when a client-server paradigm is used, we usually see a network, a set of servers, and a set of clients. They will all contribute in their own way to an overall value of a parameter such as packet delay and the variability of that parameter over time.

The effects of these different factors on the end-user perceptions of QoS can be analyzed and presented in several ways. For example, one could regard them as arising from interactions of three distinct sources i.e., a **Network** N, **Servers** S and **Clients** C. Alternatively we could regard the whole QoS issue purely from the user perspective of whether or not the user experiences an undesirable event, i.e., a failure. In this case we can use reliability models to gain an understanding of this perception.

An example of a particular metric is round trip end-to-end delay, D. It can be regarded as resulting from network (propagation, transmission, routing), server (processing and queuing delays), and client (scheduling and playback) delays. In a predictive framework, measurements and estimates of contributing delays would be available over time. We can use them to formulate a functional relationship between end user observed delay and the intermediate and contributing delays. For example,

$$y_t = f(t) + e_t \tag{1}$$

where y_t is delay at time t, f is some function of time t which explains the delay observed and e_t is unobservable error. For example, it may be that the time of the day or day of the week causes delay to increase and decrease in a cyclic manner. In this case, a suitable time series model, could be used to feed a control and decision model by providing confidence bounds around mean delays that may be observed at different times of the day. This prediction could then be used to feed a control and decision model that would attempt to fulfill pre-negotiated QoS bounds.

Figure 3 shows round trip network delays measured using the ICMP *ping* [22] program. Approximately 100 packets per hour, each of size 64 bytes were sent over 138 hours (5.75 days) between NCSU and a server. Most of the response times range between 40 ms. and 400 ms. and they display a distinct trend. The first peak occurs in an interval around $x = 1200$ and the crests repeat at intervals of 2400 (i.e. 3600, 4800, 7200, etc.). Measurements were started at midnight towards the end of the week (Thursday) and the peaks therefore correspond to delays during mid-day over the Internet. The peaks seem to be damped for the next three days (till $x = 9600$) and rise again

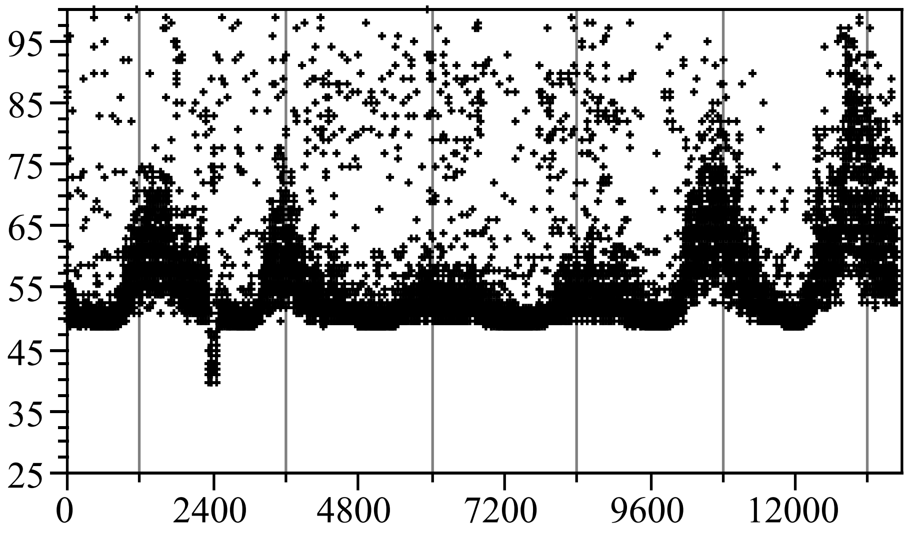

Figure 3: Periodicity in network load conditions

sharply for the next two cycles. We conjecture that this is attributable to differences in the traffic loads during weekends and the beginning of the week. The behavior of the network delays is neither surprising or unexplainable. An Internet path between a sender and receiver involves multiple hops and serialization delays are linear in the number of hops as well as the delay incurred at each intermediate hop (routing, forwarding etc.). However, the presence of such trends may permit us to *predict* delays using time dependency models. It is possible that similar measurements on individual clients and servers may yield trends based upon load and queue statistics. The utility of such predictions is for *tolerant* and *adaptive* clients [19] who argues that such clients gamble for predictive service on the assumption that the recent past is a guide to the near future.

