Applying Congestion Pricing at Access Points for Voice and Data Traffic

by

Jimmy Ssu-Ging Shih

B.S. (Massachusetts Institute of Technology) 1996
B.S. (Massachusetts Institute of Technology) 1997
M.ENG. (Massachusetts Institute of Technology) 1997

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy
in

Engineering-Electrical Engineering
and Computer Sciences

in the

GRADUATE DIVISION

of the

UNIVERSITY OF CALIFORNIA, BERKELEY

Committee in charge:

Professor Randy H. Katz, Chair
Professor Pravin P. Varaiya
Professor John C.-I. Chuang

Spring 2003

The dissertation of Jimmy Ssu-Ging Shih is approved:

_____

Chair Date

_____

Date

_____

Date

University of California, Berkeley

Spring 2003

Applying Congestion Pricing at Access Points for Voice and Data Traffic

by

Jimmy Ssu-Ging Shih

Abstract

Applying Congestion Pricing at Access Points for Voice and Data Traffic

by

Jimmy Ssu-Ging Shih

Doctor of Philosophy in Engineering-Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Randy H. Katz, Chair

For alleviating network congestion, many researchers [24-26] have advocated the use of congestion pricing, varying prices according to load, as a feedback mechanism for modifying user demand. However, it is not clear whether it can be designed to be acceptable to users and still be effective for operators. Thus we investigate user interface and system issues by applying congestion pricing at access points for voice and data traffic.

For voice, we deployed a computer-telephony service to 100 users for over one year to investigate changing prices during phone calls. We conducted user experiments to understand user response and acceptance to price changes. Using the results, we developed a user behavioral model to drive large-scale simulations for understanding how operators should manage congestion pricing and the tradeoffs they would face. The latter strongly depends on the user model. Therefore, we re-measured user reactions to price changes under a large-scale emulated service with many simulated users making calls and responding to the same price changes. We found that dynamic pricing can be effective for large user populations because users are receptive and responsive to

occasional price increases. Prices only need to change 4% of the time, and doing so can dramatically reduce call blocking rate by 50% or save provisioning by 20%.

For data, we conducted user trials with 10 participants to examine using dynamic pricing to allocate bandwidth at a LAN access link. We found that offering users three classes of service based on traffic smoothing and charging once every 10-15 minutes is effective and acceptable. Through experimentations, users can easily be enticed to have their traffic smoothed by changing prices. Using surveys, users stated that they like choosing between different average performances and making purchasing decisions at most once every 15 minutes. Using simulations, we found that applying the scheme in a large network can easily reduce its access link bursts by 20-30%.

Through user studies and simulations, we show that applying congestion pricing at access points for voice and data traffic can be effective and acceptable. Thus, it should be considered for allocating limited bandwidth at access points.

Professor Randy H. Katz

Dissertation Committee Chair

To my parents,

Scott Shih and Shiew-Mei Shih

Contents

List of Figures

List of Tables

Acknowledgements

The two great things about graduate school are the professors and the graduate students. I was fortunately to have Randy Katz as my advisor. He taught me a great deal about conducting research and working with people. I also want to thank Professor Anthony Joseph who continuously provided me with advice and support throughout my graduate study. Finally, I want to thank Professor Pravin Varaiya and Professor John Chuang for being my dissertation committee and providing me with feedback on my thesis.

Throughout my study, I was able to work with and befriend with many exceptional graduate students at Berkeley. I especially enjoy working with Helen Wang and Bhaskar Raman, and everyone at the ICEBERG and SAHARA project.

My graduate school experience would not have gone smoothly without the hard work by the group's staff members. I want to thank Keith Sklower for setting up and teaching me various computer networks and systems. I also want to thank Bob Miller and Damon Hinson for ordering all sorts of equipments for me.

In graduate school, I was very fortunate to receive strong assistance from several companies. I want to thank Ericsson, Motorola, Lucent Elemedia, and Packeteer for providing me with equipments, technical support, funding, and feedback on my research.

Finally, I want to thank my family and friends for their support and encouragement throughout my study in graduate school.

Thank you everyone

Jimmy Shih, March 2003

Chapter 1 Introduction

1.1 Problem Statement

Allocating Scarce Resources

There is never going to be enough network resources. Even though capacity increases with new technological advances, demand has always eventually outpaced supply. Furthermore, shortages will be an even more pressing issue in the future. With faster connections to networks, a few users with demanding applications, like streaming video, can easily consume tremendous amount of resources. Thus resource will exhibit a wider range of utilization (with unpredictable pattern of usage). With low average utilization but high peak usage, it will be more expensive for operators to provide resources for peak situations. Thus, congestion will occur more frequently and become more severe. In sum, resource allocation is an issue that needs to be addressed.

Using Congestion Pricing

From an economic perspective, there are two approaches for allocating scarce resources. One is to vary the quality of the resources while the other is to vary the prices (see Figure 1.1). When the prices are fixed, the quality needs to be varied. In the simplest scenario of one service class, when demand exceeds capacity, everyone needs to suffer degraded performance or some users need to be denied usage. A single service class assumes that everyone values resources equally during contention. In the scenario of multiple service classes, each class charges a fixed price and everyone within a class is treated the same. However, the burden of congestion can shift from one class to another;

thus, the capacity for each class can vary. Regarding user demand, if users can only switch between classes on a long time-scale, then multiple service classes simply take into account that different users value resources differently and would select and pay for different classes. If users can switch between classes dynamically, then multiple service classes also take into account that a user would value resources differently at different times. Even though both the demand and the capacity for each class can vary, with fixed prices, when the demand of a class exceeds its capacity, the quality for that class still needs to be varied. The second approach for congestion control is to vary the prices dynamically. The idea is to match demand to capacity by increasing prices during congestion to reduce demand, and decreasing prices during low utilization to increase usage. Thus, dynamic prices act as a feedback mechanism from resources to users (see Figure 1.2). With dynamic pricing, one can keep the quality constant or still have it varied. Additionally, one can apply dynamic pricing in situations with one service class or multiple service classes. Dynamic pricing is promising for resources like telecommunication bandwidth where congestion can occur all of a sudden and users can quickly adjust their usages[1]. While much of the network research on resource allocation concentrate on the approach of fixed prices and variable quality, in this work, we instead focus on varying prices and predictable quality as an alternative mechanism for resource allocation.

[1] The frequency of price changes depends on the predictability of demand pattern and the time-scale of concern. If demand is predictable on an hourly basis, then time-of-day pricing, a form of variable pricing, can be used to alleviate congestion on this time-sale. In this thesis, we assume that demand varies on a short time-scale (minutes and seconds) and that we want to use prices to deal with these sudden changes in demand.

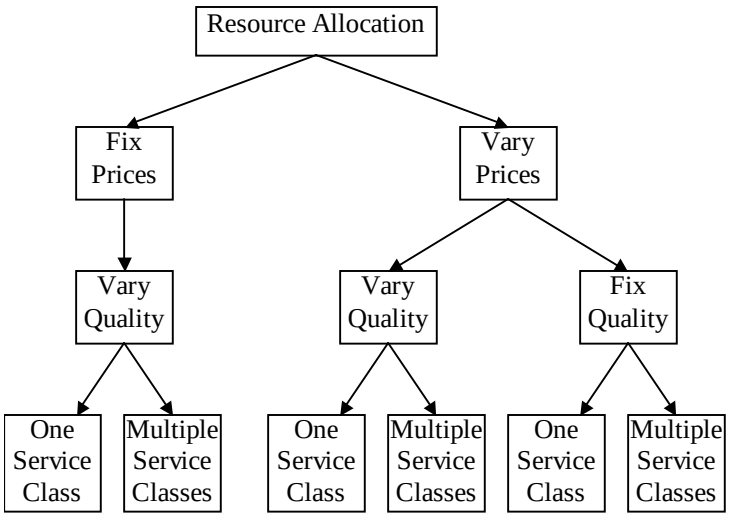

Figure 1.1: A classification of the approaches for resource allocation.

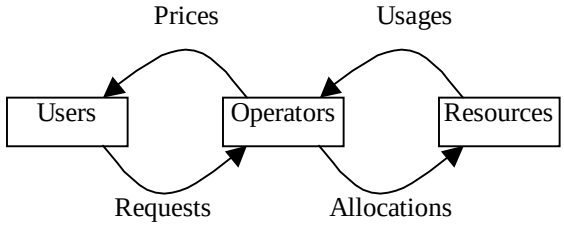

Figure 1.2: Flow of information between users, operators, and resources in dynamic pricing.

There are actually two models for varying prices to reduce congestion. One is to use *congestion pricing* where prices are set according to load and users decide how much to purchase at the current prices. The other is to use *auctions* where users make bids and resources are given to the highest bidders. The advantages of congestion pricing are that it will scale to more users and that users are more certain of the outcomes. Scaling is easier because users do not need to synchronize their actions. Users have more certainty because they know they can obtain resources if they are willing to pay. In contrast, both prices and allocations are unknown for users until the end of an auction. However, the advantage of auctions is that it provides resource owners more certainty by

instantaneously matching demand to supply without any surplus or shortage. In comparison, under congestion pricing, owners need to continuously monitor past usages and adjust prices. Thus congestion pricing places the burden of uncertainty on operators while auctions shifts the burden to users. In this work, we select congestion pricing for varying prices because it can scale to larger number of users and because owners of network resources are in a better position to handle the uncertainty of allocation than users.

Congestion pricing can benefit both operators and users. For operators, dynamic pricing can achieve *economic efficiency*, meaning that no user who is being denied of a resource would value it more than those who are currently using it. It can achieve economic efficiency by using prices to modify user behavior. More specifically, it can vary prices according to load and inform users of the current prices to encourage some users to conserve resources or to shift their usages to another time of lower contention. Thus, those who value the resources the most would use them first. Furthermore, it offers operators an extra dimension of flexibility in making tradeoffs between system performance and user satisfaction. It can improve system performance by reducing congestion, supporting more users with a given capacity, and saving money in terms of the needed capacity. On the other hand, it can reduce user satisfaction by interrupting users with announcements of price changes and causing users not to be able to predict their costs ahead of the time. Thus operators can trade user satisfaction for system performance. Congestion pricing can also benefit users through reduction in congestion level. With less congestion, less capacity would be needed, and cost and prices would decrease. Furthermore, with dynamic pricing, users have the option of obtaining good

service quality when they are willing to pay, as opposed to always suffering poor performance during congestion. Thus, congestion pricing is a promising resource allocation mechanism for both operators and users.

However, there are also drawbacks of using congestion pricing for operators and users. For operators, dynamic pricing makes implementation and accounting more complicated. For users, besides being interrupted with current price information and forced to tolerate unpredictable costs, they need to trust operators to set prices according to load. Under congestion pricing, operators have incentives to set prices higher than needed or let congestion persist unabated. However, operators can alleviate these user concerns and entice users to accept congestion pricing by passing some of its benefits, less congestion and lower infrastructure cost, to users.

At Access Points

There are four scenarios of applying congestion pricing to network resources. The first is to apply it at a single bottleneck like an *access point*, the first aggregation point for accessing network resources (see Figure 1.3A). The second is to apply it locally at multiple bottlenecks where users only need to access one of the bottlenecks at a time (see Figure 1.3B). For example, congestion pricing can be applied so as to allow users to choose between a slow modem pool and a fast modem pool. By informing users of the prices at both modem pools, congestion pricing can be used to adjust load at each modem pool and balance loads across the two. The third scenario is to apply congestion pricing locally at well-defined bottlenecks where usages need to span multiple bottlenecks (see Figure 1.3C). For example, an Internet Service Provider can apply congestion pricing at each router to allocate a path's bandwidth. The last scenario is to apply congestion

pricing at the edges of a network (see Figure 1.3D). It treats the network as an opaque cloud when adjusting the prices at the edges according to the overall congestion level within the network. These scenarios gradually increase in their complexity in determining prices.

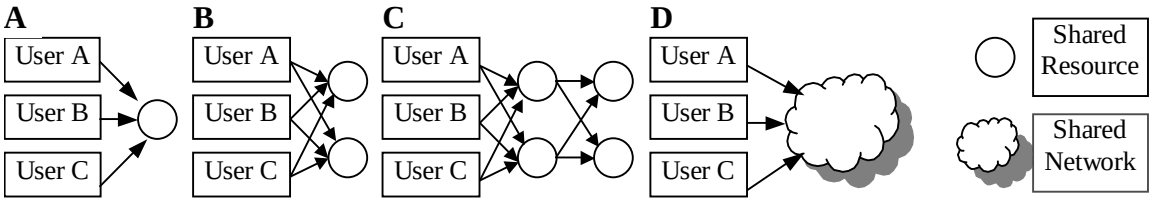

Figure 1.3: Different scenarios of applying congestion pricing.

In this thesis, we focus on applying congestion pricing at the simplest scenario, a single bottleneck. The bottleneck can be a physical or a logical resource. A good example of a single bottleneck is an access point because it is frequently congested. Furthermore, by focusing on an access point, we simplify issues like monitoring, price setting, allocation, and accounting that would have made the more complicated scenarios more difficult to study. Finally, it is good to first understand the usefulness of congestion pricing in the simplest scenario because its feasibility and acceptance in real systems are unclear.

For Voice and Data Traffic

In this thesis, we investigate applying congestion pricing at an access point for both voice and data traffic. These two resources are quite different. Voice traffic is session-oriented and each session (call) usually lasts on the order of minutes. At an access point, each session uses a bounded amount of resources. For voice calls, users only require one service class, one that can support a two-way conversation. Users

usually have some general calling pattern (e.g., some users prefer to call at night), however, users always want to call when they need to. In comparison, data usage is bursty in duration and in volume. Users have multiple service requirements in terms of quality and resources. Furthermore, users tend to send data at the same time, causing traffic to be self-similar [23] and bursty. In summary, voice traffic allows us to explore applying congestion pricing for a single class of service with constant bandwidth requirement while data allows us to investigate multiple classes of service with variable bandwidth demand.

In summary, the problem we address in this thesis is allocating scarce network resources. In particular, we explore using congestion pricing to achieve economic efficiency by modifying user behaviors. We focus on control at an access point because it is the simplest and perhaps the most feasible scenario for congestion pricing. We study voice and data traffic because they represent very different resources for evaluating congestion pricing.

1.2 Challenges

There has been much discussion [15, 24-26] on the benefits and drawbacks of congestion pricing in the literature. Other works [28, 31- 33] have used calculations and simulations to compare congestion pricing with flat-rate pricing. However, these studies strongly depend on how they model user behavior in response to price changes. There are very few works [10, 11, 22, 35] that perform pricing studies with real users in real systems. More user evaluations of congestion pricing on actual systems are needed to understand the practical engineering issues like how to apply congestion pricing and how to manage it.

The goal of user evaluations is to design a scheme that is acceptable to users and effective for operators. To be acceptable, users need to be willing to use and tolerate dynamic pricing. We can understand user acceptance by surveying users about their experience and measuring how frequent prices need to change. To be effective, users first need to be responsive to price changes. For voice traffic, it means encouraging users to terminate their sessions early or shift their sessions in time. For data traffic, it means enticing users to use less bandwidth, tolerate more delay, or experience more loss. Next, having users respond to price changes would need to actually be effective in reducing overall congestion. For voice traffic, effective means reducing call blocking rate or capacity. Similarly, for data traffic, effective means reducing burstiness (so that less packets are dropped or delayed) or capacity. We will elaborate more on the metrics of acceptance and effectiveness for voice and data traffic in the beginning of Chapter 3.

To find an acceptable and effective scheme, we need to address both user interface and system issues. The former is particularly important because congestion pricing requires users in the control loop. Thus congestion pricing needs to provide users with real-time pricing information. Users need to understand the implications of pricing information, the choices they have, and the consequences of their actions. We also need to provide users with incentives for changing their behavior, and we should not overburden them with information and decisions. For system issues, we need to make sure that having a certain percentage of users change their behavior would actually have an impact in reducing overall congestion. Furthermore, we need to ensure that the extra load imposed by congestion pricing, such as sending out periodic pricing information, is

not excessive. User interface and system issues come into play when deciding on design issues like:

- What are the goods to charge?
- What constraint to place on users?
- How to set prices?
- How often to charge users?
- How and how often to provide feedbacks to users?

For voice traffic, in addition to finding a scheme that is acceptable and effective, we need to understand the tradeoffs (e.g., expected reduction in provisioning, expected frequency of price changes, etc.) under large user populations. The tradeoffs under a small group of users are of limited importance. Each phone call only requires a fixed amount of resources. Thus the scale of operation that operators are really concerned about involves many thousands of users. However, it is difficult to conduct a large-scale user experiment. Thus we need a methodology that can make convincing the results based on a small-scale user study.

For data traffic, the main challenge is that dynamic pricing for bandwidth is a concept foreign to most users. Most people just want to accomplish their tasks when using bandwidth. They are resistant to using an extra layer of pricing functions. Furthermore, there is a large design space for applying congestion pricing to data. For example, the goods to charge can depend on bandwidth, delay, loss, etc. Thus, a congestion pricing scheme can vary in many dimensions, and we need to explore a large design space to find an acceptable and effective scheme.

1.3 Approaches, Testbeds, and Results

Approach for Voice

For voice traffic, we utilize the following methodology to scale up the results from a small-scale user study (see Figure 1.4). We first conduct a small-scale user study to understand how users behave under congestion pricing. In particular, we study whether they would react to dynamic prices, how sensitive they are to different price increases, and whether they would accept congestion pricing. Then using a user model derived from these user experiments, we use simulations to understand how an operator, when involving many users, should set the appropriate parameters for managing congestion pricing and the tradeoffs it would face. The latter strongly depends on the user model. Therefore, we combine user experiments with simulations to further re-measure the user model in an environment emulating a large-scale service. In emulation, we have users reacting to price changes set by an operator who is responding to the load and the reactions of many simulated users. Thus, we verify our user model by re-measuring real user behavior under a large-scale service setting with many simulated users behaving in a realistic manner in their resource demand and responses to price changes.

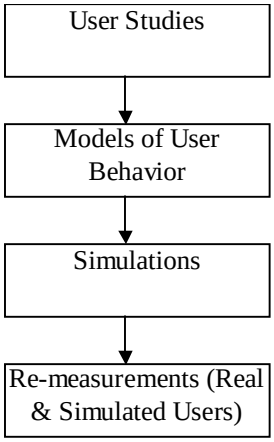

Figure 1.4: Methodology for scaling the results from a small-scale user study.

Testbed for Voice

For voice traffic, we investigate applying congestion pricing for a voice-over-IP gateway service. The service allows users to make calls using their computers or phones through the use of gateways connecting the Internet to the PSTN[2]. For the service, we assume that the owner has many gateways located at the core of the Internet (see Figure 1.5). We assume that the latency between the gateways is negligible for voice applications, and that the owner can use redirection agents to direct or migrate calls to less crowded gateways. Therefore, all the gateways together can be viewed as a single large logical access point serving many users. The service's bottleneck is the number of phone lines connecting to the PSTN at the gateways. Phone lines are a dedicated resource with fixed cost irrespective of usage. It costs money to have phone lines even if no call is placed. Thus the owner would like to minimize the number of phone lines required to achieve a certain call blocking rate.

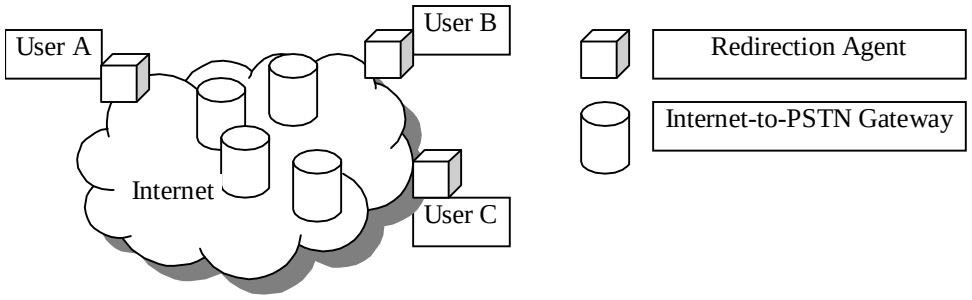

Figure 1.5: Internet-to-PSTN gateway architecture with gateways at the Internet core and redirection agents at the edges.

Results for Voice

We deployed a voice-over-IP gateway service to 100 students in dormitories for over one year. Through experimentation, we found that users are responsive to price

[2] Public Switched Telephone Network (PSTN).

increases if prices change neither quickly (at most once every three minutes) nor frequently (at most a few percentage of the time they are talking). From simulations and re-measurements, we found that increasing prices only during congested periods can easily reduce call blocking rate by 50% or reduce provisioning requirements by 20%, while only causing users to experience price changes in 4% of their usages.

Approach for Data

For data traffic, we use a simple methodology of iteratively performing prototyping, deployment, and analysis to discover a scheme that is acceptable to users and effective for operators. It is also difficult to conduct large-scale user experimentations. Therefore, we use a small-scale deployment to understand user reaction and acceptance to congestion pricing, and analysis to evaluate the tradeoffs under larger user populations.

Testbed for Data

For data traffic, we investigate applying congestion pricing at a Local Area Network (LAN). The bottleneck is the LAN's *access link*, its connection to the rest of the Internet, because it is shared among many users. Furthermore, one needs to pay Internet Service Provider money for usage and capacity of the access link; thus, system administrators have financial reasons to further limit the access link bandwidth. LAN congestion can be severe because a few users running bandwidth intensive applications can easily cause others to suffer poor performance. A possible solution is to limit each user to a certain amount of bandwidth, but this would forbid demanding applications and reduce the value of the network. In sum, system administrators would like to minimize

congestion of a shared access link so that they can provide users with large and predictable bandwidth access.

Results for Data

We conducted user experiments that use congestion pricing to allocate LAN bandwidth with about 10 users. We performed two iterations of experimental design-prototyping, evaluation, and analysis. In the first iteration, we offered users three bandwidth choices and gave each user limited tokens each day for bandwidth purchases. We adjusted the sizes' prices according to load and allowed users to dynamically switch between the sizes. We found that using rate-limiting is not suitable for users because it is hard for them to adjust their bandwidth sizes to react to short duration bursts. In the second iteration, we offered users three Quality of Services (QoSs) differing by degree of traffic smoothing. Traffic smoothing removes short-term fluctuations and adjusts a user's traffic to his/her long-term average. We then charged users certain number of tokens for using a QoS for 15 minutes. We found that this scheme is both acceptable and effective. It is acceptable because users can obtain different levels of average performance with smoothing and users only need to make a purchasing decision at most once every 15 minutes. It is effective because we can easily entice users to select a lower QoS, one with more traffic smoothing, by raising the price of a higher QoS, one with less smoothing. Using simulations, we found that using a 15 minutes charging granularity would only slightly reduce the effectiveness of congestion pricing. Furthermore, we estimated that if half of the users in a large network can be enticed to have their traffic smoothed, then the overall burstiness at the network' access link can be reduced by 20-30%.

1.4 Contributions

We present a methodology for using a small-scale user study for evaluating a system meant to serve a much larger user population. Using the methodology, we showed that applying congestion pricing to voice calls can allow an operator to make a good tradeoff between system performance and user satisfaction. Prices only need to increase during occasional high loads, users would accept and respond to occasional price increases, and doing so in a large-scale service can greatly reduce call blocking rate or provisioning. For voice traffic, we also derived a user model based on user experimentation and formulated a set of rules for operators to manage congestion pricing when involving many users. For data traffic, we found that using traffic smoothing and charging once every 10-15 minutes is acceptable and effective in reducing access link bursts. In reaching these conclusions for voice and for data traffic, we present an acceptable and effective scheme of applying congestion pricing to users, and an estimate of the benefits and drawbacks of dynamic pricing.

1.5 Dissertation Roadmap

In the next Chapter, we first review the related work on congestion pricing. In Chapter 3, we describe in more details our methodologies and testbeds for evaluating congestion pricing for voice and data traffic. In Chapters 4 through 6, we detail our effort in applying congestion pricing to voice traffic. In Chapter 4, we first explain how we rapidly prototype different computer-telephony features when building up a user community for a voice-over-IP gateway service. In Chapter 5, we discuss the various pricing policies we experimented with the user community using the service. In Chapter 6, we describe our simulation study for understanding the tradeoffs of congestion pricing

on a larger scale, and how we verified the simulation results by combining user experiments with simulations. In Chapter 7, we report our experience in applying congestion pricing to data traffic. Finally, in Chapter 8, we conclude with our findings on congestion pricing.

Chapter 2 Related Work

Research in Internet pricing started with papers [27, 37] that advocate for networks to offer multiple classes of service and to use usage-based pricing (static pricing) for providing users with incentives to choose among the classes. Next, several papers [24-26] emerged in favor of using dynamic pricing to more efficiently allocate scarce resources. Subsequently, several simulation studies [28, 31-33] performed comparisons between dynamic pricing and flat-rate pricing. Other simulation papers [17, 36, 41] evaluated the performance of dynamic pricing when applied in a distributed fashion across a network. At the same time, other papers [1, 2, 9, 12, 13, 16, 29, 38, 43] focused on implementation issues and used simulations to evaluate them. Afterwards, some prototyping efforts [14, 39, 42] demonstrated the feasibility of implementing dynamic pricing in real systems. Finally, some user studies [10, 11, 22, 35] have been conducted to better understand the effectiveness of pricing. However, from the existing work, it is still unclear how to actually apply and manage dynamic pricing in real systems.

We summarize a few overview papers [7, 20, 40] on using pricing for resource allocation in Section 2.1. In Sections 2.2 to 2.7, we describe in detail each of the groups of papers mentioned above. In Section 2.8, we conclude with the appropriate next step to take.

2.1 Overview Papers

Several overview papers [7, 20, 40] summarized the prior work on Internet pricing. Henderson, Crowcroft, and Bhatti [20] categorized different pricing schemes based on the network locations where charges are incurred. They mentioned that pricing can be applied at every node, at access links, or between service providers. In contrast, Chang and Petr [7] categorized the related work first into static versus dynamic pricing (see Figure 2.1). The difference between them is that static pricing is independent of real-time network utilization. For static pricing, they further divided the work into those advocating for per-byte, per-packet, per-time-of-day, per-connection, per-service-class, or per-volume (product of traffic rate and duration) pricing. For dynamic pricing, they separated the work into cases where users send best effort traffic (workload independent of prices), elastic traffic (workload dependent on prices), or guaranteed traffic (workload with stringent performance requirements). Using a more detailed classification scheme (see Figure 2.2), Stiller, Reichl, and Leinen [40] categorized different pricing schemes based on the following three dimensions:

- Technical dimension:
  - Service categories (e.g., connection-oriented versus datagram, one service class versus multiple service classes, etc.).
  - Charging parameters (e.g., based on peak, average, congestion, etc.).
- Economic dimension:
  - Tariff components (e.g., access fee, setup fee, usage fee, etc.).
  - Efficiency (e.g., maximize profit, maximize user utility, recover cost, etc.).
- Research dimension:
  - Theoretically oriented.

- Application oriented.

Furthermore, they mentioned that different pricing models differ in application requirements (e.g., burstiness issues), technological and economical issues (e.g., sender or receiver based payment, marginal cost, congestion/responsive pricing), and practical issues (e.g., transparence, predictability, practicability, fairness, user acceptance, and user friendliness).

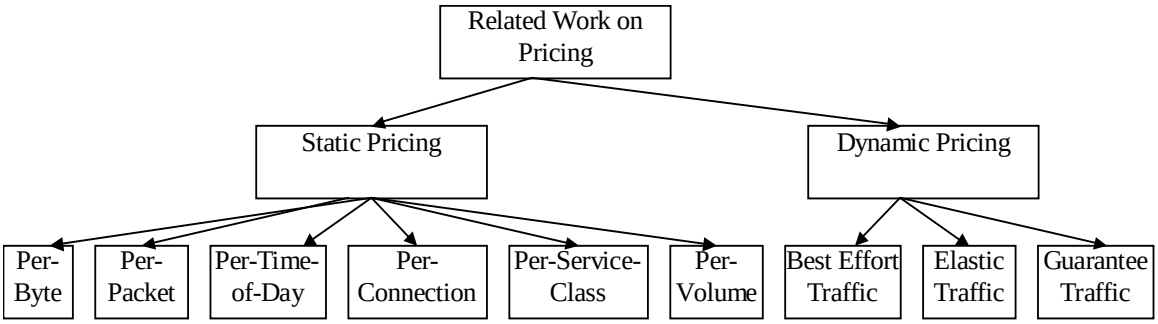

Figure 2.1: Classification in [7] on the pricing related work.

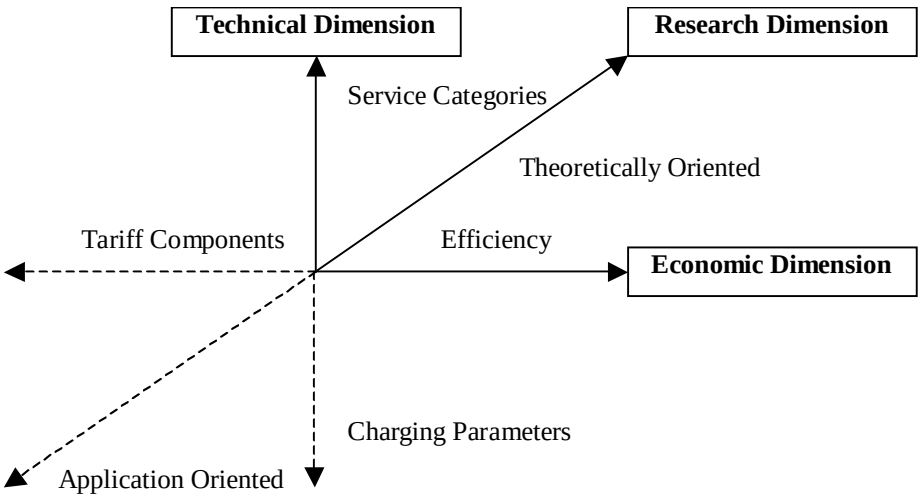

Figure 2.2: Dimensions in [40] of the various pricing schemes.

As a note, according to classification in Henderson, Crowcroft, and Bhatti [20], we focused on applying congestion pricing at access links. According to taxonomy in

Change and Petr [7], we investigated dynamic pricing for elastic traffic. For the dimensions in Stiller, Reichl, and Leinen [40], we studied the following:

- Technical dimension:
  - Service categories: one service class for voice, and multiple service classes for data.
  - Charging parameters: based on congestion.
- Economic dimension:
  - Tariff components: variable usage fee.
  - Efficiency: minimize capacity, maximize utilization, minimize congestion, and minimize user interruption.
- Research dimension:
  - Application oriented: how to apply and manage congestion pricing for voice and data.

2.2 A Case for Usage-Based Pricing

In an influential paper, Shenker [37] advocated for usage-based pricing so that networks can more efficiently meet application needs. More specifically, it recommended a *per-user, quality-of-service sensitive, usage-based* pricing. It favored per-user quality-of-service sensitive pricing so that networks can offer multiple classes of service and use pricing to provide users with incentives for specifying appropriate classes. It supported usage-based pricing so that users do not have reasons for reselling services to others. In another influential paper, MacKie-Mason and Varian [27] argued that usage-based pricing is desirable because it presents information to users about the true costs of their actions. They debunked several myths about usage-based pricing like hurting small users and increasing provider profits. Furthermore, they calculated that accounting cost for usage-based pricing, even though it is big compared to incremental cost, would be small

compared to the total cost of providing a network. These two papers generated much of the initial discussions on Internet pricing.

From looking through many simulation studies on pricing, a general simulation framework for studying pricing can be characterized by four components, *network model*, *workload model*, *user model*, and *metrics* (see Figure 2.3). The network model describes network bottlenecks and causes of congestion. Congestion in turn affects prices and network performance. The workload model describes user usage regardless of prices. The user model then describes how users would behave in response to prices and network performance. Some simulations combine the workload model with the user model by describing the user workload in response to prices and network performance. Finally the metrics are dimensions used for comparison. In this chapter, we use these four components to describe all the simulation study setups.

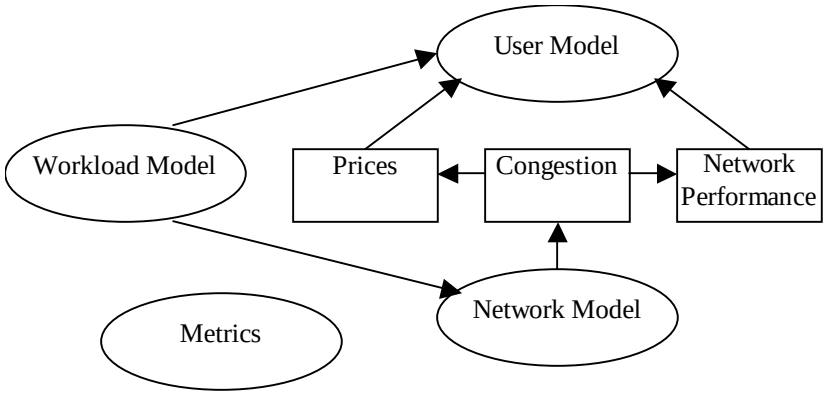

Figure 2.3: A general simulation framework for using pricing to allocate scarce resources.

Several papers [8, 15, 30] employed simulations to understand the consequences of exploiting usage-based pricing. Table 2.1 summarizes the setups for the following two studies. Parris, Keshav, and Ferrari [30] found that peak-load pricing, like time-of-day pricing, can spread load evenly. Furthermore, they found that per-packet pricing can

cause an operator to charge a high price to serve only the high-value users instead of charging a low price to serve both the low-value and the high-value users. Cocchi et al. [8] used packet-level simulations to conclude that it is possible to set prices in a priority scheme so that the performance penalty received for requesting a less-than-optimal service class is offset by the reduced price of the service. They also found that priority prices can be set without making the optimal service class so expensive that even the performance-sensitive users do not use it. However, Fishburn and Odlyzko [15] used simulations to show that in an environment where demand increases rapidly and cost decreases quickly, building a single QoS network with flat-rate pricing is cheaper than building a network containing multiple QoSs with differential pricing. These simulations pointed out the possible benefits and drawbacks of usage-based pricing.

Table 2.1: Summary of the simulation setups for usage-based pricing.

Authors	Network Model	Workload Model	User Model	Metrics
Parris, Keshav, and Ferrari	One bottleneck. Two service classes, one for phone calls and the other for video calls.	A fraction of the requests is for phone calls and the other fraction is for video calls. Each request has Poisson arrival rate and exponentially distributed duration.	A fraction of the users is rich and the other fraction is poor. Each user makes a call if he/she has enough money.	Revenue. Blocking probability. Network utilization.
Cocchi et al.	One bottleneck. Four service classes.	Four applications. Each application has a Poisson arrival rate and exponentially distributed duration. A certain number of users using each application.	Each user has a utility function that depends on cost and application performance. Each user selects a service class based on its application requirement.	User utility function.

For real world experience, there are reports of positive and negative experience with usage-based pricing. Brownlee [5] found that combining usage-based pricing with hierarchy-based pricing can easily recover the cost of an expensive link, like an

international link. On the other hand, Baeza-Yates, Piquer, and Poblete [3] found several problems with usage-based pricing. First, they found that heavy users strongly protested when usage-based pricing is used. Second, they did not know how to deal with unsolicited incoming traffic, like public ftp traffic. Third, they could not peer with another network that charged flat-fee for its international link. When they peered, all the users preferred to route their traffic through the flat-fee link instead of their usage-based fee link. In retrospect, they mentioned that flat-fee pricing would only increase cost slightly from usage-based pricing. From their experience, they concluded that variable pricing like usage-based pricing would only work for a monopoly or on a global agreement basis.

2.3 Papers on Congestion Pricing

Several papers [24-26] describe the advantages of using congestion pricing, varying prices according to load, for allocating limited resources. For operators, congestion pricing can achieve *economic efficiency*, meaning that no user who is being denied of a resource would value it more than those who are currently using it. Congestion pricing can also help operators perform capacity expansion and load balancing. For capacity expansion, if prices are frequently high under congestion pricing, then capacity should be increased because users are willing to pay for better performance. On the other hand, when congestion is detected under flat-rate pricing, it is not clear whether users would actually be willing to pay more for better quality. For load balancing, congestion pricing can entice users to balance load across different nodes through varying prices. From users' perspective, the main benefit of congestion pricing is that they can make tradeoffs between prices and performance. Fluctuation in either prices

or delay is unavoidable in a congested network. Thus those who are willing to pay can obtain good service quality while those who are willing to wait can save money. Congestion pricing also helps users solve the problem of *tragedy of commons*, where people utilize resources regardless of their impacts on others. Congestion pricing takes into account a user's actions on others by using prices to reflect the cost imposed on others. For example, in a packet network, congestion pricing can equate a user's willingness to pay for an additional packet with the marginal increase in the delay generated by that packet. Thus those who are willing to pay can compensate those who are willing to conserve. Other benefits for users are that no artificial performance limit needs to be placed, and no charge needs to be incurred when there is no congestion. Thus with congestion pricing, network operators can gain through improved network performance and increased user satisfaction while users can gain by obtaining services more closely match to their needs.

These papers [24-26] also mention several problems with congestion pricing. For operators, congestion pricing requires a more complicated accounting mechanism and is difficult to implement when locations of congestion can change rapidly. For users, congestion pricing offers less predictable prices and requires users in the control loop. Furthermore, congestion pricing creates opportunities for operators to take advantage of users by artificially introducing congestion or refusing to expand capacity.

2.4 Simulation Studies

2.4.1 At a Single Point

Several simulation studies [28, 31-33] worked with a simple scenario of a single bottleneck to compare dynamic pricing with flat-rate pricing; see Table 2.2 for the details

of the simulation setups in the following four studies. Murphy, Murphy, and MacKie-Mason [28] showed that congestion pricing can allow a network to carry more traffic and that users would value their traffic more according to their utility functions. Peha [33] used packet-level simulations to show that congestion pricing is better than allocating everyone an equal share of the bandwidth when users value their traffic differently. Furthermore, Peha [33] showed that congestion pricing based on connections is almost as effective as congestion pricing based on packets. However, Paschalidis and Tsitsiklis [31] and Patek and Campos-Nanez [32] showed through their simulations that a static pricing policy, like time-of-day pricing, is almost as effective as congestion pricing. These studies reached different conclusions because they are strongly dependent on their workload models and user models. For the workload models, congestion pricing would be more effective if the workload can vary across a wide range. Thus in the first two studies that found congestion pricing to be effective, the workload models employed are bursty video sources. For the user models, congestion pricing would be more effective if users can quickly reduce their usages by a large amount in response to price changes. In the first two studies, there are users who can immediately adjust their loads according to prices. Thus simulations studies strongly depend on the workload models and the user models selected.

Table 2.2: Summary of the simulation setups for dynamic pricing at a single point.

Authors	Network Model	Workload Model	User Model	Metrics
Murphy, Murphy, and MacKie-Mason	One bottleneck. Two service classes, one for video and the other for data.	A fixed number of video applications and a variable number of data applications. Current number of data applications is uniformly distributed. Each application's size is uniformly distributed.	Video users have a utility function that depends on bandwidth. Data users have a utility function that depends on bandwidth and delay. If a user's utility is more than cost, then use the application.	User net utility. Packet loss.
Peha	One bottleneck. Throughput depends on congestion.	Fixed number of on-and-off video sources. Durations of on and off are both exponentially distributed. When on, send at a certain rate.	Each user has a slightly different valuation of usage. If valuation is more than cost, then use the bandwidth.	Value per packet multiplied by throughput.
Paschalidis and Tsisiklis	One bottleneck. A fixed number of service classes. Each class uses a certain amount of bandwidth.	Each service class has a Poisson arrival rate and exponentially distributed duration. Each service class uses a certain amount of bandwidth.	Each user has a uniformly random valuation from a range. If utility is more than cost, then use the bandwidth.	Revenue. User utility.
Patek and Campos-Nanez	One bottleneck shared by a few large institutional users and many small dialup users. Congestion reduces performance for both groups.	Institutional users can send at three different rates with certain transitional probability between the rates, and exponentially distributed duration at each rate. Dialup users can be in either on or off state with exponentially distributed duration at each state. When in the on state, send at a certain rate.	Institutional users get reimbursed for experiencing degraded service. Dialup users send at a Poisson rate that depends on price.	Revenue.

2.4.2 Across Multiple Points

Several papers performed simulations to understand congestion pricing when used for resource allocation and load balancing across multiple points, see Table 2.3 and Table 2.4 for their simulation setups.

For resource allocation across multiple nodes, Waldspurger et al. [41] investigated using auctions to allocate usages of a network of workstations. Each workstation would periodically perform auctions to determine who can use its resources. Using simulations, they found that auctions can achieve equilibrium in price and allocation, adjust to changes, and maintain price difference in accord to the difference between the types of workstation. For computer network resources, Semret et al. [36] investigated using auctions for brokers to maintain stable and consistent service level agreements when offering several classes of service across several domains. In their simulations, each domain has a raw bandwidth seller and a broker responsible for a particular class of service. They used second-price auction for the raw bandwidth seller to sell its bandwidth to all the brokers in its domain. Afterwards, a broker would also use second-price auction to sell its service to users in its domain and to neighboring brokers offering the same service. These auctions are conducted periodically on the order of hours. In simulations, they showed the existence of stable allocation points for brokers. They also showed that the stability of each service class is independent of other service classes. However, they found that oscillation of allocations is possible, and that in equilibrium, some service classes might not be offered because some brokers might decide not to offer them. Instead of using auctions, Fulp et al. [17] modeled a network as competitive markets where each link gradually adjusts its price to match demand to supply. Through simulations, they showed that dynamic pricing used in a distributed fashion can provide high utilization on all links and fair allocation according to user utility functions.

Table 2.3: Summary of the simulation setups for using dynamic pricing to allocate resources across multiple points.

Authors	Network Model	Workload Model	User Model	Metrics
Waldspurger el at.	A group of workstations. Workstations have different speeds.	Several instances of Monte Carlo application. Instances have different funding levels.	Each instance spawns a tree of subtasks. Each task bids all its available funding on the auction with the earliest available time. With remaining funds, each task spawns two more tasks and fund them equally.	Overhead. Fairness according to funding level. Equilibrium of price. Transient behavior of new tasks entering and exiting.
Semret el at.	Three networks. Two services that span multiple networks. One raw capacity seller at each network. One broker per service per network.	30 users per service. Each user can be in either on or off state with exponentially distributed durations at each state. Each user is randomly connected to one of the networks.	Each user has a randomly generated valuation curve. Each user maximizes its valuation subject to price. Each broker maximizes profit.	Stability of allocation.
Fulp and Reeves	A simple topology with several routers.	55 users at different locations of the topology. Each user runs a trace-based video application. Each application starts at a random time from a uniform distribution.	Each user has a utility function indicating valuation for different bandwidth. A user spends all available money across all routers to obtain the best end-to-end performance.	Link utilization. Fairness according to utility. Percentage receiving certain quality.

For load balancing across multiple nodes, Gupta, Stahl and Whinston [18] studied using dynamic pricing to allocate usages across a set of servers. In simulations, they set the price of a server as a function of its load. However, users cared about both the price and the expected network delay to a server. They found that at high utilization levels, congestion pricing can balance loads across servers and increase user benefits. Similarly, Caesar, Balaraman and Ghosal [6] studied various pricing policies for allocating usage across a set of voice-over-IP gateways. They found that when the price of a gateway is a

function of both its load and its distance to a user, then dynamic pricing can provide both low blocking probability and short gateway distances.

Table 2.4: Summary of the simulation setups for using dynamic pricing to balance loads across multiple points.

Authors	Network Model	Workload Model	User Model	Metrics
Gupta, Stahl and Whinston	50 servers at access networks. Different delays to servers. Each server offers four priority classes for running services. 100 services, each service can run on multiple servers.	Users at an access network have a certain request rate for each service of a priority class.	Users have a normally distributed valuation for a request. Users want to minimize cost (price and delay).	User valuation.
Caesar, Balaraman and Ghosal	10 administrative domains. Each domain with an Internet-to-PSTN gateway and a group of users.	Each group of users makes calls with a Poisson arrival rate and exponentially distributed duration.	Half of the users would always want to use the closest gateway. The other half would use the closest gateway costing less than a uniformly distributed value.	Call blocking rate. Distance to gateway.

2.5 Evaluation of Design Space

Several papers [1, 2, 9, 12, 13, 38] discussed a range of design issues for congestion pricing. Estrin and Zhang [13] mentioned that the goal of usage feedback is to realize the benefits of efficient resource utilization according to user valuation while maintaining the benefits of statistical resource sharing. However, they mentioned three problems with usage feedback in a datagram network. First, there is neither resource reservation nor per-user state maintained in a pure datagram network. Second, applications are bursty and have various desired service requirements. Third, the unit of accounting is too small and too difficult for users to comprehend. For feedback channel,

they pointed out that it can be monetary, performance, or administrative. They suggested a combination of the three be used. For feedback policy, they mentioned that it can be based on packet, type-of-service, peak load, or priority. Finally, they listed several design issues to consider for usage feedback.

- Network mechanisms (e.g., first-come-first-serve, Diffserv [4], RSVP [44], etc.).
- Traffic and user accounting granularity (e.g., statistical accounting, per-administrative domain, etc.).
- Feedback frequency (e.g., seconds, hourly, monthly, etc.).
- Cost metrics (e.g., packet, hop, type-of-service, etc.).
- Capacity expansion issues (e.g., when to add, who to charge, etc.).
- Billing and accounting issues (e.g., authentication, authorization, verification, etc.).
- Coordination among providers (e.g., frequency of settlement, unit of accounting, nature of payment, etc.).
- Predictable prices (e.g., cost control, over expenditure, etc.).
- User interface issues (e.g., user acceptance, user involvement, etc.).

For network mechanisms, Edell, McKeown, and Varaiya [12] used real network traces to show that applying traffic smoothing, like a moving average over one second period, on each source can reduce aggregated peak load by 22%. For feedback frequency, Altmann et al. [1] suggested that prices for users can vary on a long time-scale (hourly, daily, or weekly) while rate-control mechanisms for dealing with network bursts can adjust on a short time-scale (seconds or faster). For coordination among providers, Shenker et al. [38] argued for local controls like edge pricing and service level agreements between

peers because global pricing agreement for ensuring end-to-end performance is difficult to achieve. For user interface issues, Danielsen and Weiss [9] advocated that users should specify the most willing to pay instead of specifying performance metrics (like bandwidth, buffer, delay, etc.) or perceived quality (like response time, noise, etc.). With a different perspective, Altmann and Varaiya [2] favored using user agents instead of users to deal with rapid network changes. These works are just some of the papers that deal with the various design issues of congestion pricing.

Other papers [16, 29, 43] performed simulations to evaluate different design space of congestion pricing, see Table 2.5 for their simulation setups. Fitkov-Norris and Khanifar [16] explored how prices should be set as a function of load, whether they should change linearly or exponentially. They found that linear pricing is better for improving revenue and call blocking rate, while exponential pricing is better for ensuring high utilization. After a new price is determined, Xiaowei, Mingquan and Zhenming [43] investigated the issue that different users might receive the new price information at different time. They found that even if users have different round-trip-times to a bottleneck resource, congestion pricing can still allocate the resource according to willingness to pay. After an end node receives a new price update, Neugebauer and McAuley [29] studied how user agents acting on behalf of users should react to the new price, whether they should quickly or slowly adjust usages.

Table 2.5: Summary of the simulation setups to investigate different design space.

Authors	Network Model	Workload Model	User Model	Metrics
Fitkov-Norris and Khanifar	One bottleneck.	Users make calls with certain rate and certain duration.	Rate and duration are both a negative exponential function of price.	Call blocking. Revenue. Network utilization.
Xiaowei, Mingquan and Zhenming	One bottleneck.	A fixed number of users.	Each user has a log-based utility function based on bandwidth sent. Each user maximizes utility minus price.	Fairness based on utility.
Neugebauer and McAuley	One bottleneck. A surcharge when congested.	A fixed number of users. Each user requests a certain amount of bottleneck resource.	Various strategies based on the most a user is willing to pay.	Stability of allocation.

2.6 Prototypes

For prototyping efforts, Stiller et al. [39] implemented an IP-telephony charging scheme based on performing auctions at each router on a path. They used RSVP [44] signaling to send bid requests and acquire price information. The price of a path would equal to the sum of all the prices charged at all the routers along the path. The price would then be valid until the next reservation period. Instead of using auctions, Fankhauser, Stiller, and Blattner [14] allowed each link to determine its price when prototyping a testbed network offering several classes of service. They used RSVP [44] to dynamic acquire price information at each router. Using a similar design, Wang and Schulzrinne [42] implemented congestion pricing in real routers and assigned simulated users with utility functions to show that congestion pricing can share a network's bandwidth fairly between users according to user utility functions. Table 2.6 summarizes their simulation setup. These implementation efforts show that it is feasible to incorporate congestion pricing in real systems.

Table 2.6: Summary of the simulation setup to verify a dynamic pricing implementation.

Authors	Network Model	Workload Model	User Model	Metrics
Wang and Schulzrinne	One bottleneck.	A fixed number of users. Each user sends at a certain rate.	Each user has a utility function that depends on sending rate. Each user maximizes utility function minus price.	Fairness based on utility.

2.7 User Experiments

Klausz, Croson and Croson [22] conducted simulated games to understand the effectiveness of using auctions for allocating modem usages. They assigned a small group of users different utility functions, some would indicate that users would be better off using modems during peak hours while others would indicate that users would be indifferent regarding the time of day. They limited each user to a certain number of tokens. Using this setup, they asked the users to participate in repeated auctions for modem usages. They found that auctions can improve utilization and blocking rate by smoothing out demand. Furthermore, they found that auctions can increase the utility of all types of users.

The INDEX project [10, 11, 35] performed an extensive user trial to understand the effectiveness of usage-based pricing. It recruited 80 users to measure their demands for bandwidth when using ISDN lines from home. The trial offered users several bandwidth choices with fixed prices and allowed them to change their choices at any time. It discovered that different policies, by volume or by connect time, can have a large impact on demand. For each pricing policy, it found that the demand is very sensitive to prices and that the difference among users is persistent and large. Furthermore, it noticed that users can quickly adapt to a new pricing policy. These findings for usage-based pricing are encouraging for conducting research on dynamic pricing.

2.8 Summary

The state-of-the-art on congestion pricing consists of papers [24-26] that analyze its advantages and disadvantages. Using simulations, other papers [28, 31-33] have shown that congestion pricing can be better or worse than flat-rate pricing depending on the workload model and the user model used. Congestion pricing would be more effective if the workload model is bursty and the user model enables rapid load adjustments to prices. Other simulations [17, 36, 41] have investigated the dynamics of applying congestion pricing distributely across multiple locations. With regard to implementation efforts, there are discussions and evaluations [1, 2, 9, 12, 13, 16, 29, 38, 43] of different design issues. There are also prototyping efforts [14, 39, 42] showing that it is feasible to implement congestion pricing in real systems. However, there are very few user studies [10, 11, 22, 35] that investigate the effectiveness of using pricing. In sum, simulation studies of congestion pricing strongly depend on the workload models and the user models used. There is little known about the practical engineering issues like how to apply congestion pricing to users and how to manage congestion pricing in real systems.

Operators need to understand more about the practical issues and the tradeoffs (e.g., expected reduction in provisioning, expected frequency of price changes, etc.) of congestion pricing before they are willing to use it in their networks. Thus more deployment efforts are needed to design the appropriate service, pricing scheme, and user interface, and to verify the feasibility, performance gains, and user acceptance of congestion pricing. With deployments of effective schemes, we can also use the results to formulate realistic user models for simulation studies. Thus the next step for congestion

pricing research is to deploy real testbeds with real users to examine the issues and the tradeoffs involved. In the next chapter, we will discuss the methodologies and the testbeds for conducting our user trials of applying congestion pricing to voice and data traffic.

Chapter 3 Methodologies and Testbeds

From surveying the related work on using pricing as a mechanism for resource allocation, more user evaluations are needed to understand the practical issues and the tradeoffs of applying congestion pricing in real systems. We would like to conduct user evaluations for both voice and data traffic. For voice, a typical operator has at least thousands of customers sharing a bottleneck resource like a central office. Thus, we need a voice testbed involving thousands to evaluate the tradeoffs of congestion pricing under the appropriate scale. However, arranging a large-scale testbed is nearly impossible. One approach is to find a voice operator willing to let researchers experiment with his/her customers and networks, but operators are justifiably leery of such a proposition. So researchers are left with the option of developing their own voice services. To develop a service for thousands of users is both expensive and time-consuming. But a more manageable goal is to develop a service for a small group of users, like 100 people. Thus with a small-scale user study, the challenge is to propose a methodology that can make the results convincing when scaled to thousands of users. For data traffic, the scaling issue is less of a concern because many instances of usage have a smaller group of users sharing a bottleneck resource like an access point. Nonetheless, finding a testbed and scaling the results from a small-scale study still pose challenges. However, the real challenge for data traffic is that users are not used to dealing directly with dynamic prices. Thus finding a scheme that is acceptable to users and effective for operators becomes the challenge. In sum, for voice traffic, the main challenge for congestion

pricing is to evaluate its performance under large-scale; and for data traffic, the main challenge is to investigate its acceptance by users.

For this chapter, in Section 3.1, we first discuss how we measure effectiveness and acceptance of a scheme. In Section 3.2 and 3.3, we present our methodology and testbed for tackling the challenge for voice traffic. In Section 3.4 and 3.5, we present our methodology and testbed for data traffic. In Section 3.6, we point out where the methodologies and testbeds are applied in the rest of this thesis.

3.1 Metrics for Effectiveness and Acceptance

We would like to measure how effective congestion pricing is for improving system performance and how acceptable it is for users. For effectiveness, we first measure how congestion pricing can affect user behavior. For voice calls, we measure effectiveness by how able changing prices can cause users to talk shorter, talk at another time, or talk using a lower quality. For data traffic, we measure effectiveness by how well dynamic pricing can entice users to accept more delay, jitter, or loss in their traffic. After measuring user reaction to price changes, we then measure effectiveness by the overall system improvement in performance and capacity. Ideally, a system would have a minimal capacity that is highly utilized without any congestion. For voice calls, we measure performance by call blocking rate. For data traffic, we measure performance by burstiness, with less burstiness indicating that less packets are being dropped and delayed. For user acceptance, we use both objective and subjective measurements. For objective measures, we measure how frequently prices need to change or how frequently users need to make a decision. For subjective measures, we use interviews, focus groups, and surveys, to understand the acceptability of congestion pricing (e.g., how users like

the overall scheme and how they feel about the price changes) and how much incentive is required to entice users to choose congestion pricing over flat-rate pricing. With these metrics for effectiveness and acceptance, we can then better understand the practical issues and the tradeoffs of applying congestion pricing to users.

3.2 Methodology for Voice

To understand the tradeoffs of congestion pricing for thousands of users using a small-scale user study, ideally, we would like users to react to price changes of a large-scale service. To start, we first use a small group of users to understand the feasibility and user acceptance of different congestion pricing schemes. After fine-tuning an appropriate scheme, we then conduct further small-scale user experiments to measure user reaction to price changes. In these experiments, prices are set using certain heuristics. With the experimental results, we can derive a user model for performing large-scale simulations to understand how an operator should manage congestion pricing and the tradeoffs it would face. The simulation results will strongly depend on the user model and its parameters, which in turn depend on the user experiments when the prices are set artificially. However, with a user model and an understanding of the operator behavior, we can emulate a large-scale service. With it, we can conduct further small-scale user experiments where users are reacting to the price changes of a large-scale service. Thus, this approach verifies the user model by re-measuring user response in as realistic setting to a large-scale system as possible while only requiring a small-scale user study. However, one drawback is that the small group of users might not represent the general public. However, the results from the small group can give us better insight into how the general public would respond and react to dynamic pricing.

The above approach can be formalized as a four-step methodology shown in Figure 3.1. In the first step, we develop a voice service that can attract a large group of users to use it over a long period of time. With the service and the user base, we can then explore the complexity and the overhead of using congestion pricing.

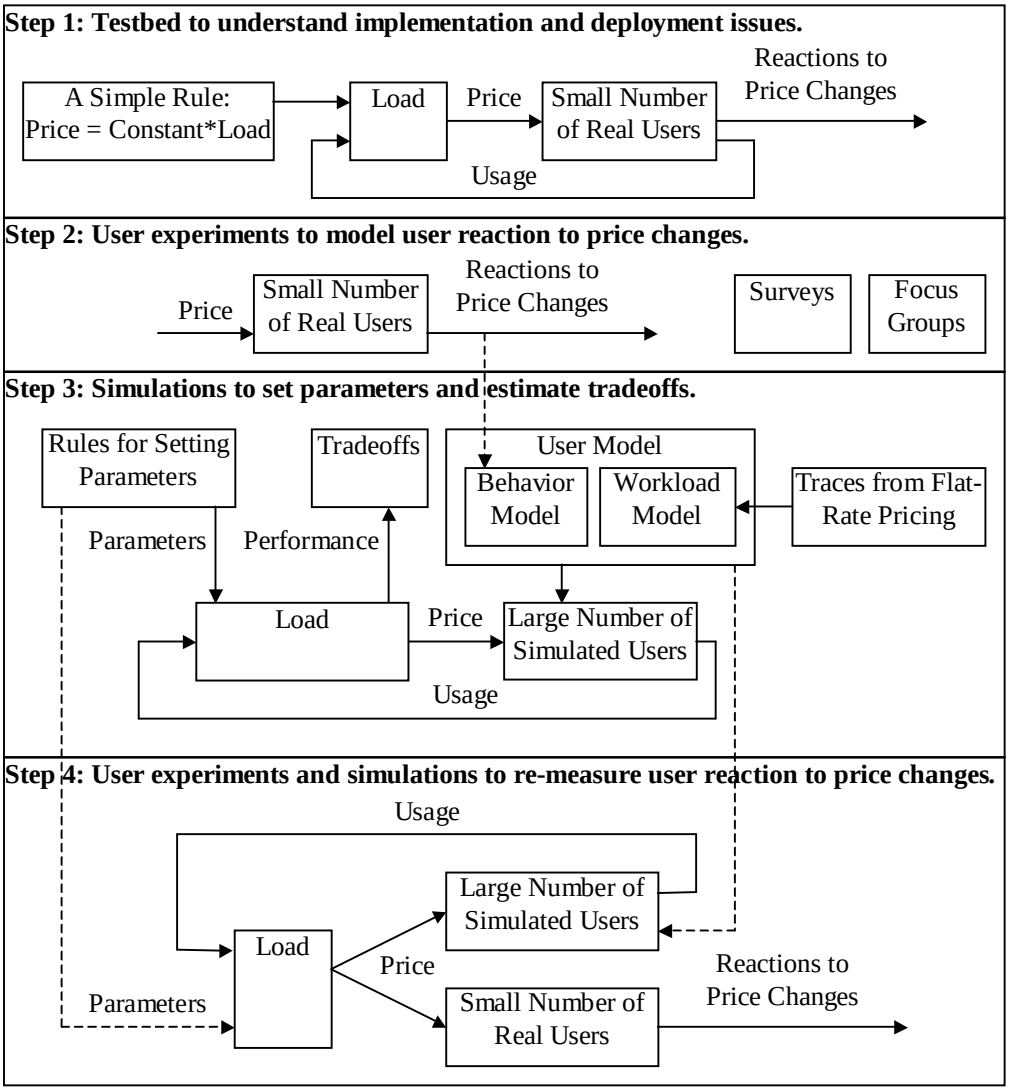

Figure 3.1: A four-step methodology for evaluating the scaling issues of congestion pricing.

In the second step, we deploy the service to users for conducting pricing experiments to observe how users would react to price changes during phone calls. With

users, we experiment with different ways of applying congestion pricing, measure user sensitivity to price increases, and use focus groups and surveys to further understand user acceptance to congestion pricing.