Another important issue with respect to predictions involves consideration of the variance during different time periods. For example, Figure 4 shows two sequences of response times superimposed on the same plot. The lower curve are measurements during the morning (8am-11am) and shows an increasing delay time towards the tail end of the curve, as expected due to increasing load conditions. The second curve, shows similar measurements in the afternoon (12pm-3pm) and shows much larger response times as compared to measurements during the morning. The afternoon measurements show a much larger variability than the morning measurements and an appropriate prediction function will have to account for differences not only in the magnitude (e.g. weighted mean) of the response times but the variability associated with the measuremets as well. An application layer

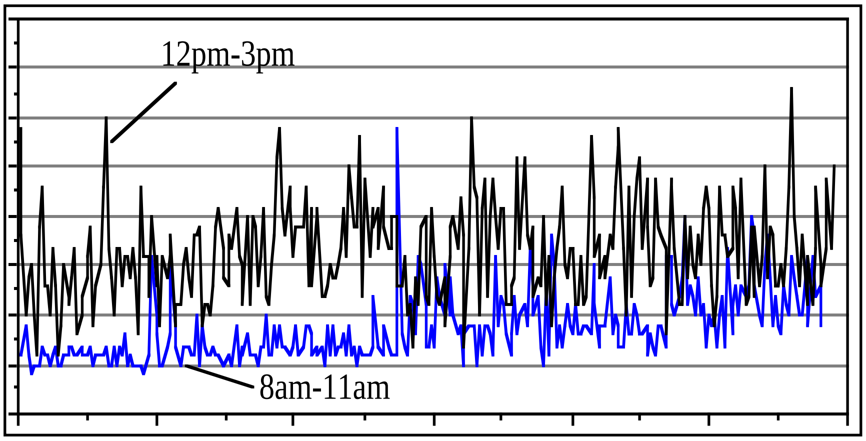

Figure 4: Variance in network response times

prediction model, which can provide predictive estimates with reasonable accuracy and confidence, is likely to be attractive to such clients since they would be willing to scale down (up) their QoS contracts based upon a *good* estimate of the service likely to be available. In an NBE system, for example, an instructor would like to determine, with reasonable accuracy whether keystroke delays are likely to be below a certain threshold (e.g. 250 ms.) for $x\%$ of a two hour class session planned during mid-day. The presence of delays are likely to be less annoying to end users in an NBE system *if* they have *advance* warning of impending delays, e.g. through a predictive estimation model.

In the absence of information about individual parameters and their functional behavior, or if we wish to provide a unified view of violations, it may be enough to regard all violations of the

QoS bounds as user perceived failures, and model them using conventional or rare event reliability and availability models. We illustrate this in the next section.

4 Reliability as a Unifying Metric

Interruptions in user sessions, loss of data, and similar often translates to a user perception that the NBE system has failed to live up to user expectations, that is the user registers a QoS failure [3]. Therefore problems which cause partial or complete loss of service can be viewed as events which correlate strongly with end-user experiences. This suggests that we may be able to represent and control QoS based on a failure model. In order to gain an understanding of the availability and reliability issues associated with a viable NBE system, we investigated the operational behavior of NovaNET. Although NovaNET is centrally operated and managed, its subscribers access it using a wide area network (satellite, fiber, or cable).

In [1], we investigated NovaNET operational reliability and found that the system consistently met end-user reliability objectives over the last 9 years. Figure 5 shows the observed failure rate for the NovaNET system for the past 8 years. The metric is the number of failures per month over the scheduled operating hours in the same month. The observations are based upon approximately $40,000$ cumulative scheduled uptime hours. The failure rate shows a decline over time typical of large computer systems [23]. It is obvious that current NovaNET reliability is very high. It is important to note that although in this case we use system level outages to model end-user perceptions, the correlation between system level failures and user perceptions is a strong one [3]. Therefore, since we classify failures as events where *all* users lost complete system functionality, it is reasonable to assume that a system failure reflects end-user perceptions accurately.