After understanding user response to price changes, in the third step, we perform simulations to measure the effectiveness of congestion pricing when there are many users using the service. In the simulations, we use traces from flat-rate pricing to generate a user's workload model and observations from the user experiments to model his/her response to price changes. We first determine the optimal parameter settings an operator should use when applying congestion pricing to thousands of users. Then using the appropriate settings, we estimate the potential benefits and drawbacks (e.g., improvement in system performance and reduction in user satisfaction) of congestion pricing.

Our estimates strongly depend on the form and the parameters of the user model that is derived when the prices are set using a heuristic, so in the fourth step, we re-measure the user model by having a new group of users test the service along with a large number of simulated users. The simulated users would make calls and react to price changes according to the model, the operator of the service would use the appropriate parameters to manage congestion pricing, and we would observe real users' reactions to price changes under this setting (e.g., how likely a user would terminate his/her call after a price increase). Thus, if users' reactions match that of the model, then we can be more confident of the estimates based on the model.

3.3 Testbed for Voice

The architecture of our voice testbed, a voice-over-IP gateway service, is shown in Figure 3.2. It is based on the H.323 protocol [21] and uses a Motorola Vanguard 6560

as a H.323 gateway for connecting the Internet to the PSTN. The gateway has a Primary Rate ISDN line connection to the PSTN for supporting 23 simultaneous calls. The service is then built on top of an H.323 proxy (for the gateway) for performing functions like monitoring, admission control, accounting, and price setting. Users on their computers then interact with a web browser in conjuncture with a H.323 client like the Microsoft NetMeeting to make calls through the gateway via the proxy. See Figure 3.3 for the web interface.

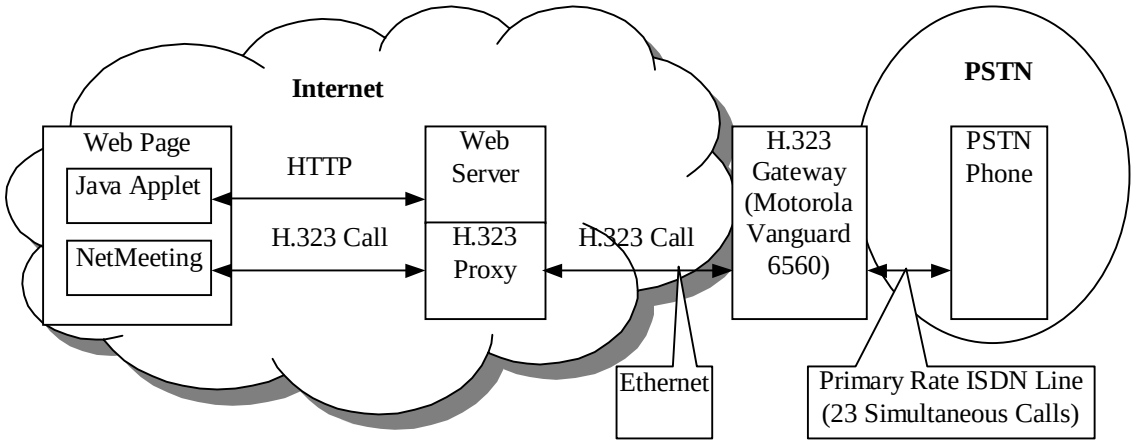

Figure 3.2: Architecture for a voice-over-IP gateway service.

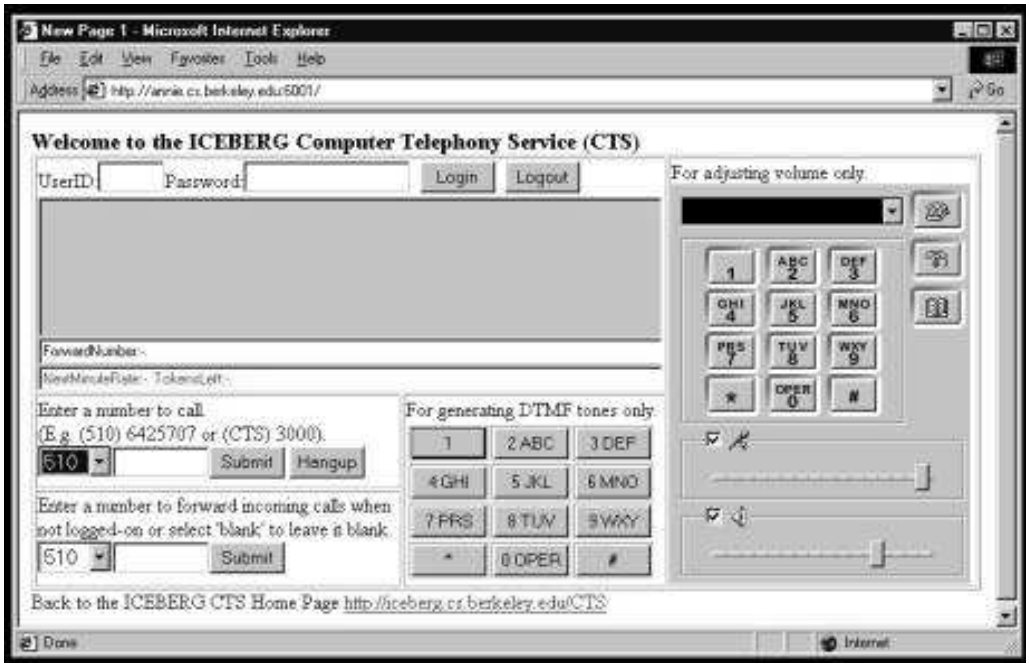

Figure 3.3: Web interface for making and receiving phone calls.

Users can invoke the service to make and receive phone calls from their computers or phones. From a computer, users access a web page, log in, and then enter the numbers they wish to call. When originating their calls from a computer, users are able to see real-time pricing information, like the current rate, the call duration, and the money left, presented on the web page. When accessing the service via a phone, users can call a phone number, enter their user IDs and PINs, and then enter the numbers they want to reach. The phone interface is very similar to using a calling card. When using a phone, users will hear the current price at the beginning of a call and whenever the price changes. Users can also use the service to receive incoming calls from any computer or phone and then redirect them to their computers or phones as they choose.

In Chapter 4, we will describe in more detail how the service is implemented and deployed. With regard to implementing congestion pricing, we found that there are only

two modifications needed for adding congestion pricing to an existing flat-rate priced service. First we needed to modify the accounting mechanism on the H.323 proxy to generate an accounting record every charging period instead of every call to support dynamic prices. Second, we needed to modify the proxy to provide real-time pricing information to users either as updates on their computers' web pages or as inserted messages in their phone calls. Thus the complexity and the overhead of implementing congestion pricing on top of a voice service is minimal.

3.4 Methodology for Data

Our approach for understanding how congestion pricing can best be applied for data traffic employ a methodology of iterative prototyping, evaluation, and analysis (see Figure 3.4). Since there is no existing congestion pricing scheme to analyze, we first quickly prototype a scheme and deploy it to a small group of users. After understanding the issues involved, we use simulations to analyze the tradeoffs of different congestion pricing schemes when there are more users involved. We then repeat the cycle by prototyping a new scheme, deploying it to users, and analyzing possible improvements. Thus we use user studies to understand user reaction and acceptance, and simulations to understand scaling issues. We will describe in more detail the prototypes, the user experiments, and the simulations in Chapter 7.

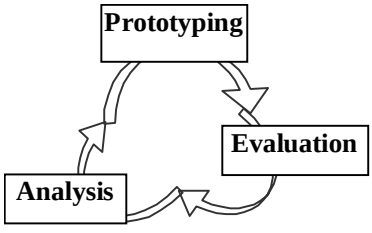

Figure 3.4: Methodology for applying congestion pricing to data traffic.

3.5 Testbed for Data

The data testbed for allocating bandwidth at a 100Mbps LAN is shown in Figure 3.5. The bottleneck is the access link to the Internet. The testbed has two components. First, a Packeteer PacketShaper, a commercial traffic shaping appliance, that can monitor and shape all incoming and outgoing flows through it[3]. It is placed at the access link to alleviate the bandwidth mismatch between the larger load on the LAN and the smaller connection to the Internet. Thus, it can dynamically apply different rate limits for the incoming and the outgoing traffic to each computer, an IP address, on the LAN. The second component, a proxy for the PacketShaper, provides users with a web interface and performs accounting functionalities. It uses a Java applet for users to receive real-time pricing and usage information, and to send purchasing commands (see Figure 3.6). In turn, it acquires real-time usage information and issues bandwidth setting commands to the PacketShaper. The proxy is placed outside the LAN so that when the LAN is congested, the PacketShaper can restrict the LAN traffic so as to guarantee a certain amount of bandwidth for the control traffic to the proxy. If the proxy is inside the LAN, then the traffic in the LAN can easily overwhelm the control traffic to the proxy and cause the proxy to be unresponsive to user commands.

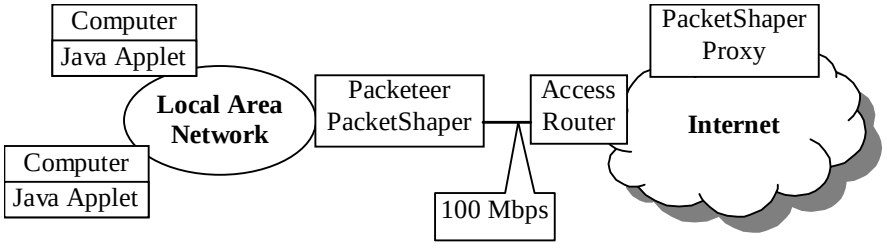

Figure 3.5: Architecture for applying congestion pricing to a LAN.

[3] See http://www.packeteer.com for more information.

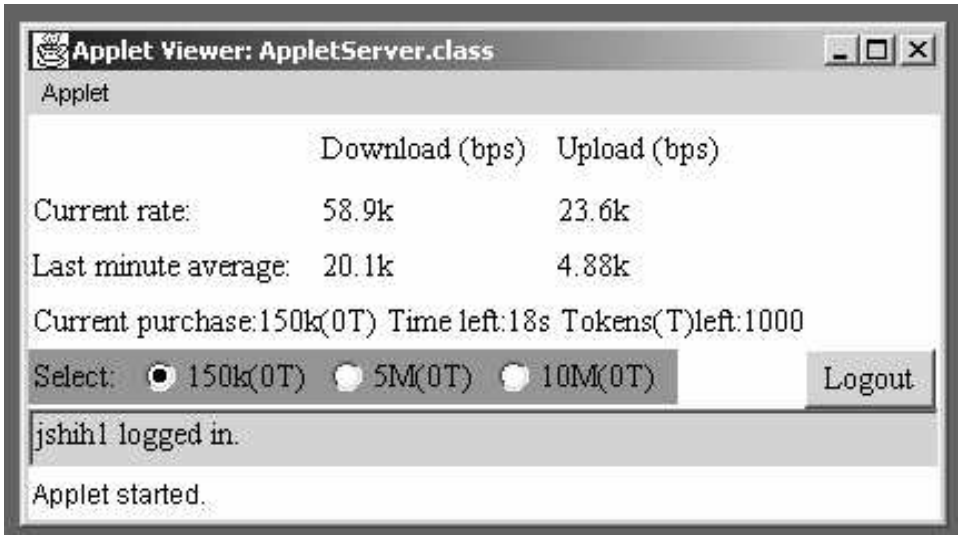

Figure 3.6: User interface for requesting different bandwidth.

When the proxy starts, it first establishes a permanent telnet connection to the PacketShaper. Through the connection, the proxy periodically polls the PacketShaper to obtain the current access link usage by each computer on the LAN. The frequency is at most once every few seconds because of the overhead in the PacketShaper in reporting the usage information. When a user on a LAN-attached computer wants to purchase more bandwidth, he/she would first use a web browser to download a Java applet from the proxy. Through the applet, the user would see his/her current bandwidth usage and the current prices. When the user makes a purchase through the proxy, the proxy first performs accounting and then sends a command to the PacketShaper to adjust the bandwidth limits on that user' computer.

Using only two components, the proxy and the PacketShaper, the tesbed was easy to build and deploy. The proxy was easy to implement because the PacketShaper provides a telnet connection for interfacing with it. To deploy the components, we were able to obtain permission from our system administrators in part because the

PacketShaper appliance has many built-in safety features to alleviate their concerns like outages and security. Thus with the availability of a commercial traffic shaping appliance, implementing and deploying a testbed to evaluate congestion pricing for data traffic becomes simple.

3.6 Roadmap

In Chapter 4 through Chapter 6, we describe applying the four-step methodology for evaluating the scaling issues of congestion pricing for voice traffic. In Chapter 4, as the first step of the methodology, we describe our effort in implementing and deploying a voice-over-IP gateway service. After deploying the service and building a user community, in Chapter 5, we utilize them to conduct pricing experiments for understanding user reaction to price changes. In Chapter 6, we apply the third and the fourth step to evaluate and verify the results of congestion pricing when applied to thousands of users. For data traffic, in Chapter 7, we report how we use the iterative process of prototyping, evaluation, and analysis to discover a congestion pricing scheme that is acceptable to users and effective for operators.

Chapter 4 Voice Traffic Testbed Development

To use a voice-over-IP gateway service for investigating congestion pricing, we needed to first develop a service that can attract a large number of users who will continue to use it for the duration of our experiments stretching across several months. The best way to attract users is to incorporate desirable features that users want. Perhaps the most overriding consideration in keeping users is to deploy and maintain a reliable service. Once the user community is well established, to study congestion pricing, we needed to quickly introduce various pricing policies. We found that we can rapidly prototype new functions and reliably deploy them within our voice testbed by modeling a voice call as a simple four-state Finite State Machine (FSM). We then only needed to add control logic at these four states to implement new functions. Using this general FSM, we could rapidly prototype a service, deploy it to users, and then quickly evolve the service for the next deployment. Through three successive prototypes, basic, intermediate, and final, we gradually incorporated functions like admission control, accounting, call redirection, and handoff. After the third version, we were able to sign up and retain 100 users who have actively used our service.

In Section 4.1, we first provide some background on the H.323 protocol. In Section 4.2, we explain our main design decision of using a H.323 proxy to implement new functionalities. In Section 4.3, we describe the proxy architecture and the use of the four-state FSM. In Sections 4.4 to 4.6, we illustrate using the FSM to rapidly prototype various features by describing the three iterations of development and deployment.

Finally, in Section 4.7, we conclude with the lessons learned from the voice-over-IP gateway testbed development.

4.1 Background on H.323

The International Telecommunication Union (ITU) H.323 protocol [21] consists of five components, all shown in Figure 4.1. Two of the components, the H.323 gateway and the H.323 client, are required while the other three, the H.323 gatekeeper, the Multipoint Control Unit (MCU), and the Multipoint Processor (MP), are optional. The H.323 gateway is used to connect the Internet with the PSTN. The H.323 client is used to make and receive telephone calls through the gateway. The H.323 gatekeeper is an optional agent on the Internet that can perform management functions like admission control and address translation. Finally, the MCU and the MP are optional components for supporting conference calls.

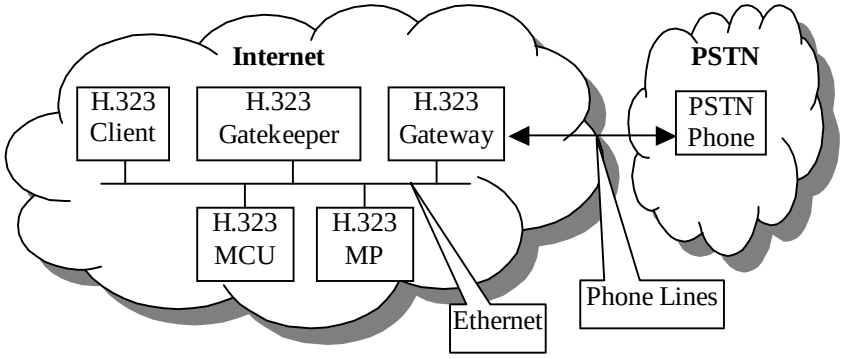

Figure 4.1: Components of the H.323 architecture.

4.2 Design Decisions

Our first design decision was to use the H.323 protocol to develop a voice-over-IP gateway service. We chose H.323 instead of other protocols like SIP [19] because in

1998, when we were developing the service, H.323 was the dominant standard for IP-telephony. Many vendors were selling H.323 gateways and H.323 protocol stacks for developing computer-telephony applications. Protocol stacks are software libraries that allow developers to program with high-level interfaces instead of low-level H.323 bits when communicating between H.323 components. Furthermore, Microsoft NetMeeting, a multimedia collaboration tool, is a H.323 client that is freely available on all Windows platforms. Thus using H.323 allowed us to quickly develop and deploy a service.

Our next design decision was to use an application level proxy to implement basic control functions like admission control, accounting, and call redirection. The proxy (see Figure 4.2) sits infront of a H.323 gateway and breaks a H.323 call in two. For example, when a user makes a call from a H.323 client, he/she would first make a H.323 call to the proxy. The proxy would then perform the control functions before placing another H.323 call through the gateway. We used a proxy because it allows us to easily add new control functionalities without changing end points like clients and gateways. If we had followed the H.323 protocol which recommends adding control functionalities at a H.323 gatekeeper as part of a single H.323 call (see Figure 4.3), we would either be restricted to the functions defined for a gatekeeper or we would need to change the H.323 signaling and the end points when adding new features. Thus a proxy approach provides simplicity and flexibility at the cost of efficiency when having to deal with two calls.

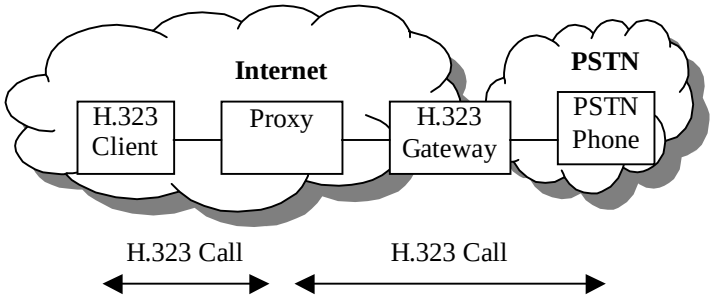

Figure 4.2: View of using an application level proxy.

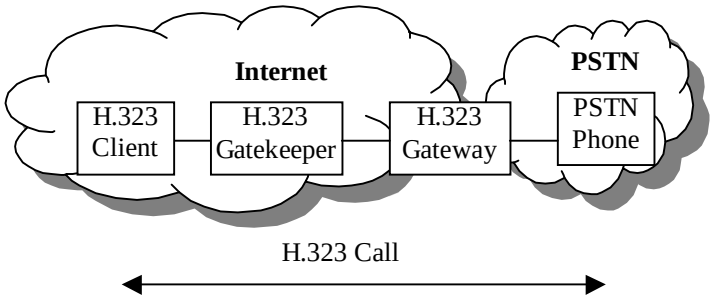

Figure 4.3: View of using a H.323 gatekeeper.

4.3 Proxy Architecture

The structure of the proxy (see Figure 4.4) consists of a C++ program and a H.323 protocol stack from Lucent Elemedia[4]. The structure is based on a sample H.323 client program that came with the Elemedia H.323 protocol stack. The C++ program contains two types of thread, a Main thread and a Call thread. The Main thread is responsible for managing global resources like IP ports assignment. When the Main thread wants to make a call, it creates a Call thread to initiate and handle a new H.323 call. The Call thread then interacts with the H.323 protocol stack to send and receive H.323 messages. Similarly, when the H.323 protocol stack receives an incoming call, it first notifies the Main thread. Afterwards, the Main thread creates a new Call Thread to handle the

[4] See http://www.elemedia.com for more information.

incoming call. Thus using the protocol stack makes developing H.323 components like a proxy manageable.

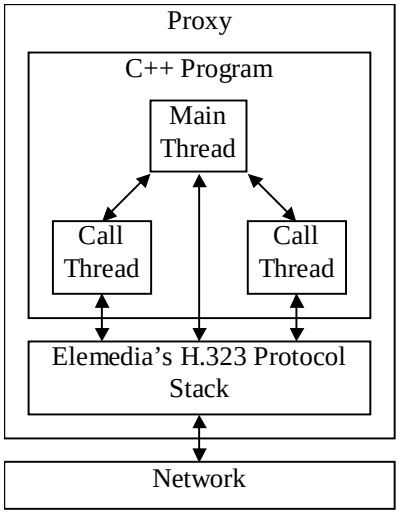

Figure 4.4: Structure of the H.323 proxy.

To allow rapid prototyping, we found that we can abstract the complicated H.323 call interface provided by the protocol stack into a simple four-state FSM. We could then quickly implement various features in a proxy by adding control logic in each state. The four states of an abstracted H.323 call are shown in Figure 4.5. The first state, for performing initialization, is just right after a Call thread has started handling a new call. The second state, for performing control functionalities, is right after the Call thread has established both the control and the data channel of the H.323 call, but before any audio packet is sent and received. The third state, for performing monitoring functionalities, is when audio packets are sent and received. Finally, the fourth state, for performing cleanup, is right after a call has terminated. We will illustrate how we used the four-state FSM abstraction to implement various computer-telephony features when describing the three prototyping efforts next.

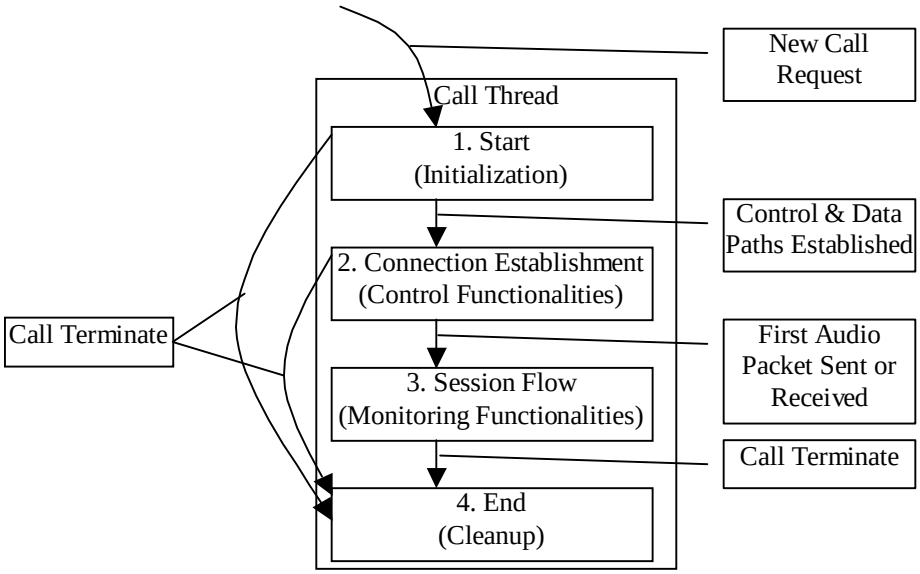

Figure 4.5: A four-state FSM model of a H.323 call.

4.4 First Prototype

4.4.1 Requirements

For the first prototype, we wanted to quickly deploy a minimal service to people in our research group to understand the issues of attracting users and conducting pricing experiments. As a minimum, we needed to implement outgoing calls, incoming calls, and accounting. For outgoing calls, we could easily allow users to use any computer to make calls. Thus, we would also like to allow users to use any computer to receive calls. However, a typical user in our group has more than one computer. Thus, we needed to dynamically update users' current computer locations to support incoming calls. Our initial solution for location update is to simply allow users to dynamically specify which computers for receiving their calls. For accounting, we needed it to experiment with different pricing policies. Our approach for accounting is to limit each user to a certain number of tokens and charge a certain token rate per minute. With location update and accounting, we aimed to deliver a basic service to users.

4.4.2 Architecture

Our architecture (see Figure 4.6) consists of a Microsoft NetMeeting as the H.323 client, a proxy, and a Motorola H.323 gateway. The NetMeeting's user interface is shown in Figure 4.7. The gateway initially has two phone lines connected to the PSTN. Through the proxy, users can use the gateway to make and receive calls between the Internet and the PSTN.

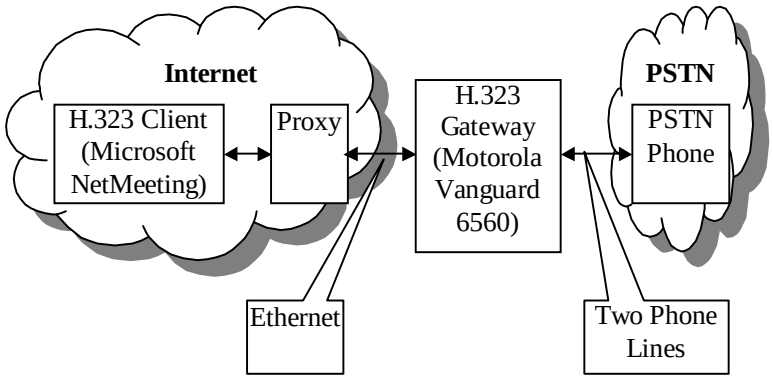

Figure 4.6: First prototype's architecture.

Figure 4.7: NetMeeting's user interface.

When a user wants to make a call from his/her computer, she/he would need to run NetMeeting and then enter as the phone number his/her user ID and the callee's number. When the proxy receives the call from the NetMeeting, it will use the user ID for admission control and accounting, and then use the callee's number to generate another H.323 call to the gateway to reach the callee. When the callee answers, the proxy will then connect the two H.323 calls by asking each call to forward its incoming audio packets to the other call's end point. To receive incoming calls on a new computer location, a user would need to first perform a location update by encoding his/her user ID

and the new IP address as the phone number. When the proxy receives the call, it will update the user's computer location. Later on, when the proxy receives an incoming H.323 call (from the PSTN) for that user, it will issue another H.323 call to the user's computer. A NetMeeting demon running on the computer will prompt the user to answer. After answering, the proxy will then connect the two H.323 calls together. Thus it is relatively straightforward to redirect calls using a proxy.

4.4.3 Proxy Implementation Details

We found that we can easily connect two H.323 calls together in a proxy by adding a small amount of control logic at each state of a simple four-state FSM call (see Figure 4.8). When the proxy first receives an incoming call from either a gateway or a client, the handling Call thread first obtains its call identifier at the first state. Then at the second state, the handling Call thread performs admission control and accounting[5], and then asks the Main thread to generate a second Call thread to complete the call. When the second Call thread reaches its first state, it obtains its call identifier, associates the two Call threads together with their call identifiers, and then performs accounting for its call. After the second Call thread reaches its second state, it uses the call identifiers to setup both Call Threads to forward incoming audio packets to each other's end point. Finally, when one Call thread terminates, it first performs accounting on its call, and then asks the Main thread to cause the other Call thread to terminate. All accounting information relevant to each call are kept in its Call thread and all data relevant to each user, like the current computer location and the number of tokens that the user has left, are kept in the Main thread.

[5] Accounting is performed by keeping accounting states with each Call thread and then once every minute generating an accounting record based on the states.

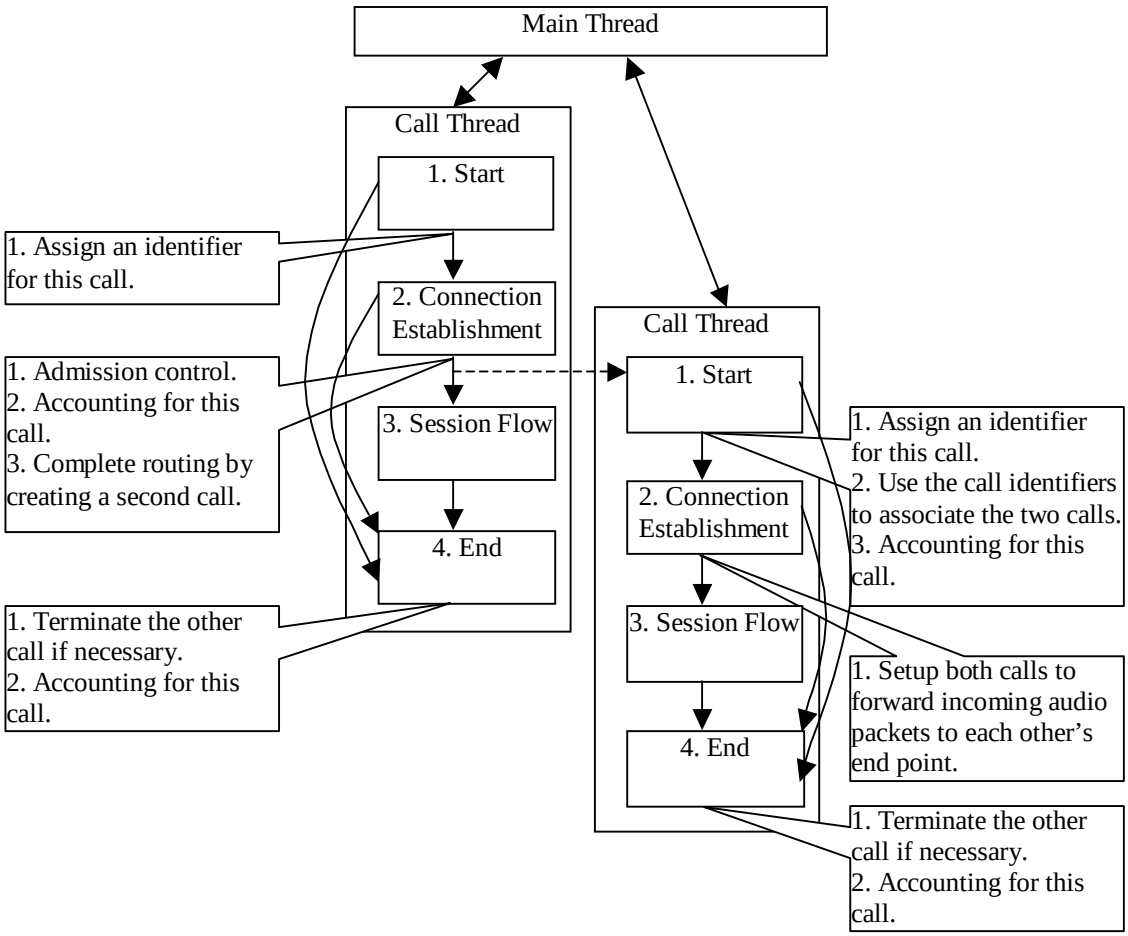

Figure 4.8: Implementation of the first prototype using the four-state FSM.

4.4.4 Deployment

We offered the service to 12 people in our research group during the Fall of 1999[6]. We assigned each user a four-digit user ID for making and receiving phone calls. We limited the outgoing calls to local numbers in the Berkeley area. To receive an incoming call, callers can call one of the two phone lines for the gateway and then enter the user ID of the user they want to reach. We created a simple directory service by placing users' IDs on a web page so that people in our group can call each other.

[6] Our users have shared PSTN phones in their offices that can call anywhere.

We obtained lots of usage during the first week because users were curious. However, during the second week, usage quickly declined. We tried to encourage users to make more phone calls during the third week; however, usage remained low the following weeks. See Figure 4.9 for the weekly usage chart.

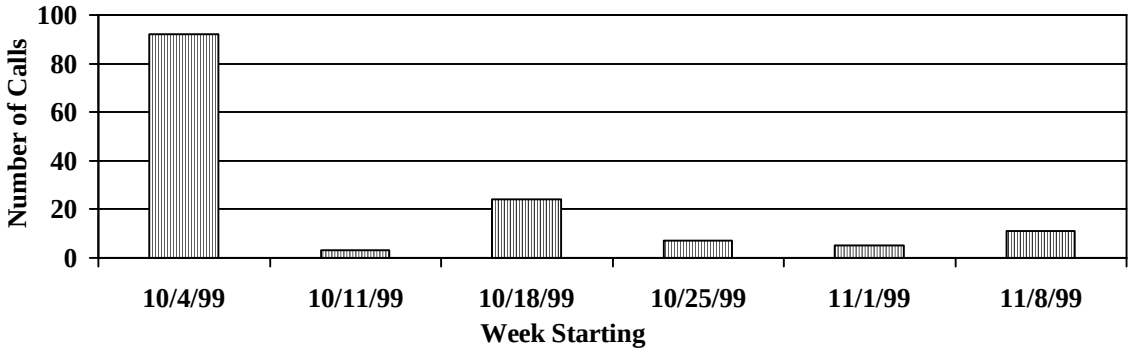

Figure 4.9: Weekly usage of the first deployment.

4.4.5 Lessons

We found that we can abstract the complicated H.323 call into a simple four-state FSM and quickly implement functions like admission control, accounting, and call redirection in a proxy by adding control logic at each state. Using a simple four-state FSM also made debugging and testing the proxy simple.

The service was easy to deploy and maintain. To deploy it, we only needed to ask users to download and install NetMeeting on their computers. To maintain it, we only needed to reboot the proxy machine once a week for precaution. Otherwise, the proxy was available all the time. The service was reliable because the four-state FSM abstraction allowed us to reliably add and test new functionalities.

This deployment experience taught us that it is not easy to convince users to use their computers instead of phones. Based on surveys, users stated that they stop using the

service because they need to use a headphone, run NetMeeting, and adjust sound before they can make a call. Users indicated that they are hesitant to use the service for receiving incoming calls because they need to give out new phone numbers. Finally, with location update, users found that it is hard to remember to update their computer locations whenever they switch computers.

4.5 Second Prototype

4.5.1 Requirements

The goal of the second prototype was to attract a larger user community. We decided to target the undergraduates in our department because they only have access to pay phones when they are in school. We believed that it would be easier to build a user community for those who do not have easy access to phones. Based on the lessons learned in the first prototype (and to attract and retain more users), we decided to support the following features to make the service more usable. For outgoing calls, we wanted to allow users to use any computer in the undergraduate computer cluster. After logging in with a password, they can enter either phone numbers or user IDs of the people they want to call. To increase incoming calls, we wanted to forward calls to users' computers if they are logged on, and forward them to phone numbers chosen by users otherwise. Using logins to indicate where the users are, we eliminated the issue of requiring users to dynamically specify which computers for receiving their calls. Forwarding incoming calls to phone numbers when users are not logged on saves us the work of implementing a voice mail service. During a call, we would like users to be able to send DTMF tones[7] from their computers to check their answering machines for messages. Furthermore, we

[7] DTMF tones are touch tones on phones that allow users to dial numbers and send & receive control tones during a call.

would like to provide users on their computers with real-time pricing and accounting information. With these features, we hoped that the undergraduates in our department would find the service useful.

4.5.2 Architecture

Our architecture now consists of a web page, a web server, a proxy, and the Motorola gateway (see Figure 4.10). We upgraded the Motorola gateway with a Primary Rate ISDN line that can support 23 simultaneous calls between the Internet and the PSTN. For incoming calls, we setup a hunt group for the 23 lines so that callers on the PSTN only need to dial one phone number to locate an available line. For the user interface, since NetMeeting does not support text display or DTMF, we decided to use a web page (see Figure 4.11) to provide users with better and more flexible inputs and outputs, like real-time prices and DTMF. To simplify development, we reused NetMeeting as the H.323 client by incorporating it inside the web page.

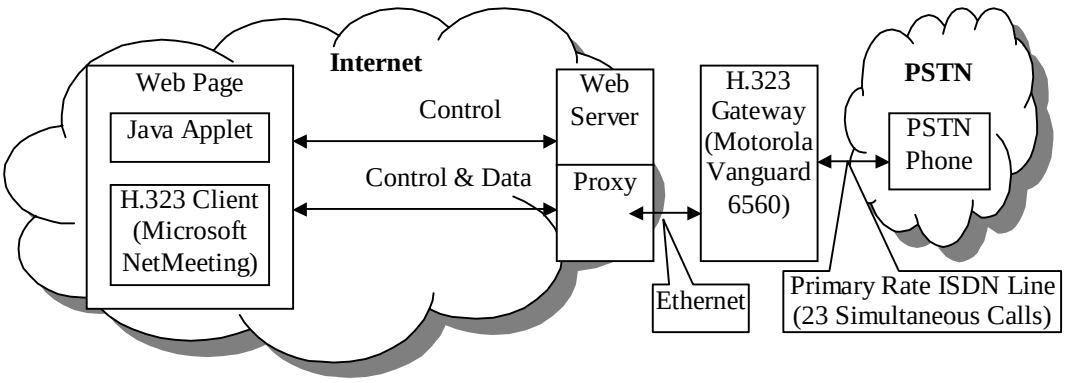

Figure 4.10: Second prototype's architecture.

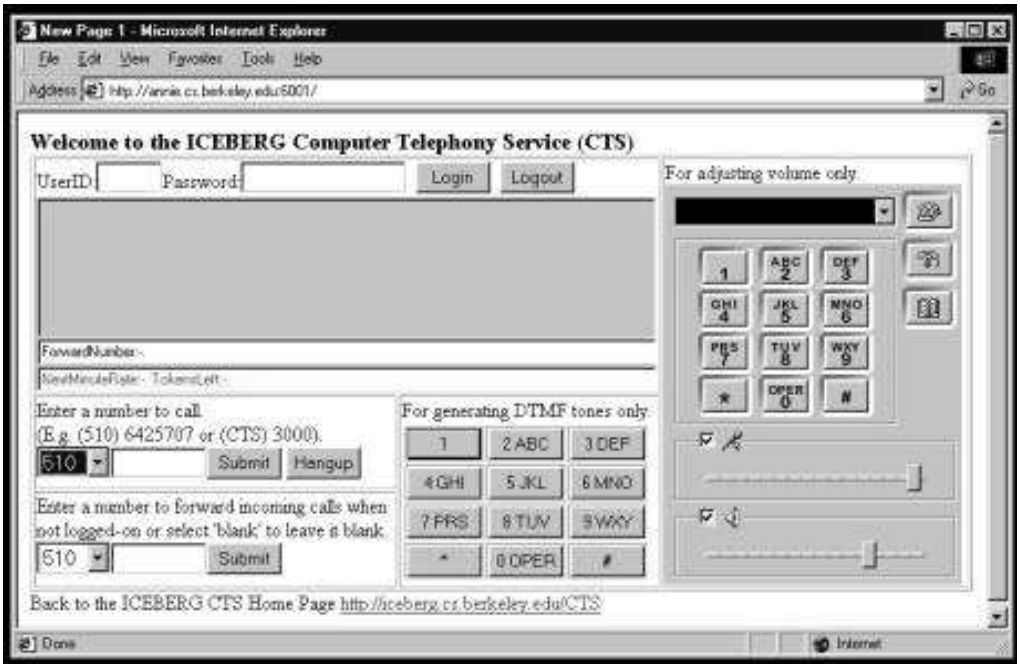

Figure 4.11: Second prototype's web interface.

When a user wants to use a computer to make a call, he/she first needs to login through the web page using his/her user ID and password. The web server will then pass the login request to the proxy. After the proxy authenticates the user, it will generate a cookie and ask the web server to pass it back to the web page for storage. When the user wants to make a call, he/she would enter the phone number or the user ID of the person she/he wants to reach. The web page would then combine the cookie with the number when using NetMeeting to make a H.323 call to the proxy. When the proxy receives the call, it uses the cookie for admission control and accounting, and uses the number to issue a second H.323 call to the callee. When the callee answers, the proxy then connects the two H.323 calls together. For incoming calls, the proxy will forward the calls to users' computers if they are logged on. If not, the proxy will forward them to PSTN numbers chosen by users. During a call, users on their computers can send and receive DTMF

tones. When the proxy receives a DTMF request from a web page via the web server, it will generate a DTMF tone to the other end point. Similarly, when the proxy receives a DTMF tone from the other end point, it will ask the web server to forward it as a text message to the web page. During a call, the proxy will periodically ask the web server to display status messages like the current price information on the web page. Thus the web interface acts as an out-of-band control channel complementing the H.323 protocol.

4.5.3 Proxy Implementation Details

Only one additional thread, the WebServer thread, needs to be added in the proxy software to interface with all the web pages (see Figure 4.12). When the WebServer thread receives control messages from the web pages, it can forward them to the Main thread or the Call threads. Similarly, when the WebServer thread receives control messages from the Main thread or the Call threads, it can forward them to the appropriate web pages.

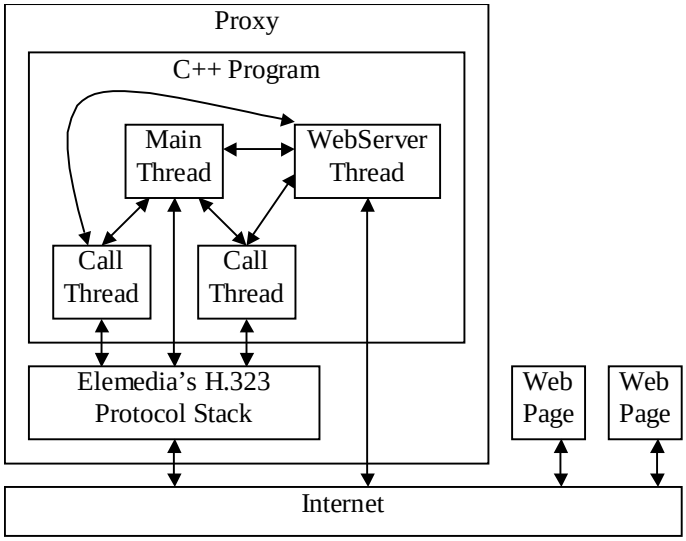

Figure 4.12: Program structure of the H.323 proxy for the second prototype.

To support interaction with the WebServer thread, the control logic of the four-state FSM used by the Call thread only needs to be modified at two places (see Figure 4.13). First, after the Call thread reaches the first state, it needs to ask the Main thread to associate its call ID with the caller's or callee's user ID. Second, after the Call thread reaches the fourth state, it needs to ask the Main thread to dissociate itself from the user ID. Thus the WebServer thread can use the call-ID-to-user-ID association in the Main thread to locate a Call thread, and the Call thread can also use the association to send messages to a particular web page. As a note, in the second prototype, the control logic is simplified by only having the first Call thread perform accounting. We found that we can keep all the accounting state in the first Call thread and still be able to keep track of who to charge (caller and/or callee)[8].

[8]In the first prototype, we thought that we had to keep accounting states in both Call threads. For outgoing calls, the person to charge is at the first Call thread; and for incoming calls, the person to charge is at the second Call thread.

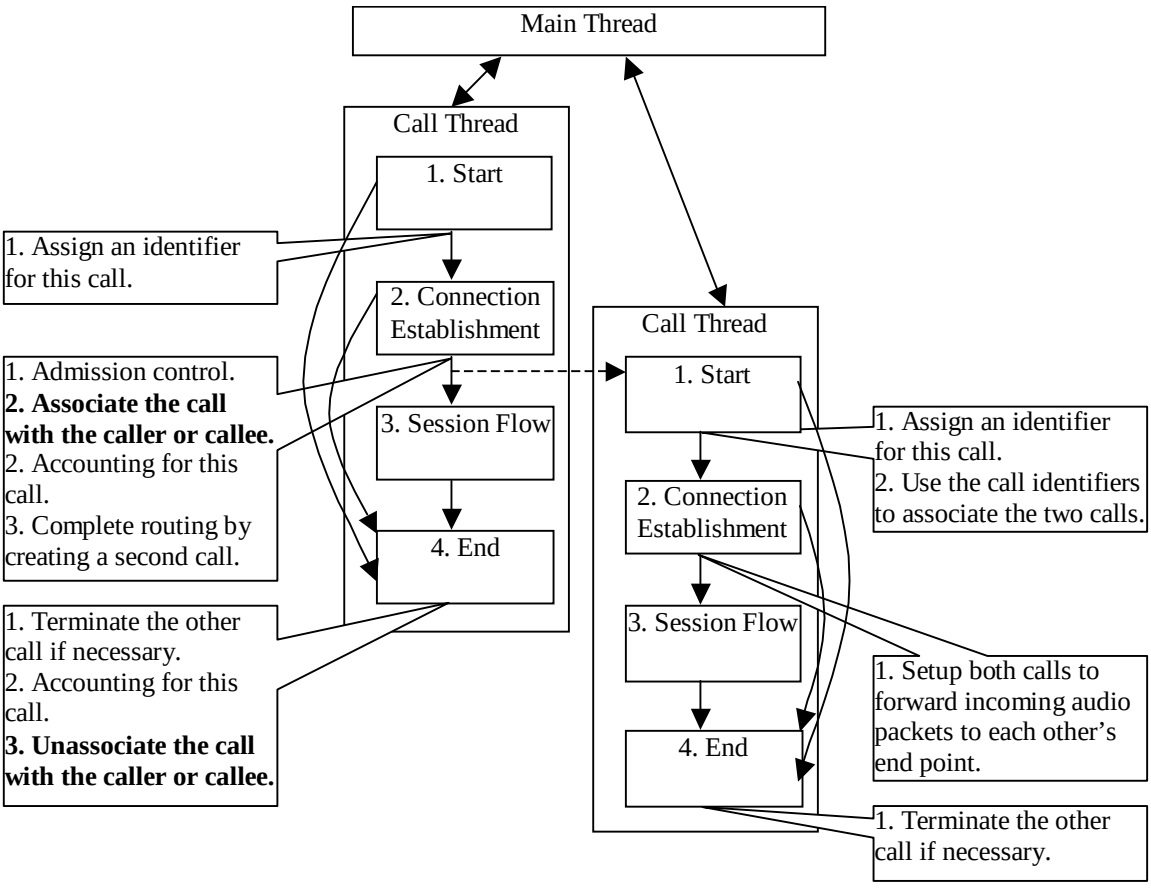

Figure 4.13: Implementation of the second prototype using the four-state FSM.

4.5.4 Deployment

We offered the second prototype to 50 students in our department in the Spring of 2000. 25 graduate students and 25 undergraduate students signed up. We assigned each user a four-digit user ID and allowed them to place their IDs on a web page serving as a phone directory. We setup our service in one undergraduate computer cluster containing 29 Windows NT machines and on five information kiosks in our building. Thus the students could use these public computers, as well as their own computers, to make and receive phone calls. We limited the outgoing calls to local numbers around Berkeley.

We did not obtain as much usage as we had hoped and only half of the users who signed up used the service. Figure 4.14 shows the weekly usage. During the Spring Break (week of 3/27/00), we modified the service to allow outgoing calls to all Bay area. With the expanded call coverage, we were only able to maintain the same usage level as before.

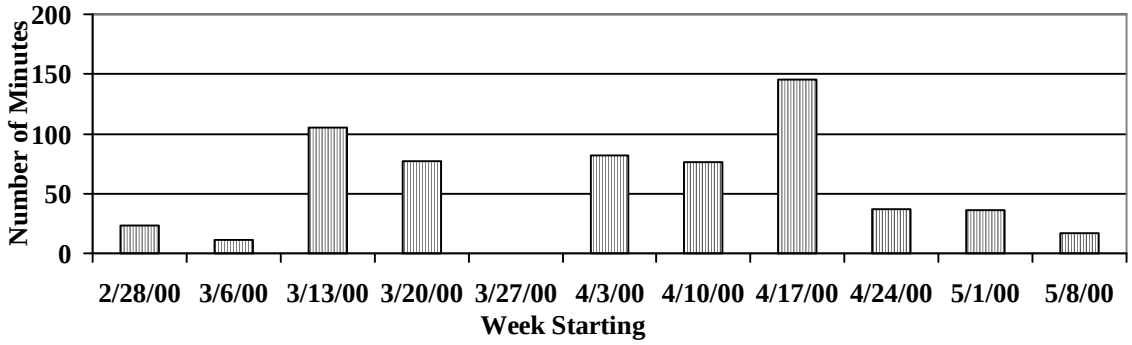

Figure 4.14: Weekly usage of the second deployment[9].

4.5.5 Lessons

We found that it is easy to integrate the H.323 protocol with an out-of-band control channel like a web interface when the H.323 call is abstracted as a simple four-state FSM. During deployment, the service was again easy to deploy and maintain. Users only needed to install NetMeeting to access the service through a web page. The service was available all the time except for the weekly maintenance reboot. From deployment, we still found that it is hard to replace phones with computers. Through usage, we found that we can attract users to make calls from their computers. We even enticed some users to use the service to receive incoming calls on their phones. However it is hard to persuade users to receive incoming calls on their computers. During the service, we sent

[9] We did not have all 50 students sign up until after 3/13/00.

out surveys and held focus groups. In general, users stated that the service and the web interface are easy to use. However, they suggested several improvements like adding dial tone, ringing, etc. Students said that they do not really need to make many calls while at school. At home, they stated that they can easily use free Internet-telephony services like Dialpad and Net2Phone instead of our service. Finally, users indicated that they are reluctant to use a computer-telephony service because the voice quality on the Internet can be poor sometime.

4.6 Third Prototype

4.6.1 Requirements

The goal for our third prototype was again to obtain more usage to enable user experiments. We decided to target students in the dormitories because they have high-speed Internet connections (Ethernet) and would make lots of calls in their rooms. Based on the lessons learned from the first two deployments, we decided to support the following features.

For outgoing calls, we would like to allow users to use their computers or phones to make PSTN calls. From a computer, users would use a web interface as before. From a phone, users would first call the gateway and use an Interactive Voice Response system to enter their user IDs and PINs followed by the phone numbers they want to call. The phone interface would be similar to using a calling card. By allowing users to make calls from a phone, we offered a function that the free Internet-telephony services do not have.

For incoming calls, we would like to forward users' calls to their computers when they are logged on, and to their chosen PSTN numbers when they are not. To make it easier to receive incoming calls, users can receive them from any phone or any computer

running NetMeeting. Thus anyone can use their computers or phones to call our users. As a note, free Internet-telephony services do not support incoming calls.

During calls, we would like to provide users with an option when experiencing poor voice quality by transferring an active call from a computer to a phone. Furthermore, during calls, we would also like to allow users to send and receive DTMF tones, hear in-band voice messages, and receive real-time connection quality and pricing information. Again, at the time of the prototype, these features are not supported by the free Internet-telephony services.

4.6.2 Architecture

The architecture is the same as the one shown in Figure 4.10 except that anyone using NetMeeting can also call the proxy to reach users. Calls from the Internet to the PSTN work as before. However, calls from the PSTN now contain an extra step. When a user calls the proxy from a phone, he will first hear a recorded voice message from the proxy like “welcome, enter your user ID and PIN, or enter the user ID of the person you want to reach”. Then based on the DTMF tones that the user presses, the proxy will play out other recorded messages like “enter the phone number”, “not enough tokens”, “user not reachable”, etc. Finally, after gathering enough information, the proxy will then route the call.

For call transfer, we decided to automatically transfer a user’s active call to the last device he is accessing from. Call transfer works slightly differently depending on whether a user wants to transfer his call to a computer or to a phone. To transfer his active call to a computer, the user first needs to login to the web page at that computer. When the proxy receives the login request from the computer, it will notice that the user

is already in an active call. Thus the proxy will disconnect the user at his old location and ring the user at the computer. When the user answers, the proxy will then connect him to his active call. For the other scenario of a user wanting to transfer his active call to a phone, he will need to first use the phone to call the gateway. After entering his user ID and PIN through the phone, the proxy will notice that the user is already in an active call. Thus the proxy will immediately disconnect the user at his old location and connect his active call to the phone.

4.6.3 Proxy Implementation Details

The control logic for implementing the third prototype is the same as the one shown in Figure 4.13 except at one place. For calls coming from the PSTN, the first Call thread at the second state needs to first play out recorded messages and receive DTMF tones before continuing with admission control, accounting, and routing.

To support call transfers between devices, we found that we needed to clearly define the control functions added at each state of the four-state FSM so that we can easily change a Call thread's state. To illustrate, we will use the more complicated scenario of a user wanting to transfer his active call from his phone to his computer. When the proxy receives a login request from the computer, it will notice that the user is already in an active call. Thus the proxy will first terminate the Call thread handling the user's phone and then roll back the other Call thread to the first state of a first Call thread. Thus the rolled back Call thread can ask the Main thread to create a new Call thread to connect to the user's computer. After the computer answers, the proxy will then connect the two Call threads together to complete the call transfer. With the actions at each state clearly defined, we can easily roll a Call thread forward or backward through the four-

state FSM to connect two existing Call threads for supporting call transfers. As a note, in the third prototype, we found that it is cumbersome to move the accounting states at the first Call thread to a new first Call thread. Thus we placed the accounting states with each user record in the Main thread and access them through a pointer (call-ID-to-user-ID association).

4.6.4 Deployment

We deployed the service to students in the dormitories during the 2000-2001 academic year and were able to sign up 100 users at the end. Figure 4.15 shows the weekly usage for the Fall of 2000. We were able to use the service to conduct various pricing experiments.

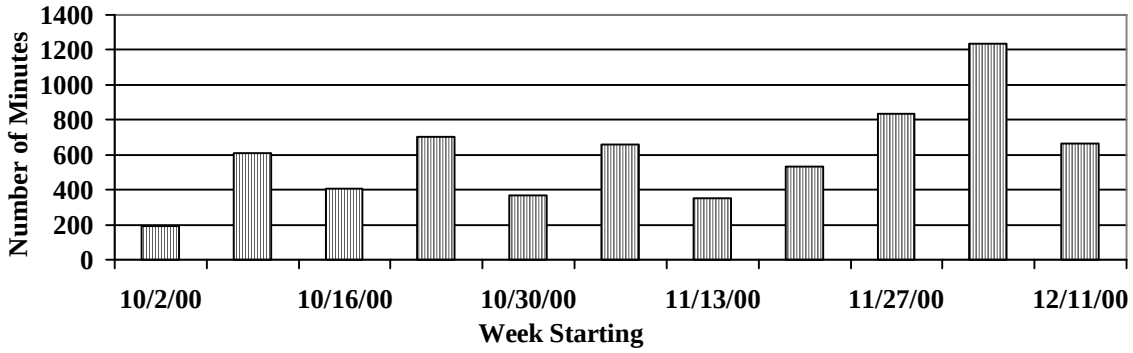

Figure 4.15: Weekly usage of the third deployment.

4.6.5 Lessons

We found that we can use our abstract FSM to easily implement device handoff. To handoff a call, we needed to clearly specify the control logic added at each state so that we can easily roll a call forward or backward through the four-state FSM. The

deployment was again easy with the use of NetMeeting and web interface, and the service was available all the time except for the two-minute weekly maintenance reboot.

4.7 Conclusion

From prototyping three versions of a voice-over-IP gateway service, we found that we can easily implement various features like admission control, accounting, call redirection, and handoff in a proxy. In the first version, we found that we can rapidly prototype various features in the proxy by viewing a H.323 call as a four-state FSM and adding simple control logic at each state. In the second version, to extend user interface for supporting additional inputs and outputs, we found that we can easily use the FSM to integrate the H.323 protocol with an out-of-band control channel like the web interface. In the third prototype, to provide users with an option when the voice quality on the Internet becomes poor, we found that we can support call transfer between a phone and a computer by clearly specifying the control logic added at each state of the four-state FSM and rolling a call forward or backward through the states.

There are several lessons we learned from our deployment efforts. First, it is not easy to entice users to use their computers instead of phones. We found that it is especially difficult to persuade users to use their computers for receiving incoming calls. To make using a computer-telephony service more attractive, we needed to inform users the call status all the time and provide them with an option, like switching to a PSTN phone, when the voice quality on the Internet becomes poor. After successfully attracting a large group of users to our service, in the next Chapter, we will describe how we used the service and the users to conduct various pricing experiments.

Chapter 5 Voice Traffic Pricing Experiments

As the second step of the four-step methodology for voice traffic described in Chapter 3, we used the voice-over-IP gateway service developed in Chapter 4 to conduct pricing experiments with a small number of users. Our goal for this step is to understand how dynamic pricing can modify user behaviors in a desirable way. For example, can it encourage user sessions to become shorter, be deferred, or accept a lower quality of service. To investigate dynamic pricing, we first conducted user experiments to find a scheme that is most effective in changing user behavior. Afterwards, we conducted further experiments to measure the scheme's performance and user acceptance. The critical performance metrics are the effects of the size and the frequency of price changes on user behaviors. To determine user acceptance, we surveyed users about their experience and preferences with dynamic pricing. Among the answers we sought is how much monetary incentive is needed to entice users to choose congestion pricing over flat-rate pricing. With a better understanding of user response and acceptance to price changes, we can formulate a user model for simulation studies to quantify the tradeoff between system performance and user satisfaction.

In Section 5.1, we first describe the setup of the user experiments. We explain the design of the experiments for verifying the setup and measuring the effects of dynamic pricing in Section 5.2. In Section 5.3, we summarize the results of the experiments. We report the user surveys conducted after the experiments in Section 5.4. Finally, in Section 5.5, we conclude with the lessons learned from the user study.

5.1 Experimental Setup

We targeted the voice-over-IP service to dormitory students because they have high-speed Internet connections (Ethernet) and regular PSTN phones in their rooms. Furthermore, these students are familiar with using computers and like to talk on the phone. The students who signed up for the experiments are mostly freshmen and sophomores, and come from a wide variety of intended majors. Admittedly, our users had many options for making and receiving phone calls. On their computers, they could use free Internet-telephony services like Dialpad and Net2Phone to make free long distance calls to anywhere in the U.S. However, our service has better voice quality because it is on the same local area network as their computers. From their room phones, users could make free local calls and could pay additional for long distance calls. Furthermore, half of our users also had a cell phone. These outside phone options made it more challenging for us to build a user community in order to conduct statistically significant pricing experiments. However, with all these outside phone options, we can better understand how to apply dynamic pricing so that it would be acceptable and effective.

To constrain users, we used a token system that limits each user to 1000 free tokens a week and charges him/her a certain token rate per minute. We chose the charging rates so that an average user will run out of tokens by the end of a week. The unused tokens disappear at the end of a week so that we can perform a different pricing experiment each week. However, having tokens disappear makes them less valuable and might cause users to be less responsive to price changes. Users might even attempt to use all of their remaining tokens near the end of a week. We did not charge users real money because it would have complicated the approval process for conducting our user study in

the dormitories. Thus we needed to ensure that using our token system can place a constraint on users.

When users are using the web interface from a computer, they will see the current rate, the next minute rate, the call duration, the total call charge, and the tokens left (see Figure 5.1). When users are using the service from a phone, they will hear the current price at the beginning of a call and whenever it changes. We used these real-time pricing information to encourage users to talk less, talk at another time, or talk using a lower connection quality after a price increase.

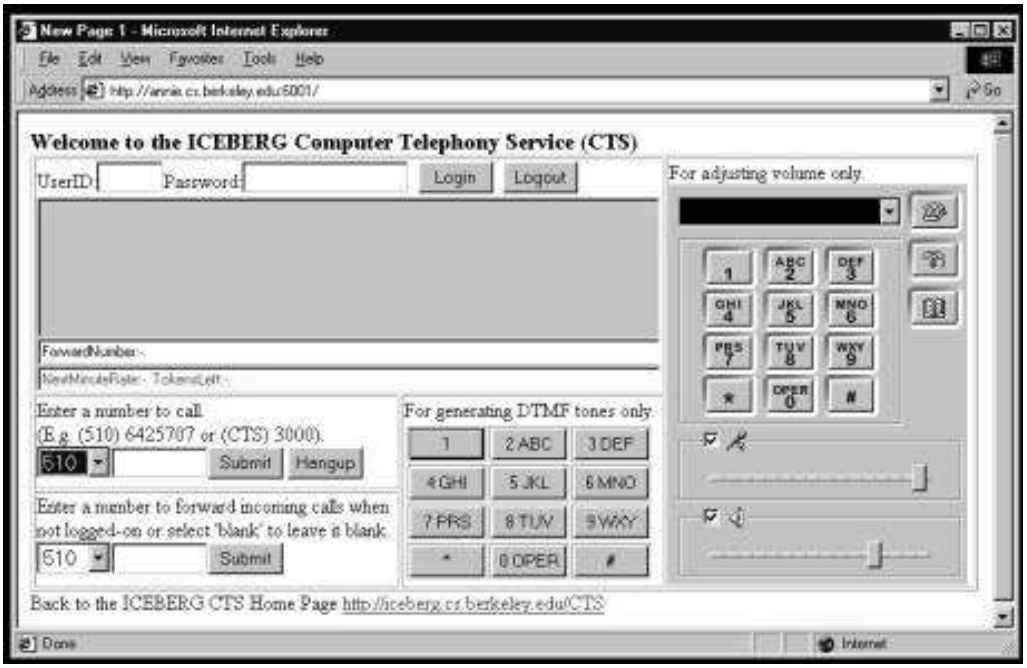

Figure 5.1: Web interface for making calls.