We found that the logarithmic Poisson execution time model (LPET) provides the best fit and prediction [24]. The model has an exponentially decaying mean value function. It's failure intensity function, $\lambda(t)$, is given by:

$$\lambda(t) = \frac{\lambda_0}{\lambda_0 \theta t + 1} \tag{2}$$

where λ_0 and θ are parameters associated with the model and t presents the execution time. Visual inspection of Figure 5 indicates that the model appears to fit the data well and this was confirmed using goodness of fit techniques. From the figure, it is also evident that after the first 12 months of operation, the system has consistently maintained failure rate less than 0.05 failures per month, which research by [3] indicates is consistent with end-user satisfaction. In practice, only recent data may be an indicator of the present reliability experienced by the users. Thus, it is important to assign greater weight to recent events. An approach often used in software reliability modeling is that of <u>data aging</u> [25] in which recent data is given more weight than past reliability data through piecewise failure modeling. Figure 6 shows NovaNET failure data between 1993 and 1997 (51 months). The observed failure intensity, $\lambda(t)$, is in the range

$$0 \leq \lambda(t) \leq 0.0395 \tag{3}$$

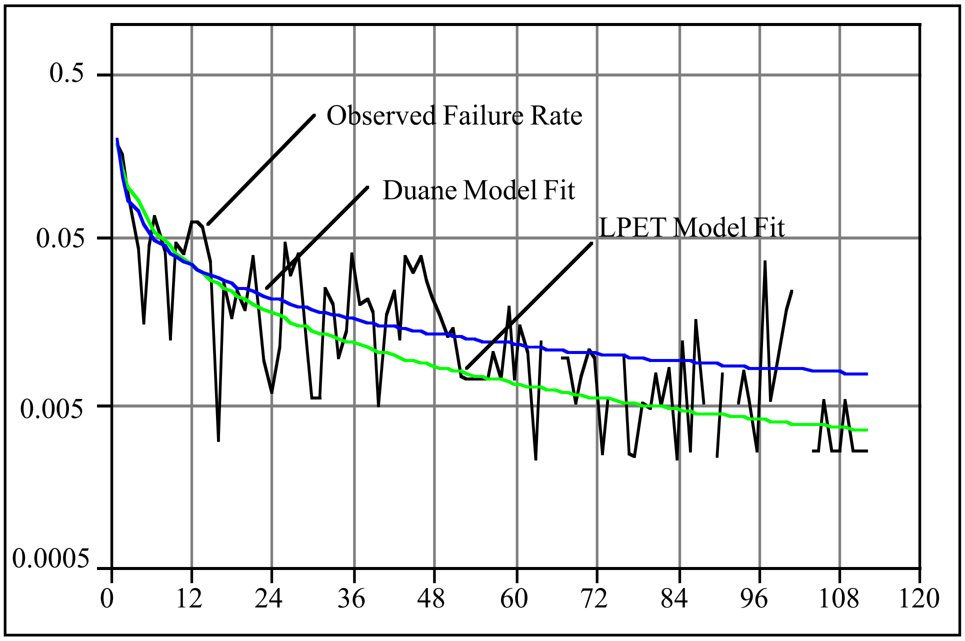

Figure 5: System Reliability 1988-1997

failures per operational hour per month. For most observations, the failure rate is less than 0.015. We find that the mean of the observed data may be an adequate estimator for such a low and stable failure rate. Hence the model we advocate is the constant mean-interval Non-homogeneous Poisson Process (NHPP) model which uses 95% confidence bounds around the mean. In our case $\overline{\lambda(t)} = 0.066$ which translates to [0,6] failures per month.

As an example, consider an instructor who wishes to use this information to plan a class over one whole semester. The above confidence bounds indicate that the instructor might experience anywhere between no failures to perhaps at most 3 failures during the semester [3]. Of course, what matters when a failure occurs is how long it takes for the system to recover from that failure. Analysis of the NovaNET system shows that per failure downtime is about 24 minutes. Since a typical class lasts between 50 and 75 minutes, the instructor would have to be prepared to substitute computer assisted instruction with conventional instruction (or backup from a local system) on perhaps as many as 3 occasions. If the failures occur in a correlated fashion, this may translate into repeated failures during perhaps one class period. This may or may not be acceptable for synchronous teaching, but may be quite acceptable for asynchronous teaching. Asynchronous teaching workflows usually are more flexible, and students who take the course may be able to easily absorb a 30 minute interruption. Our experience is that above reliability guarantees are in most cases sufficient for both synchronous and asynchronous teaching.

Average failure rates are useful for describing a more *global* reliability trend but they are less useful in describing *short term* trends. For example, hourly and weekly variations in the failure

[3] If a semester is assumed to consist of a 3 hour class for 16 weeks equivalent to 48 lesson hours per course

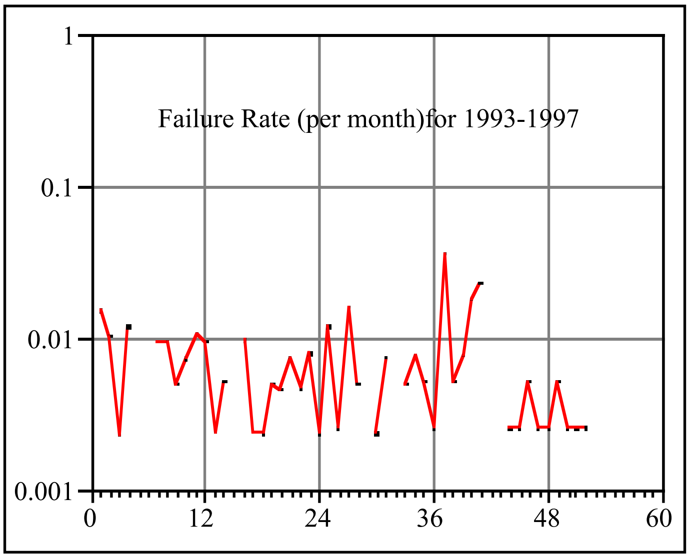

Figure 6: System Failure Rate 1993-1997

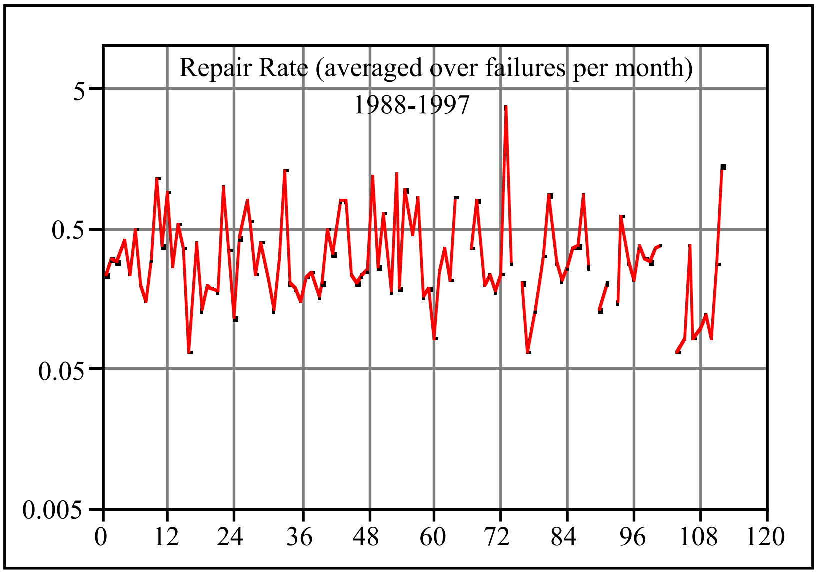

Figure 7: Monthly average downtime per failure

rate will in most cases be smoothed out due to aggregation of the data. Since hourly variation in the reliability of an interactive system are usually more important to an end-user as compared than average variation, we feel it is important to evaluate reliability at both a *micro* (short term) as well as *macro* (long term) levels.

As an example consider the average amount of time the system was unavailable per failure. This is an important metric since it provides us with an estimate of the time users were without service per failure occurrence. For the NovaNET data, the mean downtime per failure per month is shown in Figure 7. It varies between [.05, 5] hours but the mean downtime seems to be roughly constant around 0.5 (actual $\mu = .407$) hours per failure per month throughout the observation period.