We started the experiments during the Fall of 2000. At the time, we only allowed outgoing calls to the Bay area and had only 40 users signed up. All the users face the same pricing policy each week. The policies are announced several weeks beforehand and users are reminded of the current policy in the beginning of a week through email.

During the week, users can access up-to-date accounting information. At the end of a week, users receive email statements summarizing their usages. Thus users can look at how quickly they used up their tokens to better understand how they should react to price changes in the coming week.

5.2 Experimental Design

Our approach is to first experiment with static pricing policies, where users know ahead of the time the charging schedules, to verify that our experimental setup (target users, token system, and user interface) can affect user behavior. Afterwards, we will use the same setup to investigate dynamic pricing.

During the Fall of 2000, we decided to use the third prototype of the voice-over-IP gateway service described in Chapter 4 to experiment with the following five policies:

- *Flat-rate pricing* – same rate all the time.
- *Time-of-day pricing* – a higher rate during peak hours.
- *Call-duration pricing* – a higher rate as a call lasts longer.
- *Access-device pricing* – a higher rate when using a phone and a lower rate when using a computer.
- *Congestion pricing* – a rate that rises and falls with the total number of active calls.

The first four policies are *static*, meaning users know ahead of the time the charging schedules. We used *flat-rate pricing* as a baseline for comparison with *congestion pricing*. We selected *time-of-day pricing* and *call-duration pricing* because they are static pricing policies that have the benefits of *congestion pricing*. *Time-of-day pricing* encourages users to talk at another time while *call-duration pricing* encourages users to

shorten their call sessions. We experimented with *access-device pricing* to understand if pricing can encourage users to call from their computers instead of phones. We attracted users to use our service instead of other computer-telephony services because we allowed them to make calls from their phones. Thus, we were curious to determine if pricing can encourage users to use their computers or to transfer their active calls from their phones to their computers. More importantly, by enticing users to use their computers instead of phones, *access-device pricing* helps reduce the demand on our service's bottleneck, the phone lines to the PSTN.

Our general approach for conducting user experiments was to take one small step at a time. Table 5.1 lists the experiments conducted during the Fall of 2000. We started with *flat-rate pricing* of 10 tokens/min to observe users' basic usage like calling time and call duration. With 1000 tokens and 10 tokens/min, users can talk 100 minutes a week. Thus we also wanted to observe if our users would consume sufficient minutes so that the tokens represent an adequate constraint. Afterwards, we decided to experiment with *access-device pricing* because it is the simplest policy for users to understand, a higher price for making calls from a phone and a lower price from a computer. Afterwards, we extended the experiment with *time-of-day pricing*, where users are charged a higher price during the peak hours from 7pm-11pm, because users are already familiar with such policy. We found that we can easily cause users to shift their calling pattern depending on the peak-hour price. Thus *time-of-day pricing* gave us confidence that using a free but limited token scheme can influence user behavior. We then decided to experiment with *call-duration pricing* to determine if we can entice users to limit their call durations. We were also curious to understand whether users would value the first few minutes of a call

differently from the later minutes. After obtaining positive results from these two static policies, *time-of-day* and *call-duration*, we moved on to experiment with *congestion pricing* where the price depends on the actual number of calls using the service. During the last week, we decided to experiment with *flat-rate pricing* of 5 tokens/min to observe if users would make more calls. If users do, then it would confirm that charging 10 tokens/min do place a reasonable constraint on users.

Table 5.1: Experiments during the Fall of 2000.

Week Starting	**Pricing Policy**	**Total Users**	**Active Users**	**Total Calls**	**Total Minutes**
10/2/00	Flat-rate: 10 tokens/min	22	12	66	190
10/9/00	Flat-rate: 10 tokens/min	32	12	102	612
10/16/00	Access-device: Computer: 10 tokens/min Phone: **20** tokens/min	32	15	91	406
10/23/00	Access-device: Computer: 10 tokens/min Phone: **10** tokens/min	35	12	92	702
10/30/00	Access-device: Computer: 10 tokens/min Phone: **30** tokens/min	37	12	61	367
11/6/00	Time-of-day: 11pm-7pm: 10 tokens/min 7pm-11pm: 30 tokens/min	28	13	117	657
11/13/00	Call-duration: 1^{st}-3^{rd} min: 10 tokens/min 4^{th} min on: 30 tokens/min	41	12	59	349
11/20/00 Thanksgiving	Congestion: 10X tokens/min X: number of active calls	41	10	63	530
11/27/00	Congestion: 10X tokens/min X: number of active calls	41	14	85	837
12/4/00	Flat-rate: **5** tokens/min	41	12	110	1238

We did not obtain enough samples for the *congestion pricing* experiments conducted during the Fall because with a small user group, there were only a few instances of more than one user using the service. Thus during the Spring of 2001, we decided to extend the call coverage to all California and were eventually able to enlist

100 users. During the first part of the Spring (see Table 5.2), we experimented with static policies (*flat-rate*, *time-of-day*, and *call-duration*) to confirm their results before experimenting with *congestion pricing*. For *congestion pricing*, we were surprised that changing prices had no influence on user behavior even though *time-of-day pricing* was able to cause users to defer their usages to another time and *call-duration pricing* was able to entice users to shorten their calls. Through surveys, we found that when we allowed prices to change from one minute to the next according to load, users did not terminate their calls early because their past experience suggested that prices would drop in the next few minutes. Thus users did not respond to price changes.

Table 5.2: Experiments during the first part of the Spring of 2001.

Week Starting	**Pricing Policy**	**Total Users**	**Active Users**	**Total Calls**	**Total Minutes**
2/5/01	Flat-rate: 10 tokens/min	45	15	74	553
2/12/01	Time-of-day: 11pm-7pm: 10 tokens/min 7pm-11pm: 30 tokens/min	60	21	133	927
2/19/01	Call-duration: 1st-3rd min: 5 tokens/min 4th-10thmin: 10 tokens/min 11th-20th min: 20 tokens/min 21st min on: 30 tokens/min	70	31	128	961
2/26/01	Flat-rate: 10 tokens/min	76	41	196	1925
3/5/01	Congestion: **10X** tokens/min X: number of active calls	81	42	251	2078
3/12/01	Congestion: **5X** tokens/min X: number of active calls	87	42	282	2445
3/19/01	Flat-rate: **5** tokens/min	88	40	296	2441
3/26/01 Spring Break	Call-duration: 1st-5th min: 5 tokens/min 6th-15th min: 10 tokens/min 16th-23rd min: 20 tokens/min 26th min on: 30 tokens/min	91	28	134	660

For the second part of the Spring semester (see Table 5.3), we experimented with making each price increase under *congestion pricing* more costly. For example, we

experimented with dynamic pricing where prices would only increase during a call or where each price increase would last at least a few minutes before changing again. In these experiments, we decided to use heuristics to set prices instead of setting them according to load to ensure that we have enough samples each week. During this period, we had more users and decided to divide them into two groups so that we can experiment with more variations of *congestion pricing*. After finding a *congestion pricing* scheme that can change user behavior, we then conducted further experiments to measure the effects of different price increases in causing users to terminate their calls early. Near the end of the semester, we decided to experiment with *quality-based pricing*, where users would pay a higher rate for a higher quality, to encourage users to use a lower connection quality. Thus if there are two gateways, one closer that has higher quality and one further away that has lower quality, pricing can be used to encourage users to use the gateway further away when the nearby gateway is congested.

Table 5.3: Experiments during the second part of the Spring of 2001.

Week	Group 1 Policy	Group 2 Policy	Group 1 Statistics (Total User/ Active User/ Total Calls/ Total Minutes)	Group 2 Statistics (Total User/ Active User/ Total Calls/ Total Minutes)
4/2/01	Time-of-day: 11pm-7pm: 10 tokens/min 7pm-11pm: **20** tokens/min	Time-of-day: 11pm-7pm: 10 tokens/min 7pm-11pm: **15** tokens/min	(46/ 17/ 94/ 772)	(47/ 24/ 198/ 1253)
4/9/01	Call-duration: 1st-5th min: 5 tokens/min 6th-15th min: 10 tokens/min 16th-25th min: 15 tokens/min 26th min on: 20 tokens/min	Call-duration: 1st-5th min: 20 tokens/min 6th-15th min: 15 tokens/min 16th-25th min: 10 tokens/min 26th min on: 5 tokens/min	(46/ 17/ 109/ 913)	(47/ 25/ 178/ 1261)
4/16/01	Congestion: Initial rate: 10 tokens/min. **Rate increases** by 10 tokens and **then decreases** by 10 tokens. Rate can change from one minute to the next, but on average once every 10 minutes.	Congestion: Initial rate: 10 tokens/min. **Rate can only increase**, by 5 tokens each time, with a maximum rate of 25 tokens/min. Rate can increase from one minute to the next, but on average once every 10 minutes.	(47/ 18/ 129/ 907)	(47/ 27/ 223/ 1280)
4/23/01	Congestion: Initial rate: 10 tokens/min. Rate increases by 10 tokens and then decreases by 10 tokens. Rate can change at most once every **3** minutes, but on average once every 10 minutes.	Congestion: Initial rate: 10 tokens/min. Rate increases by 10 tokens and then decreases by 10 tokens. Rate can change at most once every **5** minutes, but on average once every 10 minutes.	(47/ 18/ 101/ 627)	(48/ 25/ 202/ 1084)
4/30/01	Congestion: Initial rate: 10 tokens/min. Rate increases by **5** tokens and then decreases by **5** tokens. Rate can change at most once every 3 minutes, but on average once every 10 minutes.	Congestion: Initial rate: 10 tokens/min. Rate increases by **15** tokens and then decreases by **15** tokens. Rate can change at most once every 3 minutes, but on average once every 10 minutes.	(48/ 18/ 125/ 765)	(49/ 26/ 196/ 1498)
5/7/01	Quality-based: Low quality: 10 tokens/min High quality: **20** tokens/min	Quality-based: Low quality: 10 tokens/min High quality: **20** tokens/min	(49/ 21/ 106/ 833)	(49/ 29/ 164/ 1048)
5/14/01	Quality-based: Low quality: 10 tokens/min High quality: **15** tokens/min	Quality-based: Low quality: 10 tokens/min High quality: **25** tokens/min	(49/ 18/ 81/ 562)	(49/ 26/ 169/ 892)

5.3 Results

In this section, we will summary the results for each of the pricing policies, starting with the static policies.

5.3.1 Flat-Rate Pricing

We used *flat-rate pricing* to calibrate our price levels. As we decreased the price from 10 tokens/min to 5 tokens/min, users in general talked a third more in a week. This suggested that we did place a constraint on users by limiting each to 1000 tokens a week and charging about 10 tokens/min.

Flat-rate pricing also provided us with a better understanding of user usage. Figure 5.2 shows the calling pattern of all the calls (1140 calls totaling 9299 minutes) under *flat-rate pricing*. From the figure, users tend to call between 7pm-11pm. This period accounts for 54% of the usages. There is also a slight variation of the calling pattern between weekdays and weekends. For call durations, Figure 5.3 graphs the percentage of the calls that are longer than a certain duration. As shown in the figure, 30% of the calls are longer than 5 minutes, 20% longer than 11 minutes, and 10% longer than 23 minutes. Thus we can also calculate the probability that a user will hang up after a certain duration (see Figure 5.4). We found that after the 3rd minute of a call, the probability that the call will end in the next minute is a constant, 5.8%, with a small standard error of 0.4%. These statistics about *flat-rate pricing* are useful for comparison with other policies later on.

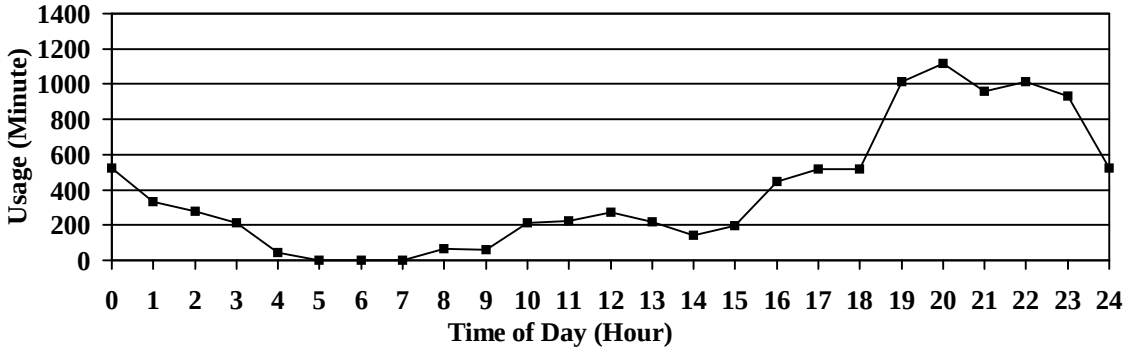

Figure 5.2: Calling pattern under flat-rate pricing.

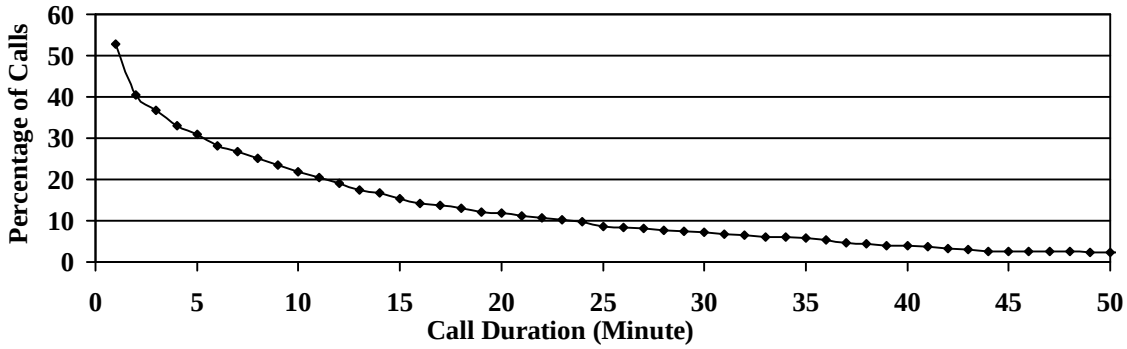

Figure 5.3: Percentage of calls longer than a certain duration.

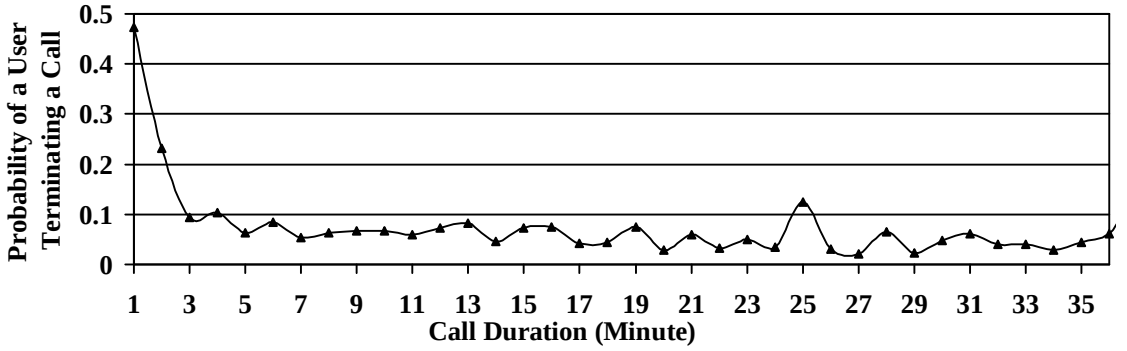

Figure 5.4: Probability of a call terminating after a certain duration.

5.3.2 Time-of-Day Pricing

For *time-of-day pricing*, we first experimented with charging 30 tokens during the peak hours, 7pm-11pm, and 10 tokens during the off-peak hours, 11pm-7pm. We selected 30 tokens because we wanted a big price difference to measure how much users would shift their usages. The experiment was conducted twice, once in the Fall and once in the Spring. When comparing with *flat-rate pricing*, we found that *time-of-day pricing* can encourage users to shift about 30% of their usages from the peak to the off-peak hours (see Figure 5.5). In the figure, all the calls from all the users in the two *time-of-day* experiments are combined and scaled to overlay with the *flat-rate* calling pattern. With the 20 token price difference, the peak usage shifted to just before and after the peak high-priced hours. There was also a small peak around 11am due to calls made during the weekends. Figure 5.6 and Figure 5.7 show the effects of *time-of-day pricing* when the price difference is smaller. These results from the *time-of-day* experiments demonstrated that our token scheme can entice users to call at another time.

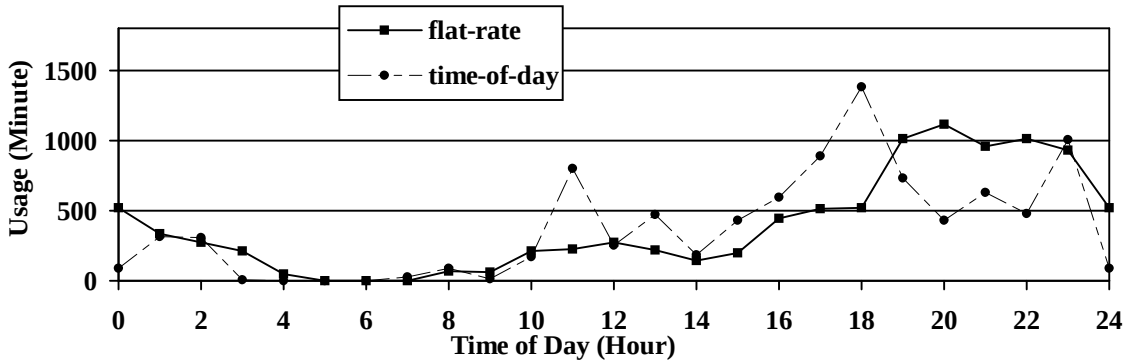

Figure 5.5: Time-of-day pricing with 30 tokens/min from 7-11pm and 10 tokens/min otherwise.

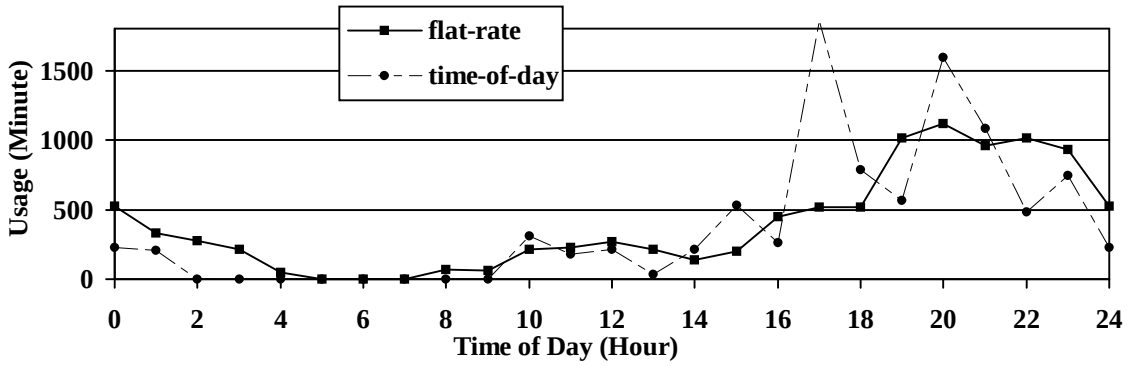

Figure 5.6: Time-of-day pricing with 25 tokens/min from 7-11pm and 10 tokens/min otherwise.

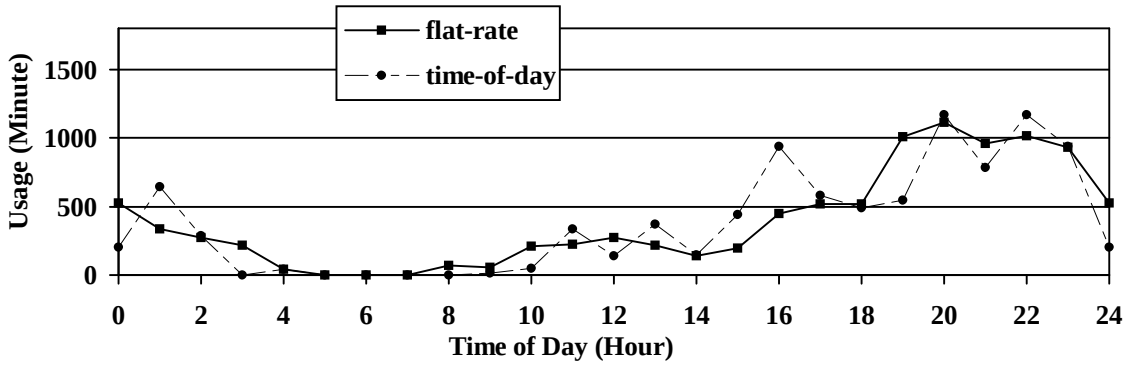

Figure 5.7: Time-of-day pricing with 20 tokens/min from 7-11pm and 10 tokens/min otherwise.

5.3.3 Call-Duration Pricing

We experimented with different parameters for *call-duration pricing* and found that it can encourage users to talk less. For all the experiments, we informed users in the beginning of the week when a price increase would occur during a call. In the first experiment, we increased the price after the 3rd minute and could not cause more calls to terminate. In the second experiment, we increased the price after the 3rd, the 10th, and the 20th minute. We found that the price increase after the 3rd minute again had no effect, but the price increases after the 10th and the 20th minute caused about three times as many calls, 18% instead of 5.6%, to terminate in the next minute (see Figure 5.8). In the third

experiment, the price is increased after the 5th, the 15th, and the 25th minute. Each price increase is again able to cause about three times as many calls to terminate when compare to *flat-rate pricing* (see Figure 5.9). These results indicate that the first few minutes of a call are important to users, but increasing prices after the first few minutes can easily entice users to talk less.

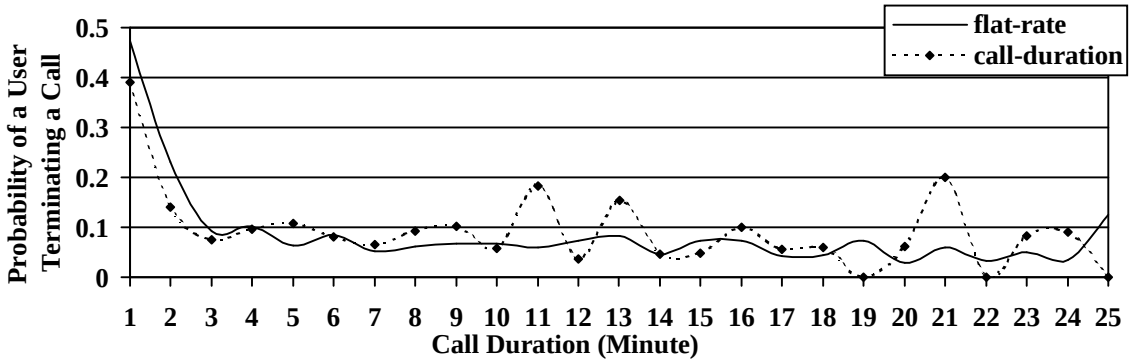

Figure 5.8: Call-duration pricing (week of 2/19/01) with a price increase after the 3rd, the 10th, and the 20th minute of a call.

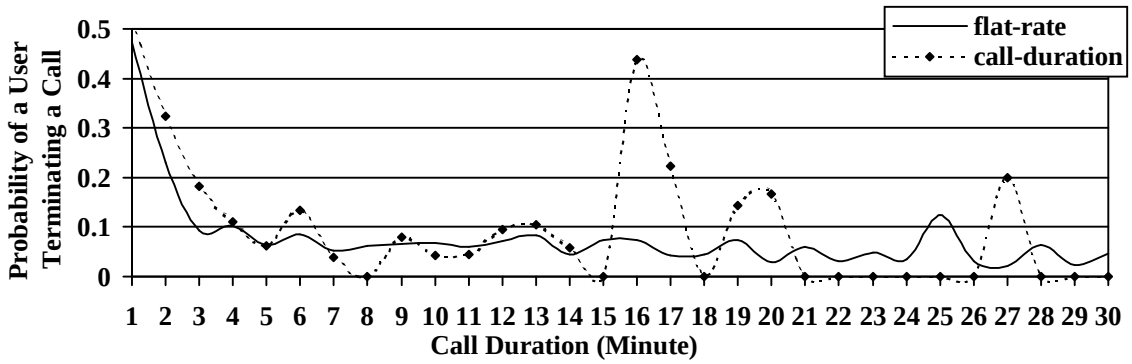

Figure 5.9: Call-duration pricing (week of 3/26/01) with a price increase after the 5th, the 15th, and the 25th minute of a call.

5.3.4 Access-Device Pricing

For *access-device pricing*, we found that it is not easy to entice users to use their computers instead of phones (see Table 5.4). With a 20 token price difference, 76% of the users still preferred to use their phones. Using surveys, users indicated that if they want to use their computers, then they can easily use free computer-telephony services like Dialpad and Net2Phone. Thus with free outside phone options, it was difficult to entice users to shift their usages from phones to computers.

Table 5.4: Results from access-device pricing.

Rate Using a Phone (Tokens/Min)	Rate Using a Computer (Tokens/Min)	Minutes Using a Phone	Minutes Using a Computer	% Minutes Using Phone
10	10	668	34	95%
20	10	347	59	85%
30	10	278	89	76%

5.3.5 Quality-Based Pricing

For *quality-based pricing*, we also found that it is difficult to convince users to use a lower quality (see Table 5.5). For the lower quality, we added one second of delay to model playout buffer for absorbing delay and jitter when using a gateway further away. We selected one second because we wanted to be sure that users can easily distinguish between the two qualities. We verified with a few users that they can still use the lower quality to carry on their conversations. However, with a 20 token difference, 77% of the users still preferred to use the higher quality. Thus, users were reluctant to use a lower quality that had an extra second of delay.

Table 5.5: Results from quality-based pricing.

Rate of High Quality (Tokens/Min)	Rate of Low Quality (Tokens/Min)	Minutes Using High Quality	Minutes Using Low Quality	% Minutes Using High Quality
15	10	549	13	98%
20	10	1571	310	84%
25	10	685	207	77%

5.3.6 Congestion Pricing

For *congestion pricing*, we set the current rate as a function of the number of simultaneous calls using the service. Thus the rate increases when more people are calling and decreases when less are. We conducted two *congestion pricing* experiments during the Spring semester. In the first experiment, we set the price equal to ten times the number of simultaneous calls. In the second experiment, we set it to five times. During the experiments, we had up to five people calling at the same time. During the first experiment when prices are set according to load, a price increase was announced 57 times and a price decrease was announced 56 times. During the second experiment, a price increase was announced 80 times and a decrease was announced 79 times. There are usually several minutes between price changes. However, in both experiments, we found that after a price increase or decrease, the percentage of calls that terminate in the next few minutes remains unchanged at around 6% (see Figure 5.10 and Figure 5.11). We expected to observe a higher percentage of calls terminating after a price increase and a lower percentage after a decrease. From surveying users, they mentioned that they did not react to price changes because they do not know how long the price increases or decreases would last. Whenever they noticed a price increase, they just hope that the increase is only temporary. Thus to make *congestion pricing* effective, price changes need to be more permanent to better entice users to change their behaviors.

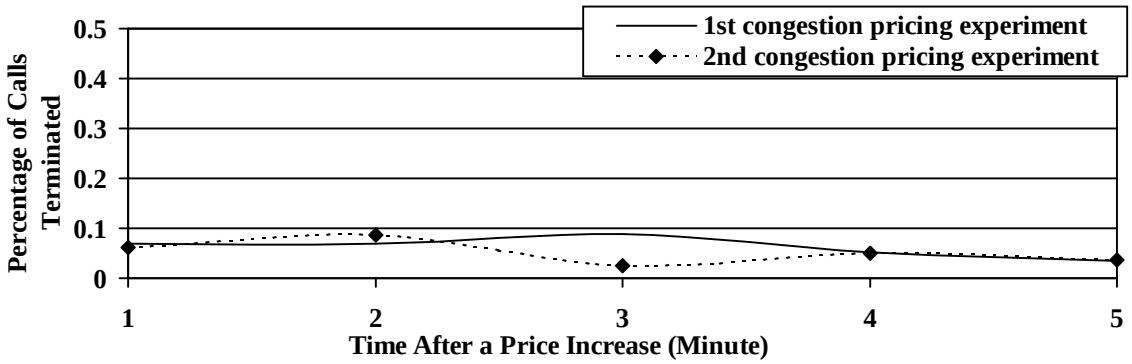

Figure 5.10: Percentage of calls terminating after a price increase.

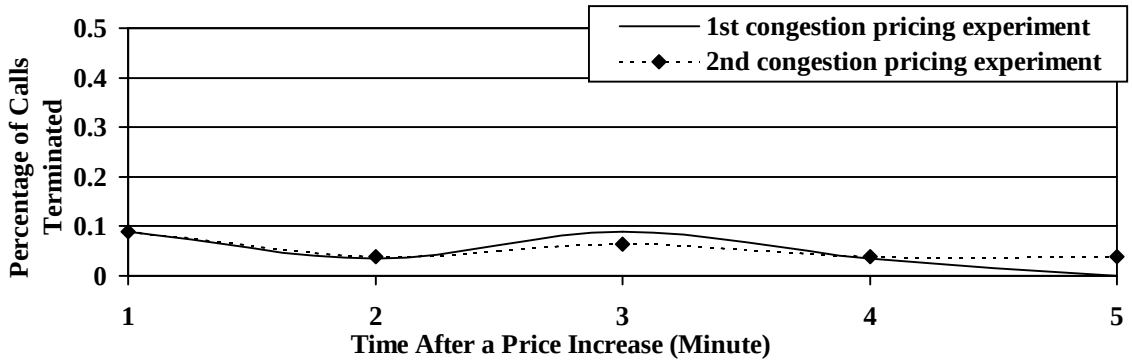

Figure 5.11: Percentage of calls terminating after a price decrease.