Avner [3] has found that indices reflecting the frequency and duration of failures are negatively correlated with an instructor's perception of the quality of computer based education system and in this context. Schneiderman [14] has found that user perceptions of QoS stem from system behaviors observed in short intervals of time.

In fact it is likely that certain *types* of failures will impact QoS more than others. For example, a short communications failure which causes complete system outage may be more irksome than a lengthy software failure that only partially degrades the performance. This has prompted us to investigate short term correlations among failures observed for NovaNET. Interestingly we found that quite often there is correlation between consecutive failure states of the system where a failure state represents a particular failure type. Figure 8 illustrates this phenomena. It shows a probability-transition matrix for the NovaNET failure categories. From the diagonals we see that a self transition is more likely than transition out of a state. From a QoS perspective, this

	C	H	O	S	U
C	**.551**	.138	.073	.086	.150
H	.288	**.447**	.039	.082	.142
O	.330	.190	**.223**	.115	.140
S	.335	.171	.068	**.445**	.184
U	.227	.126	.098	.076	**.470**

Figure 8: Probability Transition Matrix for different failure states

means that users are more likely to encounter failures of a particular type, in a series, than as a random sequence of failure types. This phenomena may be unsuitable for teaching sesions where consecutive failures, all of the same type, may be more disruptive. More investigation would be needed to completely map user experiences with failure type categories.

From the discussion, it is evident that system reliability in NBE systems can be an important end-user QoS parameter since partial or complete loss of system functionality has a negative impact on end-user expectations [3]. Of course a lacks in other QoS indicators such as round trip delay, jitter, availability of lesson material and can be and should be considered through a failure model. Therefore in modeling end-user QoS, it is important to address unified NBE system reliability and availability which includes not only server and client reliability but also the network reliability (including performance failures such as excessive keystroke delays).

5 Conclusions

In this paper we argue that there is a considerable need to manage end-user QoS not only through reactive but also through proactive (predictive) QoS models. We also argue that part of this modeling should be a unified reliability oriented view of the QoS failures as perceived by NBE users. There are some significant differences between existing approaches and the one propose here. First, while guaranteed QoS bounds are often achievable in data networks (e.g., [26]), they are usually intended for end-to-end network connections and do not account for QoS violations resulting from end-user applications (including server and client user level effects). Second, they are often expensive and may lead to wasted resources since *tolerant* and *adaptive* clients, such as NBE system users, may be willing to tolerate different levels of violations given reduced cost. Third, most approaches do not explicitly incorporate the stochasticity (evident in end systems due to applications, scheduling, adapters, etc.) in addition to the variability in network performance. We propose using traditional reliability models for modeling QoS violations and we regard *reliability* as a unifying QoS metric. In this context we presented results from a reliability evaluation of a wide area NBE system called NovaNET. Examination of the reliability trends observed over the last 9 years shows that NovaNET's performance, as an NBE system, was within acceptable bounds and it's future performance can be described well using simple reliability models. We are in the process of collecting additional data and constructing models and control mechanisms which will be suitable for client-server and broker agents that are oriented towards management of end-user QoS.

Acknowledgments

The authors would like to thank Mr.Chris Alix of University Communications Inc., Urbana-Champaign, IL, for his comments and help with data acquisition.

References

[1] P Dixit, M.A Vouk, D.L. Bitzer, and Chris Alix. Reliability and Availability of a Wide Area Network Based Education System. In Proceedings of the 7th International Symposium on Software Reliability Engineering, pages 213–218, White Plains, New York, October 1996. IEEE Computer Society Press.

[2] M.D Bitzer and D.L Bitzer. Teaching Nursing by Computer: An Evaluative Study. Computers in Biology and Medicine, 3(3):187–204, October 1973.

[3] Richard Avner. Measuring Reliability of Networked Educational Computer Systems. Internal CERL Evaluation Report, UIUC, Illinois, August 1993.