We then experimented with *congestion pricing* by informing users that prices would change at most once every three minutes. We artificially increased the price by a certain increment from an initial rate and then decreased it back. The price changes on the average once every 10 minutes. Users did not know that the price is adjusted artificially. We found that depending on the price increment, we can easily get different percentage of the active calls to terminate right away (see Figure 5.12). Thus based on user experiments and surveys, it is important that when prices change, they do not change

from one minute to the next so that users can better predict the cost of not terminating their calls earlier.

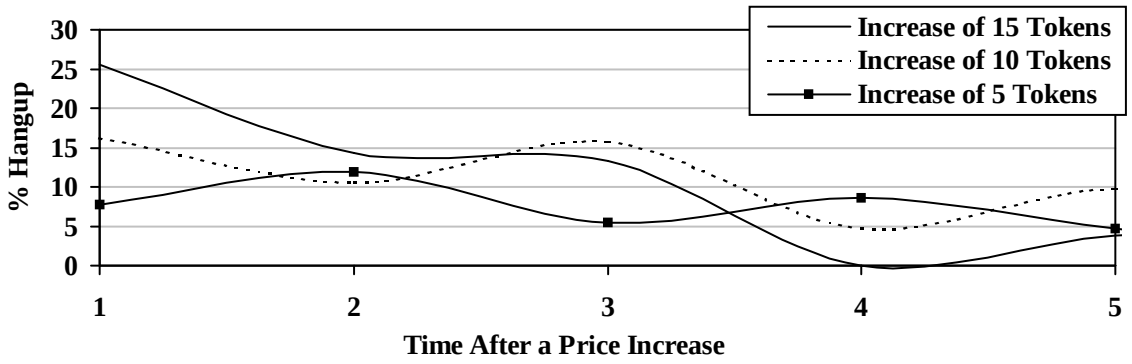

Figure 5.12: Prices can change at most once every three minutes[10]

There are three other interesting results we found from our experiments when prices change at most once every three minutes. For the price decreases associated with Figure 5.12, we found that we still have close to 5.8% of the calls terminating the next minute instead of a percentage like 1% or 2% (see Figure 5.13). Thus users did not try to talk longer after a price decrease. Based on a follow-up survey, users indicated that they feel there is no need to talk longer after a price decrease. Thus price decreases can occur without generating extra demand. Second, we found that the higher the price increase is, the more likely users would hang up and call back within a minute to obtain the lower initial rate (see Figure 5.14). Thus this behavior further confirms that users do react to price increases. However, this behavior also reduces the effectiveness of *congestion pricing*. Thus we might want to either eliminate the initial lower rate or set the period of the initial lower rate small. Third, we found that *congestion pricing* causes the peak

[10] Data fluctuates because it contains only 47 samples of price increase of 15 tokens, 68 samples of price increase of 10 tokens, and 91 samples of price increase of 5 tokens. These price increases occurred in 6.2% of the minutes used by users and their associated price decreases occurred in another 2.6%.

period to shorten from that of *flat-rate pricing* (see Figure 5.15)[11]. Thus *congestion pricing* can also reduce congestion by enticing users to defer some of their peak usages to another time so that they would encounter less price changes. Therefore, the general calling pattern of users can be shifted by the use of dynamic pricing.

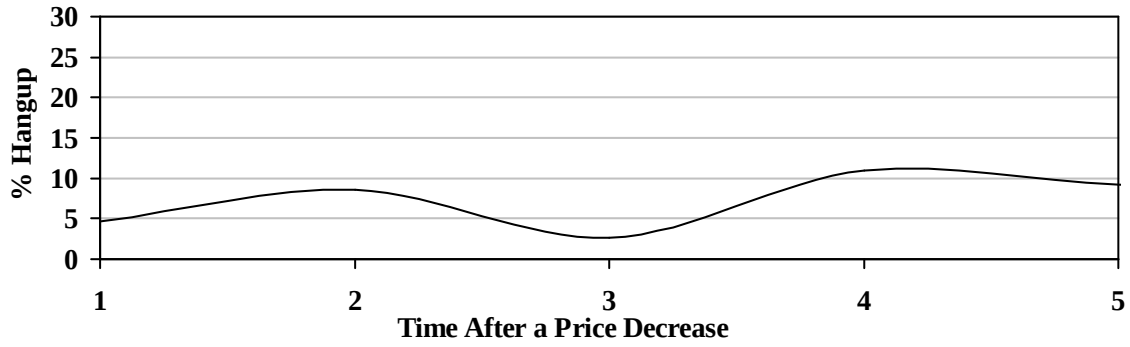

Figure 5.13: Percentage hang up after a price decrease[12].

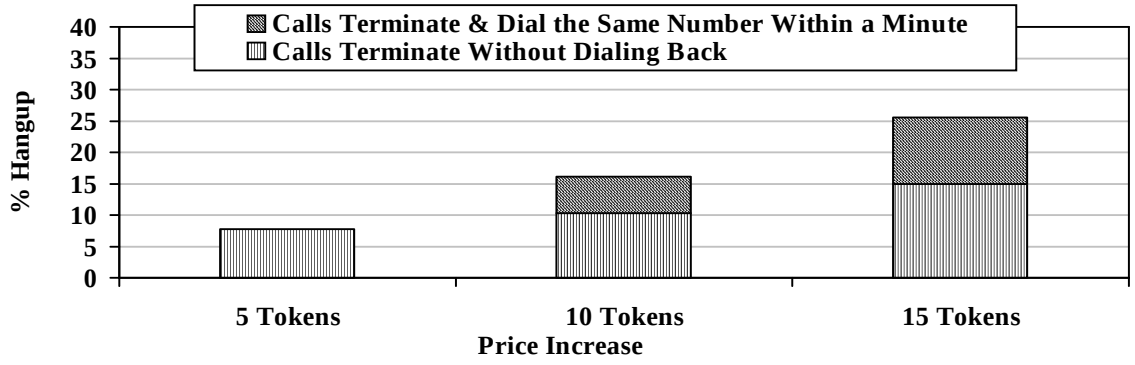

Figure 5.14: Breakdown of calls that terminate within a minute of a price increase.

[11] In the figure, the peak under congestion pricing is higher than the peak under flat-rate pricing assuming both pricing policies handle the same number of minutes. However, based on our later simulation results, we estimate that congestion pricing can reduce the peak load by 20%, thus making both peaks the same.
[12] Data contains 86 price decrease.

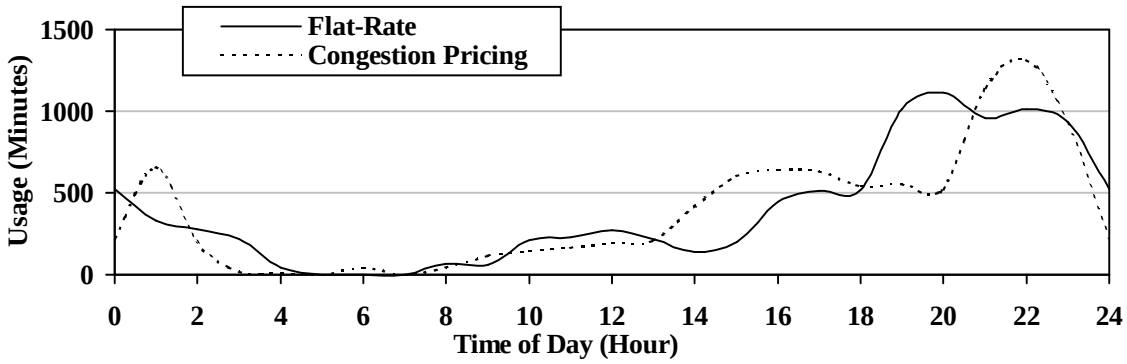

Figure 5.15: Flat-rate and congestion pricing calling pattern[13].

5.4 Surveys

After the experiments, we used a follow-up survey to confirm the results of our observations. We received 23 responses. From the survey, we wanted to find out more about user acceptance to dynamic prices, financial incentives for choosing *congestion pricing*, and stated behaviors to price changes. Appendix A contains the survey questions and additional written responses.

For user acceptance, we first asked users whether they prefer *congestion pricing*. Most of the users did not like it because of the disrupt effects of the price-change announcements. However, when we pointed out that *congestion pricing* can reduce call blocking rate or make prices cheaper, more users preferred it. We also asked users how they would like *congestion pricing* if we use beeps, instead of recorded messages, to indicate the current prices. For example, 1 beep for 10 tokens, 2 beeps for 20 tokens, etc. We found that more users would accept *congestion pricing* if the user interface is not as disruptive as inserting recorded messages in the middle of phone calls. Thus, from the

[13] Data contains 9,299 minutes of calls under flat-rate pricing and 3,347 minutes of calls under congestion pricing. The data from the congestion pricing is scaled to the data of the flat-rate pricing for comparison.

survey, we believe that *congestion pricing* can be designed to be acceptable to users. See Table 5.6 for the questions and their average scores.

Table 5.6: Survey regarding user acceptance.

From a score of 1-5 (5:like it a lot, 4:like it, 3:OK, 2:dislike it, 1:dislike it a lot),	**Average Score (Standard Error)**
How do you like congestion pricing?	2.61 (.19)
What if it can reduce the chance that your calls might be blocked?	3.04 (.20)
What if it can make it cheaper for you to use the service?	3.91 (.16)
What if there is a less disruptive way of announcing price changes like using beeps?	4.70 (.12)

For the financial incentives for choosing *congestion pricing*, we wanted to know how much discount in tokens we need to offer to users for them to choose *congestion pricing* over *flat-rate pricing*. In the survey, we asked the users to choose between a *congestion pricing* policy of a certain average rate or a *flat-rate pricing* of a certain rate, see Table 5.7 for the choices. As the rate under *flat-rate pricing* increases, we can easily entice more users to choose *congestion pricing* over *flat-rate pricing*. This indicates that the users would be willing to use *congestion pricing* if given a small discount as an incentive.

Table 5.7: Survey about financial incentives.

Do you prefer congestion pricing with an average rate of 12.5 tokens/min (80% 10 tokens, 15% 20 tokens, and 5% 30 tokens) or	**0 for congestion pricing and 1 for flat-rate pricing. Average Score (Standard Error)**
flat-rate of 12.5 tokens/min?	0.89 (.07)
flat-rate of 15 tokens/min?	0.37 (.11)
flat-rate of 20 tokens/min?	0.11 (.07)
flat-rate of 25 tokens/min?	0.05 (0.5)

Finally, for the stated behaviors to price changes, we questioned our users how they would react to a price change. Most users answered that they would talk shorter after

a price increase, but not longer after a price decrease. However, our users stated that they would behave about the same whether the price changes from one minute to the next or changes at most once every three minutes. Thus from using surveys, it is difficult to distinguish the effects of the frequency of price changes. See Table 5.8 for the questions and their average scores.

Table 5.8: Survey on stated user behaviors.

When the price can change	**1 for yes and 0 for no. Average Score (Standard Error)**
from one minute to the next, would a price increase affect your behavior?	0.36 (.13)
from one minute to the next, would a price decrease affect your behavior?	0.21 (.11)
at most once every 3 minutes, would a price increase affect your behavior?	0.32 (.12)
at most once every 3 minutes, would a price decrease affect your behavior?	0.14 (.10)

5.5 Conclusion

We used a voice-over-IP gateway service to study the effects of pricing on user behavior. We deployed the service for two semesters and signed up about 100 users. Each user is limited to a certain number of free tokens a week and charged a certain token rate per minute when using the service. With this setup, we found that we can easily use static pricing policies, like *time-of-day pricing* and *call-duration pricing*, to entice users to talk shorter or at another time. However, we had more difficulty using this setup to encourage users to talk using a lower quality. For *congestion pricing*, we found that it can entice users to talk both shorter and at another time if prices do not change rapidly, e.g., at most once every three minutes. Using surveys, we found that when prices change slowly and infrequently, dynamic pricing can be designed to be acceptable to users. With this work, we demonstrated a scheme that varies prices in the middle of phone calls to affect user

behavior. In the next chapter, we will describe how we use the results to formulate a user model to drive simulation studies and conduct further user experiments.

Chapter 6 Voice Traffic Simulation Study

Voice operators typically have thousands of subscribers sharing an access point. When managing expensive shared resources, operators would like to minimize capacity, maximize utilization, reduce congestion, and increase user satisfaction[14]. Thus operators would like to understand the potential benefits and drawbacks of applying congestion pricing to a large user population. The benefits of congestion pricing are that it can reduce capacity, increase utilization, and reduce congestion. However, congestion pricing improves these dimensions by decreasing user satisfaction through price changes. Thus operators would like to know the expected improvement in provisioning or call blocking rate, and the expected frequency of price changes. There is always a tension between system performance and user satisfaction when applying congestion pricing. Flat-rate pricing is one extreme where users are very satisfied because prices do not vary, but resource management is expensive or difficult. Operators would either need to overprovision resources or subject users to poor and unpredictable quality during congestion. The other extreme is when prices vary frequently and user annoyance is high, but operators can efficiently allocate available resources to the highest paying subscribers. Thus operators can use congestion pricing to make a tradeoff between system performance and user satisfaction.

To investigate the tradeoff on a larger scale, we continued with the third step and the fourth step of the four-step methodology described in Chapter 3. In the third step, we

[14] Operators would also like to maximize profit, but this has more to do with their market power than dynamic pricing. Thus we assume that operators rely on access fees to maximize profit and use dynamic pricing as a feedback mechanism to efficiently allocate scarce resources.

used the results from the user experiments conducted in Chapter 5 to develop a user behavioral model for large-scale simulation studies. Using simulations, we proposed a set of rules for configuring the parameters that operators can vary when managing congestion pricing to achieve large-scale performance. These rules include when and how quickly operators should adjust prices. By using the suggested values for the parameters, we estimated the tradeoff between call blocking rate (or required provisioning) and price announcement rate. Our estimates strongly depend on our derived user model, so in the fourth step of the methodology, we further re-measured the model by combining user experiments with simulations. We did so by subjecting real users to price changes set by an operator, who in turn responded to the load and the reactions of many simulated users. Thus the time and the increment of the price changes mimic that of a large-scale service. As a consequence, we performed user measurements when users are reacting to price changes of a large-scale emulated service. With these two steps, we present a user model for modeling user reaction to price changes, a set of rules for operators to manage congestion pricing, and an estimate of the benefits and drawbacks of congestion pricing. We found that operators can effectively use congestion pricing for voice calls because they can significantly reduce call blocking rate or save resource provisioning while only submitting users to occasional price changes.

In Section 6.1, we describe the models used for simulations. Then in Section 6.2, we describe the simulation setup followed by the results in Section 6.3. In Section 6.4, we describe how we re-measure the models and calculate the confidence of our user measurements. Finally, in Section 6.5, we present our conclusions and findings on using congestion pricing for a large-scale voice service.

6.1 Modeling

6.1.1 Operator Model

We would like to understand how an operator should set the following four parameters for managing the limited resource, the number of phone lines to the PSTN, at a voice-over-IP gateway service.

- *Capacity* – how many phone lines at the Internet-to-PSTN gateway to purchase.
- *Threshold* – at what load level should congestion pricing be applied.
- *Interval* – how quickly should prices be allowed to change.
- *Init* – how long to wait on new calls before applying congestion pricing to them.

We modeled the operator by assuming that he/she has a limited *capacity* of phone lines to the PSTN. When all the phone lines are used, the operator would need to block new calls. However, the operator can adjust prices to affect load. To limit fluctuation of prices, we assumed that there is a minimum and a maximum rate. The operator would charge the minimum rate most of the time. However, when the load is above a *threshold* value, the operator would increase prices to everyone, and when the load is below it, the operator would decrease prices. We also assumed that the operator changes prices by the same increment each time. Finally, the operator sets *interval* and *init* to limit the frequency of price changes.

Each of these four parameters provides operators with a tradeoff between system performance and user satisfaction. If *capacity* is too high, then most of the phone lines would be idle and users would rarely encounter congestion. On the other hand, if *capacity* is too low, then utilization would improve, but contention would increase. For the *threshold* value, if it is set too low, then prices would start increasing even when there is

plenty of available resources. However, if the *threshold* is set too high, then calls might be blocked due to congestion. For the *interval*, if it is set too small, then users would be annoyed and might not even respond to price changes. On the other hand, if the *interval* is too large, then operators might not be able to change prices in response to sudden changes in load. For the *init* value, if it is set too small, then users would be annoyed and unresponsive because the first few minutes of a call are important. On the other hand, if *init* is set too large, then users might hang up and call back to obtain the lower initial rate; thus, rendering congestion pricing ineffective for reducing the load on the system. Thus, these four parameters need to be set appropriately to fully achieve the benefits of congestion pricing.

6.1.2 User Model

There are two parts to modeling a user's behavior. The first is the pattern of making calls, and the second is the reaction to price changes. For both parts, we modeled the user when he/she is limited to 1000 tokens a week so that we can exploit the results from our user experiments.

For a user's calling pattern, we used traces from flat-rate pricing instead of congestion pricing because we wanted a calling pattern that is not yet affected by dynamic pricing. Thus, the calling pattern does not take into account that a user might shift his/her calling pattern due to variable prices. For flat-rate pricing, we had traces of 1140 calls totaling 9299 minutes. These calls were made by 175 users in a week. Thus to generate a user's workload, we randomly selected from the traces the number of calls he/she would make in a week (see Figure 6.1), when these calls would be made (see Figure 6.2), and how long these calls would last (see Figure 6.3). For example, Figure 6.4

shows an instance of the aggregated load made by 10,000 simulated users in a week. The load in the figure exhibits periods of high utilization followed by low utilization.

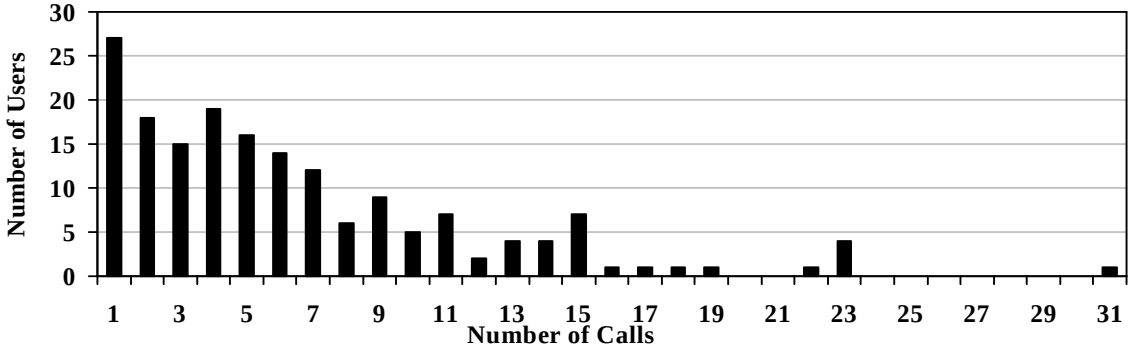

Figure 6.1: Traces of the number of calls a user makes a week.

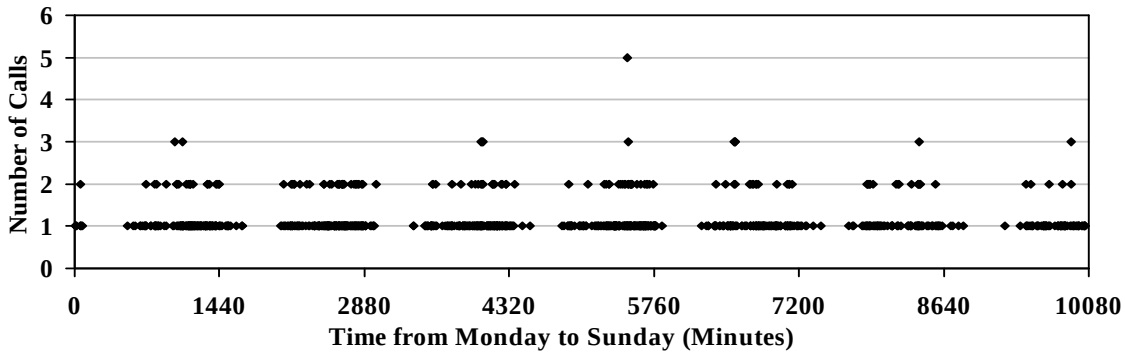

Figure 6.2: Traces of call starting times.

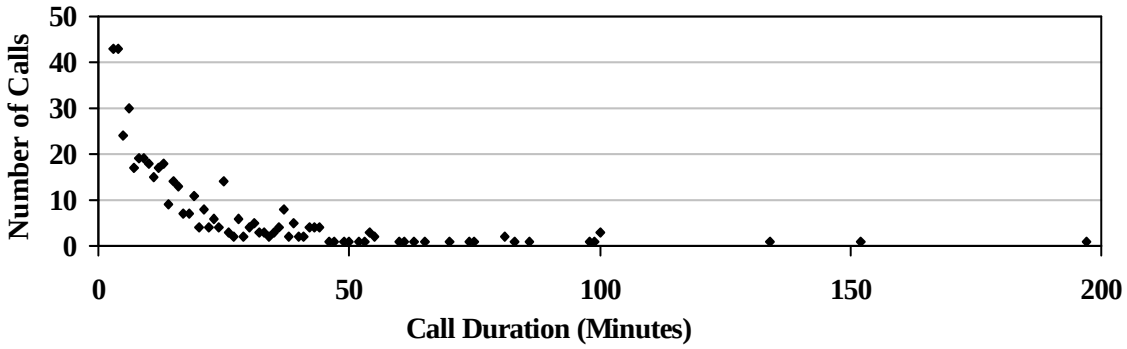

Figure 6.3: Traces of call durations[15].

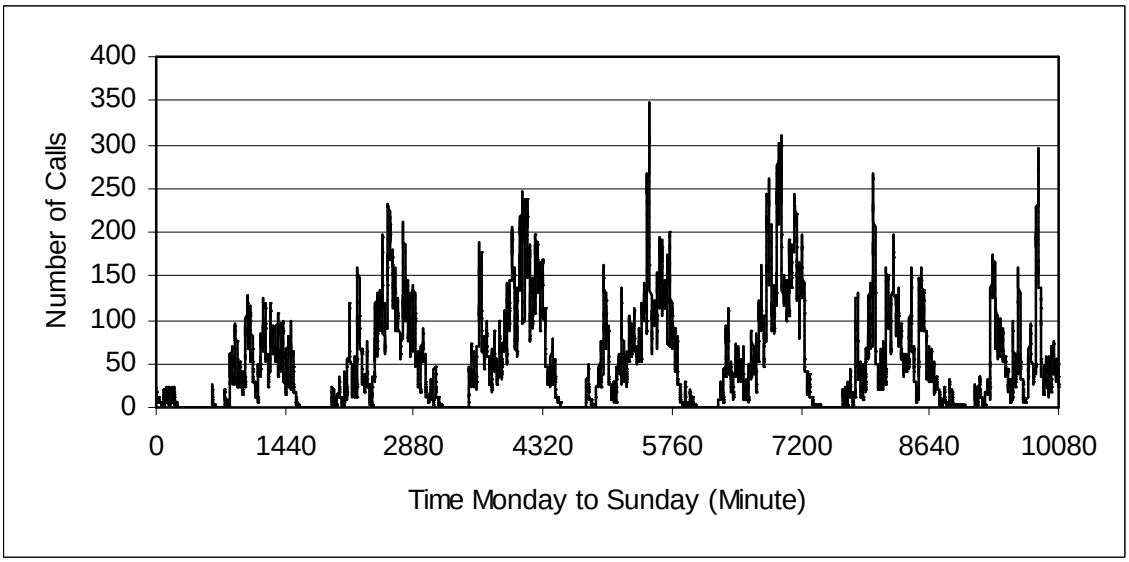

Figure 6.4: One week of calling pattern by 10,000 simulated users[16].

The second part of the user model is how a user would react to changing prices. We assumed that users know the minimum rate and the maximum rate for the price. Users also know that the price would not change in the first few minutes of a call, when it changes, it would change at most once every few minutes with a fixed increment each time. Users also know that the price increases only due to occasional congestion. With

[15] Not shown are calls lasting one minute (539 calls) and two minutes (139 calls).
[16] In the figure, users make 64,510 calls totaling 516,235 minutes.

this knowledge about the pricing policy, we assumed that a user would terminate his/her call with a certain probability after a price increase. Based on prior user experiments, we assumed that the probability is strongly dependent on the price increment[17]. Furthermore, we assumed that the user would not talk longer after a price decrease. Later on, we will conduct further user experiments to verify these assumptions about the user model. See Figure 6.5 below for a summary of the user model.

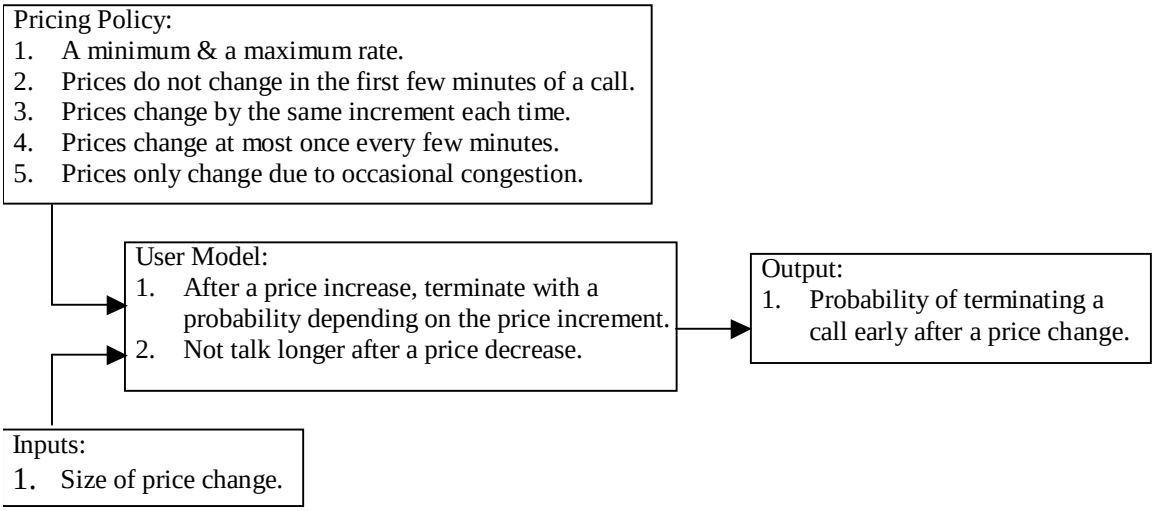

Figure 6.5: Summary of the user model.

6.2 Simulation Setup

For the workload model, we simulated the effects of congestion pricing when there is 5,000, 10,000, 50,000, and 100,000 users. For the pricing policy, we set the minimum rate to 10 tokens, the maximum rate to 40 tokens, and the price increment to 10 tokens. Based on prior user experiments, we expected that the probability a user would end his/her call within a minute of a 10 token price increase, *probability_end_10token*, to be 15%. However, we also experimented with *probability_end_10token* of 12% and 20%.

[17] A more detail model would have the probability depend on both the price increment and the price level.

In simulations, we varied the four parameters, *capacity*, *threshold*, *interval*, and *init*, and measured the overall call blocking rate and the percentage of the minutes used by users that would encounter a price change. See Table 6.1 for the settings of the simulation variables and the ranges of the parameters.

Table 6.1: Summary of the simulation variables and the parameter ranges.

Workload	**Range**
Number of users	5,000, 10,000, 50,000, 100,000
Pricing Policy	**Value**
Minimum rate	10 tokens/min
Maximum rate	40 tokens/min
Price increment	10 tokens
User Model	**Range**
Probability_end_10token	12%, 15%, 20%
Parameters	**Range**
Capacity	1 to maximum number of simultaneous calls
Threshold	0-100% of capacity
Interval	1-5 minutes
Init	1-5 minutes

6.3 Simulation Results

The most important parameter we found for affecting performance is the *threshold* value. We found that operators should set the *threshold* to 90% of the *capacity*. In general, congestion control mechanisms should only be applied when the load is close to *capacity*. Figure 6.6 shows the simulation results when there are 10,000 users, the *probability_end_10token* is set to 15%, the *interval* is set to three minutes, and the *init* is also set to three minutes. If a lower *threshold* value is used, then there would not be much improvement in call blocking rate. Call blocking rate cannot be reduced because of the sudden surges in load. On the other hand, if a higher *threshold* value is used, then the call blocking rate would quickly increase. Thus by setting the *threshold* value to 90% of the *capacity*, operators would not need to change prices frequently and would still be able to

obtain the lowest call blocking rate. We also found that there is a tension between call blocking rate and price announcement rate. If one wants to sharply reduce the price announcement rate, one would need to use a *threshold* value higher than 90%, but the call blocking rate would quickly increase. Figure 6.7 graphs the price announcement rate as the *threshold* value changes under the same settings.

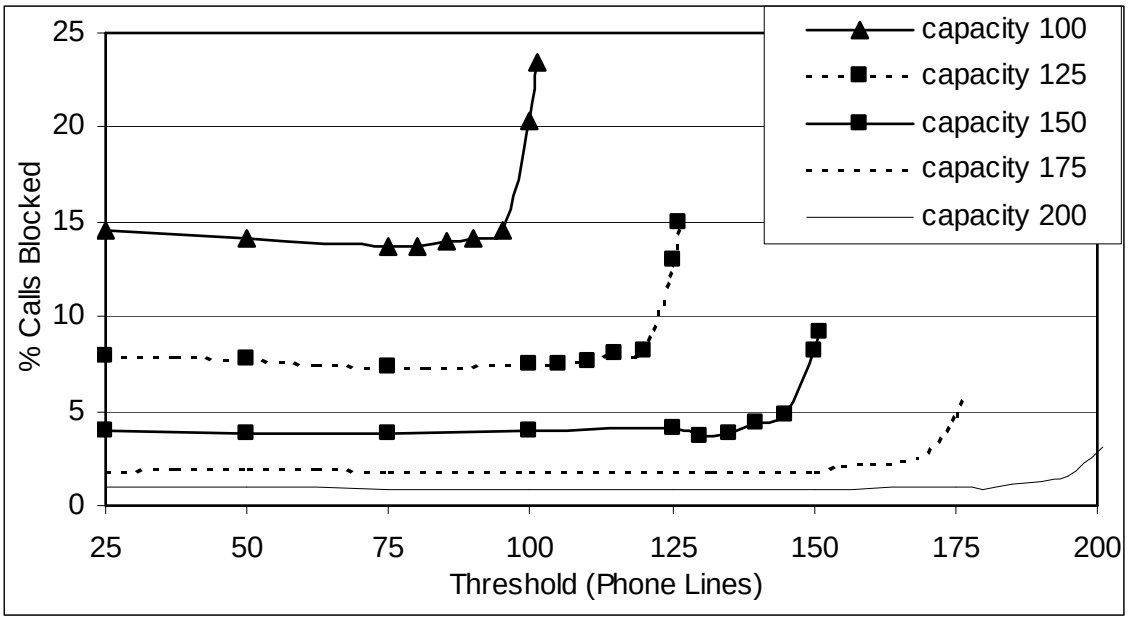

Figure 6.6: Call blocking rate for different threshold values.

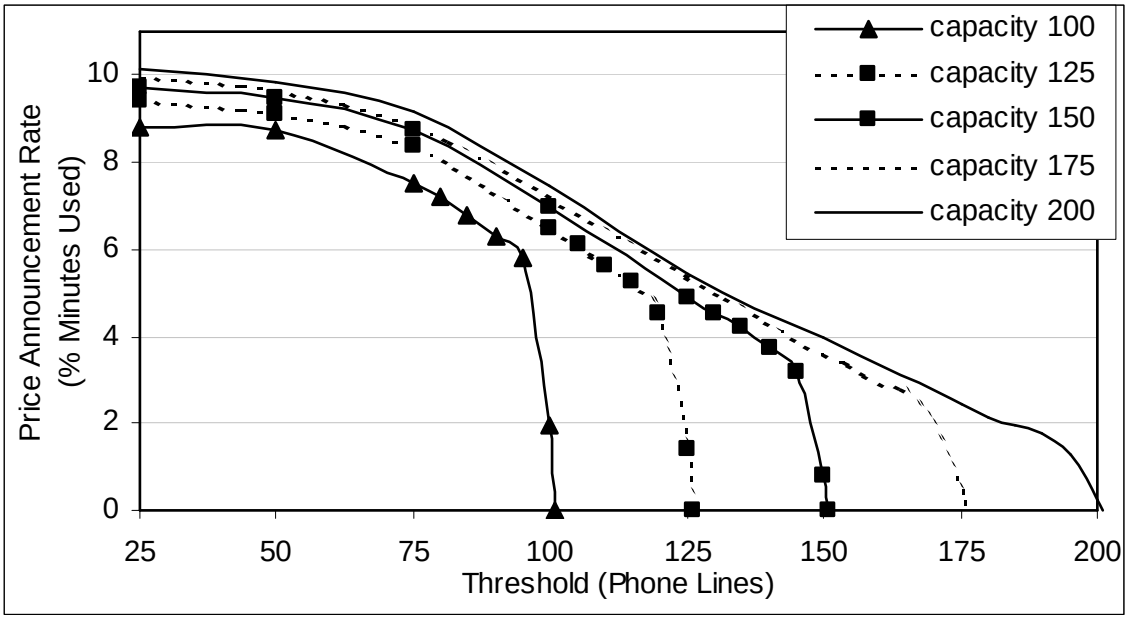

Figure 6.7: Price announcement rate for different threshold values.

We then varied the *interval* and the *init* parameters to observe their effects on performance. Figure 6.8 and Figure 6.9 show the call blocking rate and the price announcement rate as the *interval* and the *init* vary. In the figures, there are 10,000 users, the *probability_end_10token* is set to 15%, the *capacity* is set to 150 phone lines, the *threshold* is set to 90% of the *capacity*, the *interval* varies from 4 minutes to 2 minutes, and the *init* also varies from 4 minutes to 2 minutes. As we vary the *interval* and the *init*, we can gradually decrease the call blocking rate from 4.8% to 3.4%, but at the same time gradually increase the price announcement rate from 3.6% to 5.0%. From simulations and user experiments, a good setting for the *interval* is three minutes because it can cause users to respond to price changes while only slightly reduce the ability of dynamic pricing to reduce call blocking rate. For the *init*, three minutes is also a good compromise because it allows users to obtain low rates for short duration calls while reducing the incentives for users to hang up and call back again.

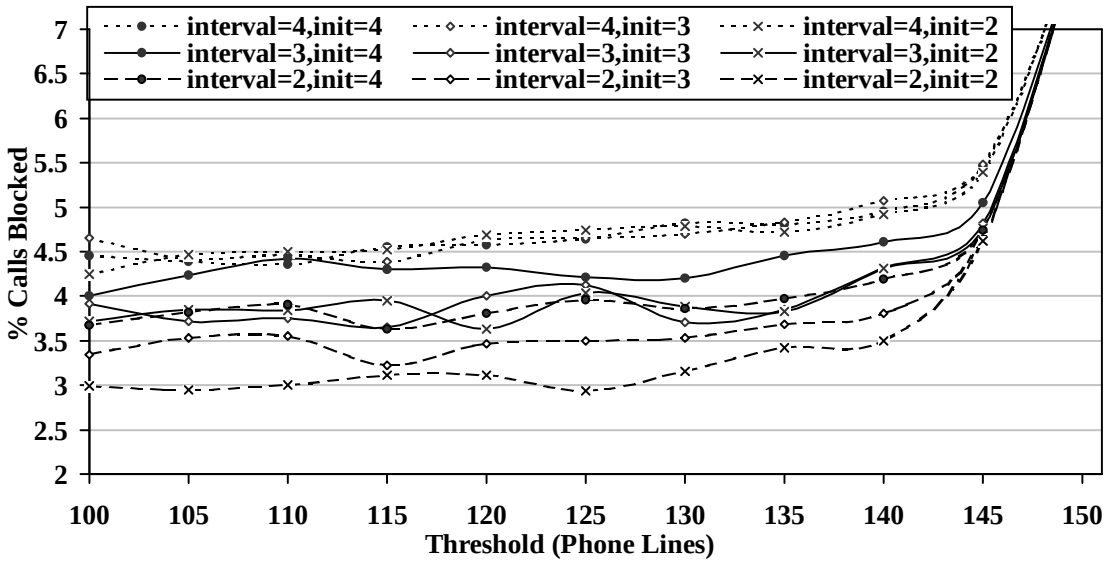

Figure 6.8: Call blocking rate as the interval and the init change.

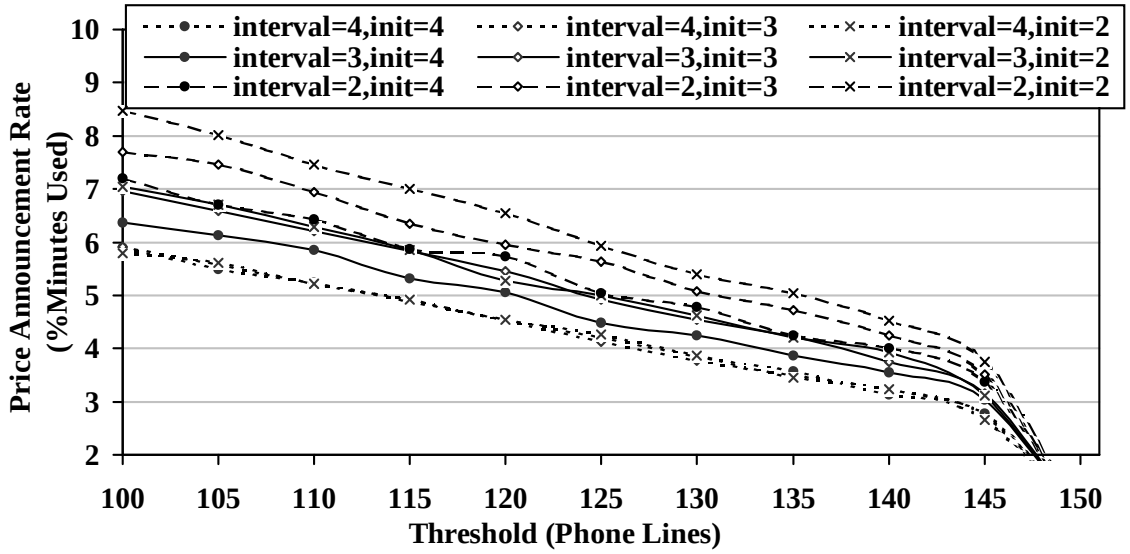

Figure 6.9: Price announcement rate as the interval and the init change.

Finally, operators should choose a *capacity* to meet their call blocking rate and price announcement rate requirements. In Figure 6.10 and Figure 6.11, we graph the call blocking rate and the price announcement rate that operators can expect when there are

10,000 users, the *probability_end_10token* is 15%, the *threshold* is set to 90% of the *capacity*, the *interval* is set to three minutes, and the *init* is also set to three minutes. For example, with 150 phone lines, an operator can expect a call blocking rate of 4.1% and a price announcement rate of 4.2%[18]. For the range of the user workload listed in Table 6.1, Table 6.2 summarizes the rules for operators to set the four parameters for managing congestion pricing.

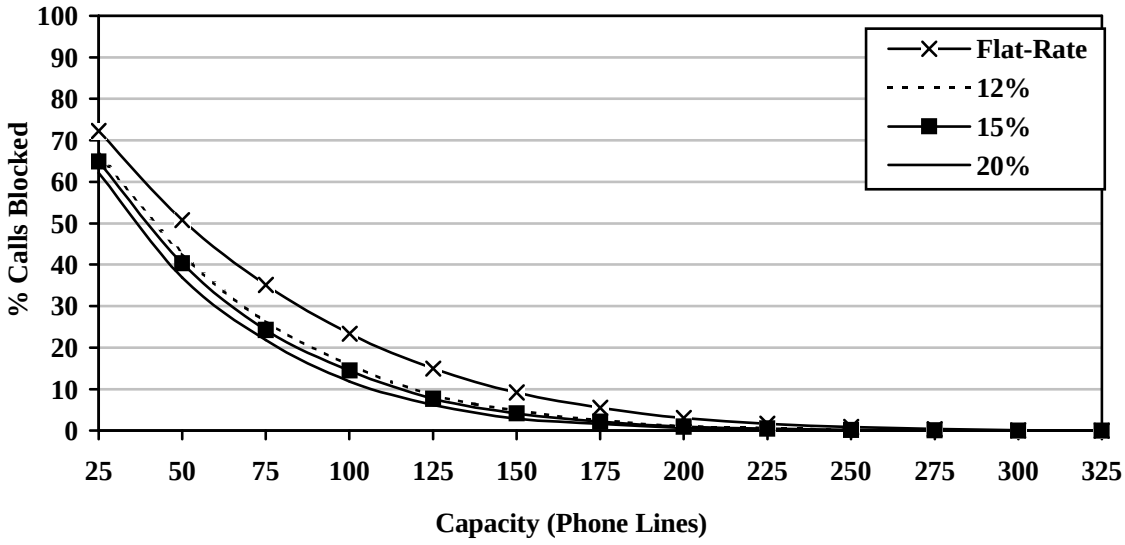

Figure 6.10: Call blocking rate for flat-rate and different probability_end_10token values under congestion pricing.

[18] Breakdown of the 4.2% is 2.9% for price increases and 1.3% for price decreases.

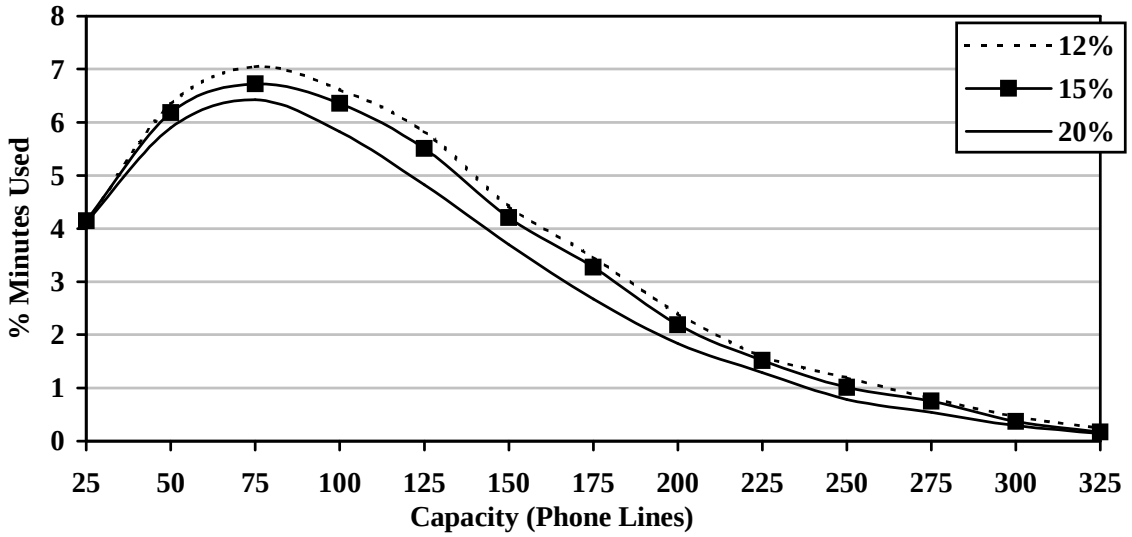

Figure 6.11: Price announcement rate for different probability_end_10token values under congestion pricing.

Table 6.2: Summary of the rules for operators.

Parameters	Rules
Capacity	Look at figures like Figure 6.10 to determine the capacity to support a certain call blocking rate for a given user population.
Threshold	Set to 90% of capacity because it can effectively reduce call blocking rate without causing unnecessary price increases.
Interval	Use three minutes because it is long enough to change user behavior and short enough to respond to congestion.
Init	Use three minutes because even though the first five minutes of a call are important to users, too long of a duration can encourage users to hang up and call back again.

Figure 6.10 and Figure 6.11 also show the expected benefits and drawbacks of congestion pricing over flat-rate pricing. For example, when there are 10,000 users, the *probability_end_10token* is 15%, an operator has 150 lines, he/she can expect the call blocking rate to be reduced by 50% (from 9.2% to 4.1%) or can save provisioning by 20% (from 150 lines to 120 lines). In terms of the price announcement rate, he/she can expect that the users would experience price changes in 4.2% of their usages. As shown in the figures, the call blocking rate and the price announcement rate are not very

sensitive to the *probability_end_10token* parameter in the user model. The benefits and drawbacks are limited because each day has only a few periods of extremely high utilization compared to the average, and price announcements can only be applied to calls longer than three minutes and have not had their prices changed in the last three minutes. Thus by using the appropriate parameters, congestion pricing will not cause many price changes, and at the same time would only be able to improve performance by a certain extent.

6.4 Validation

Our estimates of the benefits and drawbacks strongly depend on the form and the parameter of the user model derived when we randomly increased the price from an initial rate and then decreased it back. So we would like to further verify them by conducting user experiments where users would react to price changes under a more realistic setting (see Figure 6.12). More specifically, we would like the price changes to be caused by the actions of a large number of simulated users, O(10,000), and a small number of real users, O(100). However, the price changes are mainly determined by the actions of the simulated users because of their size. The simulated users would use the user model to make calls and react to price changes. As the operator, we would apply the appropriate congestion pricing parameters when managing many users and allow prices to slowly vary between a minimum rate and a maximum rate. Then we would observe if the real users still behave as predicted by the model.

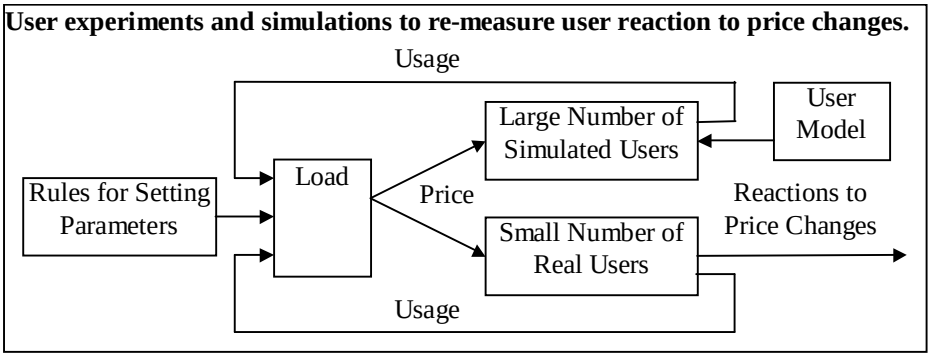

Figure 6.12: Setup for re-measuring user reactions to price changes.

In our re-measurement, we had 140 new dormitory students using the service along with 10,000 simulated users. We divided the real users into two groups, a test group and a control group, for conducting experiments (see Table 6.3). For the pricing policy, we set the minimum rate to 10 tokens, the maximum rate to 40 tokens, and the price increment to 10 tokens. We assumed that 15% of the time, a simulated user would terminate his/her call after a price increase. For the four parameters, we allocated the *capacity* with 150 phone lines, set the *threshold* at 135 phone lines (90% of the *capacity*), set the *interval* to three minutes, and set the *init* to three minutes. During the experiments, we observed the percentage of the real users' calls terminating after a price increase. From the real users, we had about 85 active users making 500 calls totaling 5000 minutes each week.

Table 6.3: Pricing policies for re-measuring the user model.

Base case: Initial rate of 10 tokens/min, minimum rate of 10 tokens/min, & maximum rate of 40 tokens/min. Prices change at most once every 3 minutes by 10 tokens each time.		
Week	**Group 1 Policy**	**Group 2 Policy**
10/1/01	Base case.	Base case.
10/29/01	Base case except price changes by 5 tokens each time.	Base case.
11/5/01	Base case except price change by 15 tokens each time.	Base case.

Our results further confirmed the form of our user model in that we can easily entice users to terminate their calls after a price increase if prices change slowly, not in the first three minutes, at most once every three minutes, and infrequently. During the experiments, users only encountered price announcements, price increases and price decreases, in about 4% of their usages. Furthermore, during the experiments, users did not talk longer after a price decrease. For the parameter of the user model, Group 1 terminated its calls more with a higher price increase (see Table 6.4 and Figure 6.13). With a price increase of 10 tokens, Group 1 terminated its calls early 15.7% of the time. However, during the same time, the control group, Group 2, consistently terminated its calls around 10% after a price increase of 10 tokens. Thus, to make a price increase of 10 tokens more effective, the price increment should vary instead of always being constant. Finally, Figure 6.14 compares the calling pattern during the experiments with that of flat-rate pricing. The figure further confirms that congestion pricing can also encourage users to defer some of their usages from peak times to off-peak times[19].

[19] The figure assumes both congestion pricing and flat-rate pricing carry the same number of minutes. However, as shown in our simulations, we estimate that congestion pricing can reduce peak-time traffic by 20%.

Table 6.4: Group 1's and Group 2' reaction to price increases.

Time After a Price Increase	**1**	**2**	**3**	**4**	**5**		**Number of Price Increases (% of Minutes)**	**Number of Price Decreases (% of Minutes)**	**Number of Price Changes (% of Minutes)**
Group1 (11/5/01) %Hang up After an Increase of **15**	25.00	2.08	6.38	0.00	0.00		64 (2.38)	32 (1.19)	96 (3.57)
Group1 (10/1/01) %Hang up After an Increase of **10**	15.73	12.00	9.09	0.00	6.67		89 (2.56)	40 (1.15)	129 (3.71)
Group1 (10/29/01) %Hang up After an Increase of **5**	10.53	10.29	6.56	0.00	0.00		76 (3.12)	20 (0.82)	96 (3.94)
Group2 (11/5/01) %Hang up After an Increase of **10**	10.20	13.64	2.63	2.70	0.00		49 (3.00)	18 (1.10)	67 (4.11)
Group2 (10/1/01) %Hang up After an Increase of **10**	8.33	3.90	5.41	5.71	1.52		84 (3.72)	52 (2.30)	136 (6.02)
Group2 (10/29/01) %Hang up After an Increase of **10**	11.76	10.00	3.70	0.00	0.00		34 (2.10)	15 (0.92)	49 (3.02)

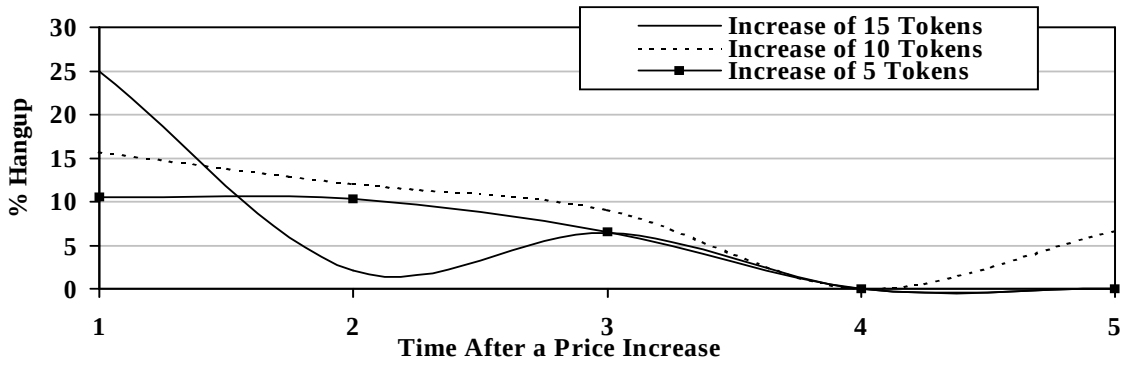

Figure 6.13: Group 1's reaction to different price increases.

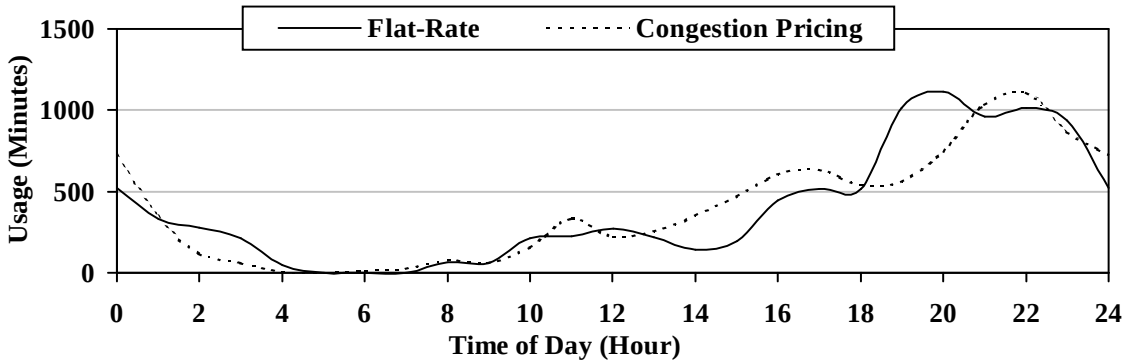

Figure 6.14: Flat-rate and congestion pricing calling pattern[20].

The estimates of the benefits and drawbacks (Figure 6.10 and Figure 6.11) depend on the *probability_end_10token*. Thus we would like to calculate the confidence of our measurements of user response to different price increases. For a certain price increase, let:

- N: number of samples of a price increase.
- X: number of samples that hang up within a minute of a price increase.
- P: probability that a user will terminate his/her call after a price increase.

By assuming each sample is independent of others and that the underlying probability distribution for all users is the same, then we can view P as a binomial process. From elementary statistics, the sample mean of P is X/N and the standard error of P is $\sqrt{(X/N)(1-(X/N))/N}$. In Table 6.5, we show the sample mean and the standard error of P, and also the number of samples required so that the sample mean of P will be at least two standard errors away from the mean of the lower price increase. Thus, we can be confident that users do react differently to different price increases. For the table, we

[20] Data contains 9,299 minutes of calls under flat-rate pricing and 14,110 minutes of calls under congestion pricing. The data from the congestion pricing is scaled to the data of the flat-rate pricing for comparison.

combine the data from Group 1 with the data from the Spring 2000 semester since both groups experience various price increases and have the same estimates for P. The standard errors of the measurements are small, thus we are confident of using 15% for the *probability_end_10token* in the user model.

Table 6.5: Sample mean and standard error of the probability that a user will terminate his/her call after a price increase.

Price Increase	Number of Samples	Probability of a User Terminating His/Her Call. Sample Mean (Standard Error)	Samples Required to be Two Standard Errors Away from the Next Lower Price Increase.
15 tokens	111	25.2% (4.1%)	88
10 tokens	157	15.9% (2.9%)	113
5 tokens	167	9.0% (2.2%)	320
No increase		5.8% (0.4%)	

6.5 Conclusion

We performed simulations to evaluate congestion pricing when there are many users using a voice-over-IP gateway service under a token constraint. We used the data from the prior user experiments to model when a user would make calls and how he/she would react to a price change. From simulations, we determined how we should set the congestion pricing parameters for managing thousands of users. Using the appropriate parameters, we estimated that congestion pricing can reduce call blocking rate by 50% or save provisioning by 20% while causing users to experience price changes in 4% of their usages. We further re-measured the user model by having real users using the service along with simulated users. We found that the real users' reactions to price changes is the same as specified in our user model. Furthermore, the user measurements confirm the parameter setting (*probability_end_10token*) in the user model. Thus, these results

increase our confidence in using the user model to estimate the benefits and drawbacks of congestion pricing.

Our study indicates that congestion pricing can be effective for voice calls because of users' calling pattern and reactions to price changes. The calling pattern is very bursty, is characterized by periods of high utilization and low utilization, and is dominated by short duration calls. Thus congestion pricing only needs to be applied during high utilization and to long duration calls. For reactions to price changes, from our user experiments, we found that a price increase can easily entice users to terminate their calls early if prices change neither rapidly nor in the first few minutes of a call. Furthermore, from simulations, if prices change slowly, it would only slightly reduce the effectiveness of congestion pricing. Thus operators can use congestion pricing to improve system performance (reduce call blocking rate by 50% or provisioning by 20%) at the cost of slight annoyance to users (encounter price changes 4% of the time). After finding congestion pricing to be effective for voice traffic, a single service class with fixed bandwidth requirement, we will apply congestion pricing to data traffic, multiple service classes with variable bandwidth demand, in the next Chapter.

Chapter 7 Applying Congestion Pricing to Data Traffic

We would like to understand how to apply congestion pricing to data traffic so that it would be acceptable to users and effective for operators. To investigate congestion pricing, we applied it to the simplest scenario of allocating access link bandwidth in a LAN. We prototyped two congestion pricing schemes and then deployed them to about 10 users. In the first prototype, we offered users three rate-limit sizes and charged users by the minute. We found that rate-limiting is too difficult of a mechanism for users to deal with short bursts and that charging by the minute places too much burden on users. After analyzing user usage pattern, in the second prototype, we offered users three levels of QoS that differ on degree of traffic smoothing and charged users at most once every 15 minutes. Traffic smoothing is different from rate-limiting in that it removes short-term fluctuations so that a user's load is closer to its long-term average. We found that this scheme is both effective and acceptable. It can effectively entice users to select a lower QoS, one with more traffic smoothing, by raising the price of a higher QoS. Based on surveys, users stated that selecting QoSs that differ on average performance and making purchases at most once every 15 minutes is acceptable. Using simulations, we estimated that in a LAN involving more users, if an operator can entice half of the users to have their traffic smoothed during congestion, then he/she can reduce the burstiness at the LAN access link by 20-30%.

In this Chapter, we report in detail the two congestion pricing schemes in Section 7.1 and Section 7.2. For each scheme, we describe the prototype, the evaluation, and the

follow-up analysis. After describing the two schemes, we conclude in Section 7.3 with the lessons learned.

7.1 First Experiment

7.1.1 Prototyping

General Scheme

In the first prototype aiming to quickly understand the issues involved, we offered users three sizes of connectivity to the Internet, SMALL, MEDIUM, and LARGE, and allowed them to switch between the sizes at anytime. We used bandwidth sizes because it is something easy and natural for users to understand. Using traces of our LAN under no traffic shaping, we calibrated the SMALL to 150K[21], the MEDIUM to 5M, and the LARGE to 10M. We selected these values because 150K would cover the majority of the bursts in the traces, 5M would then include another large fraction, and finally 10M would include almost all the bursts. The connectivity sizes are symmetric. Thus the SMALL allocation provides users with 150K for upload and another 150K for download.

To place a constraint on users, we decided to constrain users with free but limited tokens instead of charging real money[22]. Our experience from the voice traffic indicates that a free but limited token scheme can affect user behavior. However, we decided to constrain each user to 1000 tokens a day instead of a week[23]. Daily allocation of tokens is an appropriate length of time because it is not too long for users to wait if they run out of tokens. Furthermore, it is also not too short because it can still constrain user behavior by motivating them to conserve tokens over the allocation period. Thus, we replenished each

[21] All the bandwidth units are in bit per second.
[22] Charging real money would have made it difficult to find users to participate in our experiments.
[23] We will adjust prices so that 1000 tokens a day is a constraint to users.

user's tokens at 6AM. However, we decided to have the unused tokens from the previous day disappear to make congestion pricing more effective. There is a strong daily variation in a user' usages. Some days, a user would be a heavy user of bandwidth while others he/she would be a light user. Thus by not allowing tokens to accumulate, we can encourage the heavy users of a day to conserve during congestion while providing the light users with good quality. Thus we limited each user to at most a certain number of tokens a day for purchasing bandwidth.

Pricing Scheme

For the pricing scheme, the SMALL size is always free and users receive this allocation by default. We selected this strategy because with the chosen rate-limit levels, we expected that users would only need to occasionally request the MEDIUM or the LARGE. When using the larger sizes, users would then be charged with a certain number of tokens per minute. The charging rates for the MEDIUM and the LARGE are set so that most users would only have a few tokens left at the end of each day. Based on our experience with the voice traffic, we decided to use a charging granularity of one minute. If the charging granularity is too short, then users would need to continuously make purchases. If it is too long, then it could not modify user behavior enough to be effective in reducing congestion. To charge users, we defined a *charging session* as when a user has requested something other than the SMALL and ends when the user is using the SMALL allocation again. During a charging session, a user would be charged by the minute using the price at the beginning of a minute.