[4] Andrew Rindos, Mladen A Vouk, Steve Woolet, Jeff Hines, and James Lester. ATM Technology Enabling Educational Applications Across the North Carolina Information Highway. In Proc. TELECOM '95 - 7th Telecommunications Forum, TECHNOLOGY SUMMIT, volume 1, pages 519–522, Geneva, Switzerland, October 1995. International Telecommunications Union.

[5] A Vogel, G.v. Bochmann, and J Gecesei. Distributed Multimedia Applications and Quality of Service - A Survey. IEEE Multimedia, 2(2):10–19, Summer 1995.

[6] A Campbell, G Coulson, and David Hutchison. A Quality of Service Architecture. In ACM SIGCOMM Computer Communication Review, April 1994.

[7] Patricia G. Florissi. QoSME: Quality of Service Management Environment. Technical Report CUCS-036-95, Department of Computer Science, Columbia University, 1995.

[8] John Zhinky, David E Bakken, and Richard Schantz. Overview of Quality of Service for Distributed Objects. In Proceedings of the 5th IEEE Dual Use Conference, May 1995.

[9] Abdelhakim Hafid and Gregor v. Bochmann. Quality of Service Adaptation in Distributed Multimedia Applications. Technical Report 1045, Department IRO, University of Montreal, August 1996.

[10] L Sherry. Issues in Distance Learning. International Journal of Distance Education, 1(4):337–365, 1996.

[11] G Millbank. Writing Multimedia Training with Integrated Simulation. Presented at Writer's Retreat on Interative Technology and Equipment, University of British Columbia Continuing Studies, 1994.

[12] J McNabb. Telecourse Effectiveness: Findings in the Current Literature. In Technical Trends, pages pp.39–40, 1994.

[13] D.R. Garrison. An Analysis and Evaluation of Audio Teleconferencing to Facilitate Education at a Distance. American Jounal of Distance Education, 4(3):16–23, 1990.

[14] B Schneiderman. Response Time and Display Rate in Human Performance. ACM Computing Surveys, 16:265–285, 1984.

[15] ITU Standard G.114. Audio Delay Recommendations. Available from the ITU web site at www.itu.ch.

[16] W Tawbi, F Horn, E Horlait, and J.B. Stefani. Video Compression Standards and Quality of Service. The Computer Journal, 36(1):43–54, January 1993.

[17] Klara Nahrstedt, Ashfaq Hossain, and Sung-Mo Kang. A Probe Based Algorithm for QoS Specification and Adaptation. In Proceedings of the 4th IFIP Workshop on QoS, IWQOS'96, Paris, France. GMD-Forschungszentrum Informationstechnik GmbH, -53754 Sankt Augustin, 1996.

[18] Thomas Plagemann, Knut Saethre, and Vera Goebel. Application Requirements and QoS Negotiation in Multimedia Systems. In Proceedings of Second Workshop on Protocols for Multimedia Systems, Salzburg Austria, October 1995.

[19] David Clark, Scott Shenker, and Lixia Zhang. Supporting Real Time Applications in an Integrated Services Packet Network: Architecture and Mechanism. In Proceedings of ACM SIGCOMM, August 1992.

[20] Antonio Pulifaafito, Orazio Tomarchio, and Hermann de Meer. An Agent based Framework for Qos Management. February 1997.

[21] Maurice Kendall and Keith Ord. Time Series. Edward Arnold, Great Britain, 1973.

[22] Douglas Comer and David L. Stevens. Internetworking with TCP/IP. Prentice Hall, Englewood Cliffs, New Jersey, 1991.

[23] Mladen A Vouk and W Jones. Field data analysis. Chapter 11, Handbook of Software Reliability Engineering, pages 439–487, 1996.

[24] John Musa, Anthony Iannino, and K. Okumoto. Software Reliability: Measurement, Prediction and Application. Mc-Graw Hill, New York, 1987.

[25] Norman F Schneidwind. Software Reliability Modeling with Optimal Selection of Failure Data. IEEE Transactions on Software Engineering, SE-19(11):1095–1104, 1993.

[26] A.K. Parekh and R.G. Gallager. A Generalized Processor Sharing Approach to Flow Control in Integrated Services Network. IEEE Transactions on Networking, 2(2):137–150, April 1994.