Users can make and change their purchases at anytime. However, charges are synchronized to the beginning of a charging minute to make accounting simple. Thus

when a user performs an upgrade, it would occur immediately and the user would be charged as if he/she has requested the upgrade at the beginning of the current charging minute. On the other hand, when a user selects a downgrade, it would take effect the next minute because the user has already paid for the current minute. To minimize user involvement, we decided to keep charging users at the current size until they have made a change. However, after some initial usage trials, we found that it is easy for users to request upgrades, but harder for them to remember to downgrade. Thus, in addition of being able to select the SMALL at anytime, users would be automatically downgraded to the SMALL if their usages have been idle. After fine-tuning with a few users, we set the idle as when both the upload traffic and the download traffic have been less than 80% of the SMALL for more than three minutes[24].

We decided to experiment with two variations of dynamic pricing, *per-session* congestion pricing and *per-minute* congestion pricing to understand if the frequency of price changes might cause users to become unresponsive to dynamic pricing as was the case for voice traffic. Prices charged under the *per-session* congestion pricing will remain the same throughout a charging session while prices charged under the *per-minute* congestion pricing can be different for each minute of a session. The prices for both variations are set according to the load, incoming plus outgoing traffic, of the access link as shown in Table 7.1 and Table 7.2. The price levels are chosen initially based on the access link load traces. However, we planned to adjust the price levels after obtaining some user evaluation. To make it easier for users to remember the prices, we decided to make the price of the MEDIUM always half of the LARGE. For the *per-session*

[24] Later on, users indicated that it would be better if they can dynamically change the idle interval to suit their needs.

congestion pricing, users are charged with some amount for all the access link load levels because a charging session can last for a while. However, for the *per-minute* congestion pricing, we can charge users zero when the access link load is low because the charge only applies to the current minute. The access link load is updated once every five seconds due to a data testbed limitation[25]. To provide users with more time to reflect on a price change, e.g., when the users are at the SMALL, we made each price change last at least ten seconds before allowing to change again.

Table 7.1: Prices and color indicators of the per-session congestion pricing as a function of the access link load.

Access Link Load (Mbps)	Small (Tokens/Min)	Medium (Tokens/Min)	Large (Tokens/Min)	Color
0-5	0	10	20	GREEN
5-10	0	15	30	YELLOW
10-20	0	20	40	ORANGE
20-100	0	25	50	RED

Table 7.2: Prices and color indicators of the per-minute congestion pricing as a function of the access link load.

Access Link Load (Mbps)	Small (Tokens/Min)	Medium (Tokens/Min)	Large (Tokens/Min)	Color
0-1	0	0	0	GREEN
1-5	0	10	20	YELLOW
5-10	0	20	40	ORANGE
10-100	0	30	60	RED

User Interface

The user interface after a user has logged in is shown in Figure 7.1. It provides the user with his/her current rate and last minute average for both download and upload

[25] As explained in Chapter 3, the update frequency for the data testbed is limited by the processing on the PacketShaper, the traffic shaping appliance.

traffic. It also indicates to the user his/her current purchase, current charge, time left on the current purchase, and tokens left. The information on the user interface is updated once every five seconds.

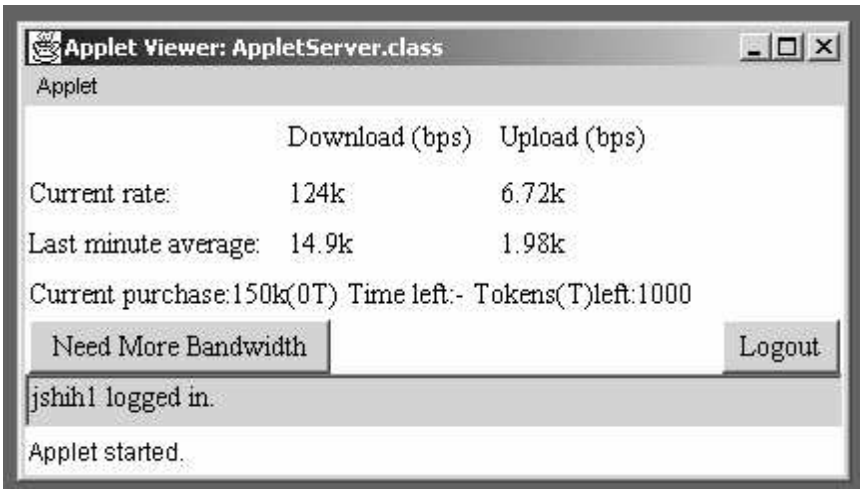

Figure 7.1: Initial user interface of the first prototype.

When a user wants to increase his/her bandwidth allocation from the SMALL, she/he will need to press a "Need More Bandwidth" button to see the current prices of the MEDIUM and the LARGE (see Figure 7.2). The prices will also be indicated by the colors mentioned in Table 7.1 and Table 7.2. The button will reappear if the user has been at the SMALL for more than one minute without making a purchase. We required users to press the button because we wanted to measure the user response time between looking at prices and making purchases. With the response time, we measured how responsive congestion pricing can be for dealing with network traffic bursts.

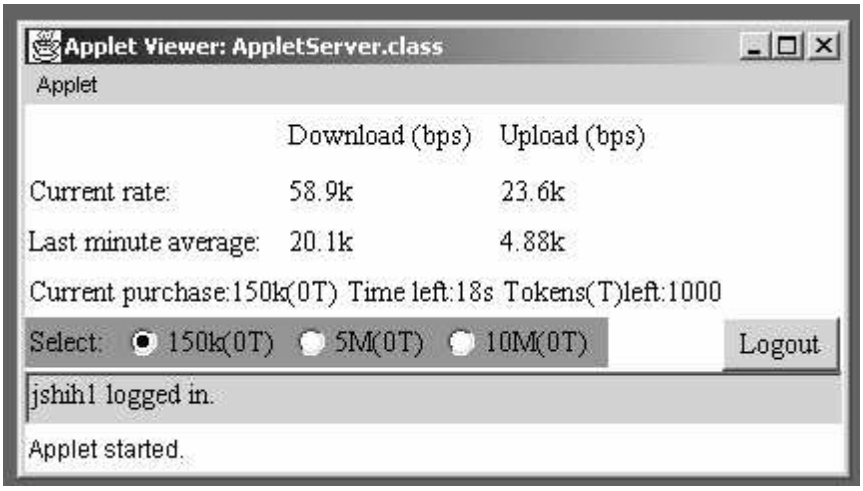

Figure 7.2: User interface of the first prototype after pressing the "Need More Bandwidth" button.

Since we automatically downgraded users to the SMALL when they are idle, we also wanted to remind them whenever they might want to upgrade. Thus we provided users with a pop-up window (see Figure 7.3) whenever their usages, either upload or download traffic, go from 80% to 90% of their current purchases. The pop-up window will disappear after one minute if users have not already closed it.

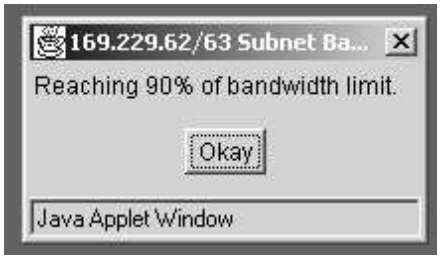

Figure 7.3: Pop-up window of the first prototype to remind users to upgrade.

7.1.2 Evaluation

Using the initial prototype, we had 12 users of our LAN testbed, graduate students and professors from our research group, agreed to participate in our study. We did not provide them with any special benefit for participating and they can drop out of the study

for any reason, e.g., if they feel that it is disrupting their work. We asked them to participate for four weeks, two weeks under unlimited tokens to get familiar with the user interface, one week under *per-session* congestion pricing, and one week under *per-minute* congestion pricing. We used the data from the first two weeks to calibrate the price levels for the last two weeks. We did so by observing how frequent users would request the MEDIUM and the LARGE to determine if the 1000 token allocation and the price levels are a constraint. At the end, we had 10 users completed all four weeks of the evaluation[26].

There were two main problems with our scheme of using rate-limiting and charging once every minute. First, users were not able to accurately adjust their purchases to their usages. Most of our users' usages were short bursts around ten seconds followed by long idle time. However, it took users some time to realize that they need more bandwidth and need to make purchases[27]. There were many instances when users' short bursts ended just before bandwidth purchases. There were also many instances when users requested upgrades anticipating a burst, but the burst never occurred. Thus the correlation between usages and purchases was surprisingly low. There was even a lower correlation between usages and the correct level of purchases. Typically, users would know that they need to make an upgrade, however it is hard for them to really know whether they need a MEDIUM or a LARGE. Thus offering several rate-limiting levels did not meet user needs. Our experience suggests that we need to provide users with tools to help them make purchases. For example, rules to automatically purchase the first minute when usage is above a certain level and visualizations to help users decide which

[26] One user dropped out because he/she needs to perform network measurements under no traffic shaping. Another dropped out because of a security concern not related to the experiment.
[27] As a note, after a user realizes that he/she needs more bandwidth and presses the "Need More Bandwidth" button, it usually takes only 1-2 seconds for her/him to decide on a purchase.

bandwidth sizes they need. The second problem with our scheme was that charging users once every minute required too much user involvement. Some of our users needed to request 20-35 purchases a day. From surveying the users, they complained that with a charging granularity of one minute, they needed to constantly worry about the price levels and tokens left. See Appendix B for the survey questions and responses. Thus, users did not like our initial scheme and strongly asked for tools to automate their purchasing decisions.

There were three other problems with our initial prototype. First, users did not like being rate-limited to the SMALL when they are out of tokens. Second, some users complained that their automatic jobs, e.g., daily backups, could not be run using the SMALL allocation if they are not around to make purchases. Third, some users were running web servers or ftp servers on their computers; they were concerned that traffic to these servers might be constrained[28].

7.1.3 Analysis

There are two issues with automating purchases. First, it is not simple to come up with rules that are easy for users to specify. The rules would need to depend on the current usage, price levels, tokens left, time till token refresh, etc. Furthermore, the rules would need to take into account tasks users are performing when estimating how users would value bandwidth. Second, for congestion pricing to work well with automated rules, there must be enough heterogeneity among users so that they would specify different preferences. For example, it would only work if heavy users of bandwidth would specify in their preferences to conserve when prices are high while light users of

[28] Thus, in the second experiment, we configured the PacketShaper not to count the traffic to users' web servers or ftp servers.

bandwidth would specify to continue no matter the prices. With our user group, we were not sure if there is enough heterogeneity in their preferences because their usages are quite similar. Thus it is not easy to use rules or preferences to remove users from the control loop and still have dynamic pricing be effective.

We then decided to explore a different design space of using traffic smoothing. Instead of offering different connectivity sizes and automating purchasing rules, we investigated using traffic smoothing to better handle the bursty nature of our users' traffic, short-duration bursts lasting about ten seconds. From analyzing user traffic, we found that congestion at an Ethernet access link is usually caused by large, but short-duration, bursts generated by one or two sources at a time. Based on usage data, a source would normally generate bursts less than 500Kbits. These small bursts do not cause problem for the access link. However, once in a while, a source would generate a large burst greater than 1M. It is these large bursts that are causing the access link to be bursty. Thus, if these large bursts can be smoothed out over a longer time period, but not too long so that they do not frequently overlap with other large bursts, then the access link bursts can be reduced.

To understand the effects of smoothing for a larger user group, we performed simulations using a self-similar Ethernet traffic generator from Glen Kramer at UC Davis[29]. Using the generator, we simulated an environment where we have 40 users on a 100Mbps Ethernet where the average network utilization is 30%. After generating one million packets, we first aggregated the traffic from each user on a 10ms granularity. We then removed the first 200 samples to eliminate the cold start effect and kept the

[29] It generates self-similar traffic by aggregating multiple sources of Pareto-distributed on and off periods. See http://wwwcsif.cs.ucdavis.edu/~kramer/code/trf_gen1.html for more information.

remaining 13657 samples. We used this data set for our analysis because the generator does not produce good self-similar traffic on larger time-scales like seconds and minutes. Since Ethernet traffic is self-similar on many time-scales [23], the data set can be used to represent bursts at larger scales by changing each sample's time-scale.

We first simulated how well smoothing function like exponential average [34] would work. An exponential average sets the current burst limit as a weighted average of the current load and the last burst limit.

- $\text{Limit}_t = \alpha\ \text{Load}_t + (1-\alpha)\ \text{Limit}_{t-1}$

If the current load is greater than the current burst limit, then the unsent load is added to the next period's load. Thus the exponential average smoothes out short-term fluctuations, so users can still send the same amount of traffic as under no smoothing. We found that by applying the exponential average to each user's traffic, we can easily reduce the access link bursts by 6-60% (see Table 7.3 and Figure 7.4). In the figure, the black lines are load when there is no smoothing and the white lines are load when smoothing is applied. The figure shows that smoothing can generally reduce the peak and the burstiness of the access link load. Our simulation experiments indicated that if we only apply the exponential average with α equals to 0.1 to half of the users, representing those who are responsive to price increases, then we can still reduce the access link bursts by 20%. Furthermore, if we apply a heavy smoothing, α equals to 0.1, to half of the users who are responsive, and light smoothing, α varies from 1.0 to 0.7, to the other half who refuse to respond, then we can reduce the bursts by 20-30% (see Table 7.4 and Figure 7.5). Other researchers [12] have also found similar performance improvements when applying traffic smoothing to actual Ethernet traces.

Table 7.3: Effect of traffic smoothing when varying α of the exponential average.

α	Reduction in Standard Deviation of Bursts at Access Link
0.9	6.0%
0.7	17.2%
0.5	28.3%
0.3	41.0%
0.1	59.8%

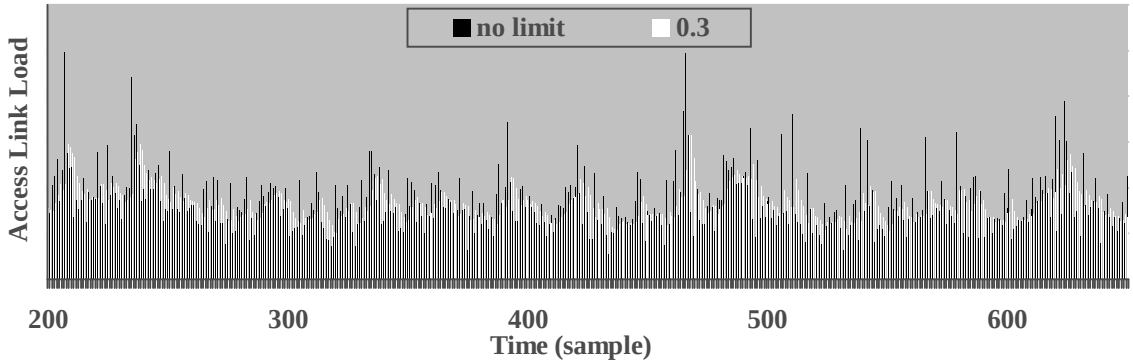

Figure 7.4: Illustration of traffic smoothing when α equals 0.3.

Table 7.4: Effect of traffic smoothing when half of the users are responsive to price increases and the other half are unresponsive.

Responsive(α)	Unresponsive(α)	Reduction in Standard Deviation of Bursts at Access Link
0.1	1.0	22.3%
0.1	0.9	26.5%
0.1	0.8	30.4%
0.1	0.7	34.1%

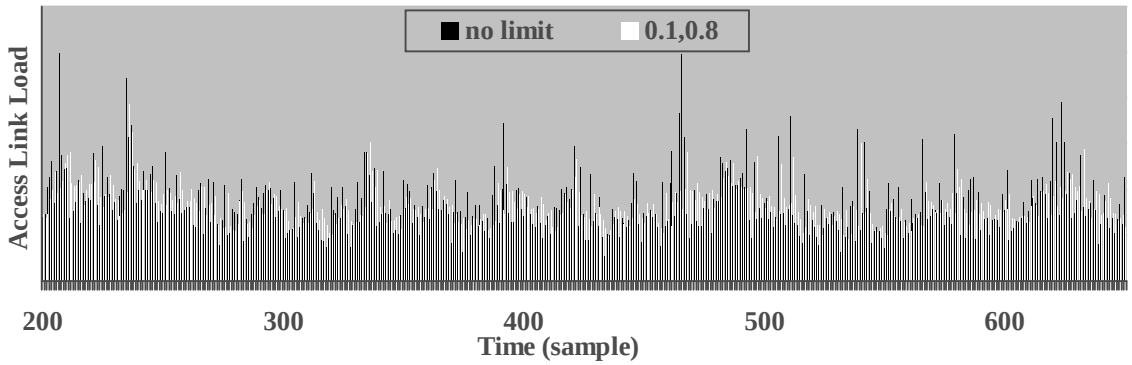

Figure 7.5: Illustration of traffic smoothing when half of the users are responsive (α=0.1) to price increases and the other half are unresponsive (α=0.8).

Our Packeteer PacketShaper, the traffic shaping appliance in our data testbed, does not have a smoothing function like an exponential average, however, it can adjust the rate limits applied to each source once every few seconds. Thus we performed simulations to determine if we can emulate traffic smoothing using our PacketShaper. We used a finite set of rate-limit levels so that we do not need to change the rate limits at the PacketShaper frequently. We set the levels to 100K, 200K, 400K, 800K, 1.6M, 3.2M, 6.4M, 12.8M, 25.6M, 51.2M, 102.4M, doubling each time. Figure 7.6 summarizes the algorithm for emulating smoothing with our PacketShaper. When load increases, we would stay at each level for a few seconds before moving up to the next level. Similarly, when load decreases, we would stay at each level for a few seconds before moving down. We performed simulations assuming that each sample of our data set is one second. We found that by adjusting the length of time at each level when load increases, we can easily emulate different degree of traffic smoothing (see Table 7.5 and Figure 7.7).

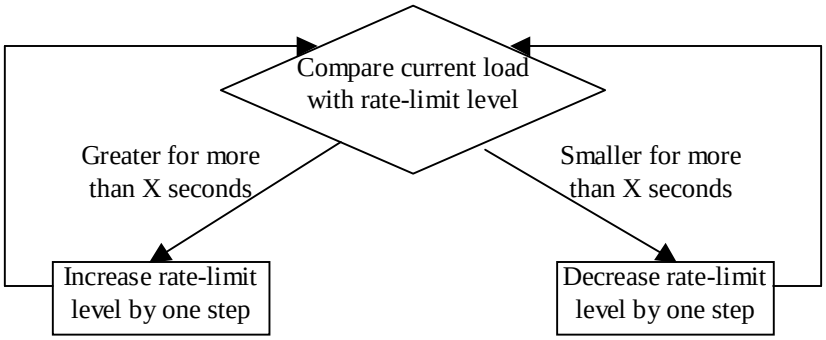

Figure 7.6: Algorithm for emulating smoothing using the PacketShaper.

Table 7.5: Effect of traffic smoothing by spending different amount of time at each rate-limiting level.

Time at Each Level When Load Increases (Sec)	Time at Each Level When Load Decreases (Sec)	Reduction in Standard Deviation of Bursts at Access Link
1	1	9.8%
3	1	25.5%
5	1	30.8%
10	1	38.4%

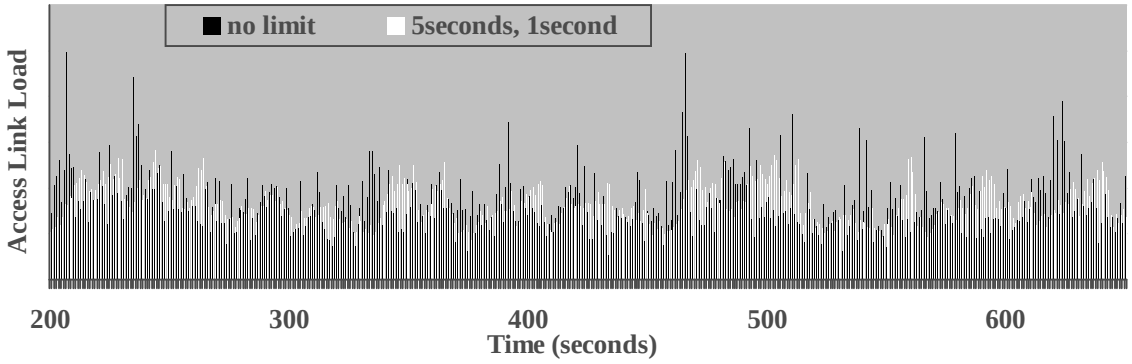

Figure 7.7: Illustration of traffic smoothing by spending 5 seconds at each level when load increases and 1 second at each level when load decreases.

We then performed simulations to understand the appropriate charging granularity to use. Using traces from the *per-session* congestion pricing and the *per-minute* congestion pricing, we assumed that a user is active if his/her upload or download traffic is above 120K. We picked 120K because all the bursts in the traces under the SMALL

limit, 150K, are not shaped. For a rough estimate, we assumed that a user would make a purchase if his/her usage is active. Then for different charging granularity, we wanted to know how often a user would need to make another purchase within a minute of the end of a purchase. We found that charging users once every 10-15 minutes is reasonable because users would not need to make frequent repeated purchases (see Table 7.6).

Table 7.6: Effect of different charging granularity on likelihood of a repeat purchase.

Charging Granularity (Min)	Active Within 1 Min (Per-Session Trace)	Active Within 1 Min (Per-Minute Trace)
1	48.6%	58.2%
5	25.1%	32.5%
10	20.1%	26.2%
15	15.1%	18.9%
20	16.2%	16.2%

We also performed simulations to understand the effect of a longer charging granularity on operators' ability to reduce access link bursts. We assumed that in the beginning of a charging session, an operator can use prices to entice users to have their traffic shaped. When applying shaping, the operator uses a moving average with α equals to 0.1. We assumed that the operator uses a simple heuristic, like if the current access link load is above 30% of the link capacity, to decide whether to entice users to have their traffic smoothed. Thus, when there is no congestion, no traffic shaping is used, but when there is congestion, the operator will start applying shaping to users. Using the generated Ethernet traces of 40 users mentioned before, we found that by changing user behavior once every 10-15 minutes as opposed to once every minute, operators only slightly reduce the effectiveness of dynamic pricing (see Table 7.7). Furthermore, a 10-15 minutes charging granularity is effective for operators because Ethernet traffic is observed to be bursty even at these time-scales [23].

Table 7.7: Effect of different charging granularity on reducing access link bursts.

Charging Granularity (Min)	Reduction in Standard Deviation of Bursts at Access Link
1	44.6%
5	39.9%
10	37.0%
15	35.1%
20	34.2%

In summary, from analyzing data and performing simulations, we decided to use our PacketShaper to emulate traffic smoothing and charging once every 10-15 minutes for the next experiment.

7.2 Second Experiment

7.2.1 Prototyping

General Scheme

In the second prototype, users are given three classes of service, RESPONSIVE, MODERATE, and SLOW-GOING, that differ on degree of traffic smoothing. These classes are based on a common set of rate-limit levels, but differ on the amount of time spent at each level when load increases (see Table 7.8). These levels and time durations are fine-tuned so that our users can easily differentiate between the three classes. 49K, 70K, and 140K are added so that users can better distinguish between the classes under low bursts. The levels are capped at 12.8M because our users rarely send bursts higher than that. For the time durations, when load increases, the RESPONSIVE stays at each level for 3 seconds before increasing to the next level. For the MODERATE, it stays about 6 seconds, and for the SLOW-GOING, it stays about 9 seconds. When load decreases, all three service classes stay at each level for only 3 seconds before dropping to the next lower level. The performance charts of the three service classes are shown in Figure 7.8 and Figure 7.9. For example, to transfer a 500Kbits web page when usage is

idle, the RESPONSIVE allocation will take four seconds, the MODERATE one will take six seconds, and the SLOW-GOING one will take nine seconds.

Table 7.8: Time in seconds at each level when load increases.

Level (Mbps)	**0.049**	**0.07**	**0.1**	**0.14**	**0.2**	**0.4**	**0.8**	**1.6**	**3.2**	**6.4**	**12.8**
SLOW-GOING (Sec)	6	6	6	6	6	9	9	9	9	9	9
MODERATE (Sec)	0	3	3	3	3	6	6	6	6	6	6
RESPONSIVE (Sec)	0	0	3	0	3	3	3	3	3	3	3

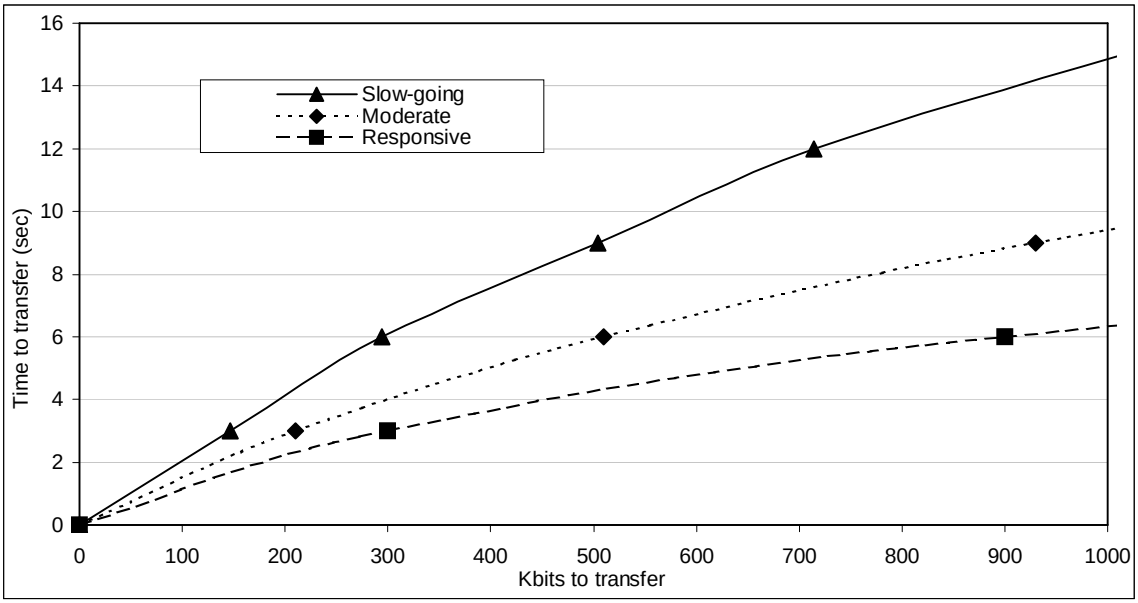

Figure 7.8: Performance of different QoSs when transferring less than 1M.

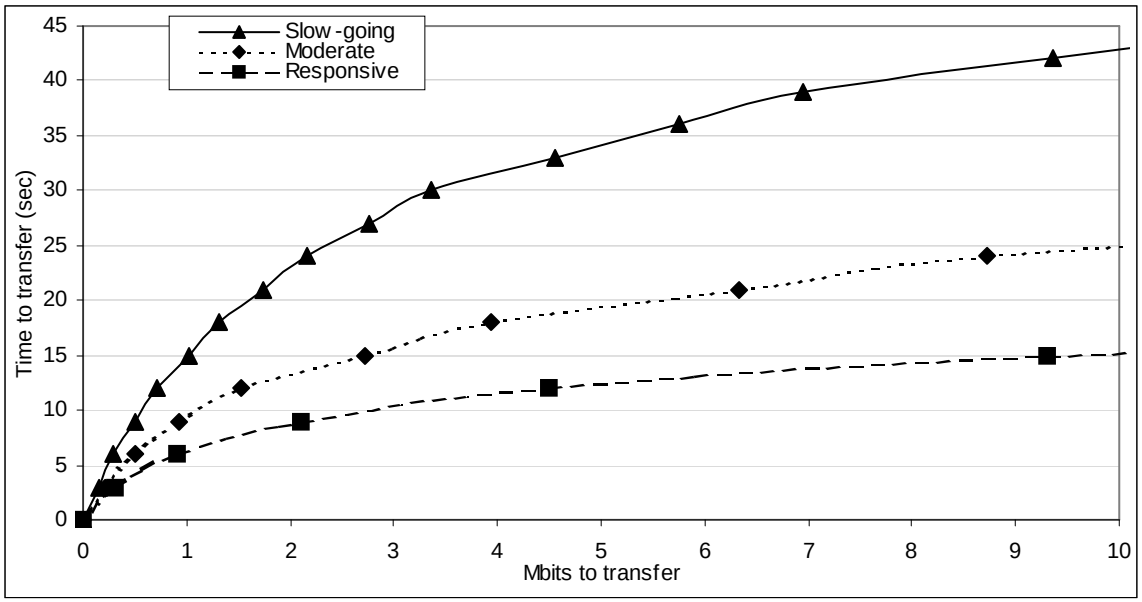

Figure 7.9: Performance of different QoSs when transferring less than 10M.

Pricing Scheme

Based on prior analysis, we conservatively set the charging granularity to 15 minutes to minimize user involvement. Thus once a user has purchased a QoS, he/she would have that QoS for the next 15 minutes. Each user is given 24 tokens a day. We decided to use a small token amount and price levels so that users can better anticipate how long their budgets would last. We found that using a large number like 1000 tokens in the first evaluation makes it harder for users to really understand their purchasing limits. The SLOW-GOING is always free and users are allocated with it by default and whenever they run out of tokens. We decided to keep the MODERATE price constant and just vary the RESPONSIVE price to entice users to change their purchases. By varying just one price, users are better able to comprehend dynamic pricing. Thus we charged the MODERATE 1 token/15 min. With 24 tokens, a user can purchase 6 hours of the MODERATE. We then charged the RESPONSIVE between 2 to 6 tokens/15min.

Thus with 24 tokens, a user can purchase 1 to 3 hours of the RESPONSIVE. Finally, when the price of the RESPONSIVE changes, e.g., when users are using the SLOW-GOING, we made sure that it changes to a user at most once every 15 minutes.

Users can make purchases or upgrades at anytime. When upgrades occur, they take affect immediately and users are charged as if they have made the upgrades since the beginning of their purchases. When downgrades occur, they have no effect because users already paid for the higher qualities. By rounding charges to a charging period, accounting becomes simpler. After a purchase expires, users are automatically returned to the SLOW-GOING. We made this design decision because from our analysis in the first experiment, we do not expect users to have to make frequent repeated purchases.

User Interface

The user interface provides users with real-time status information as shown in Figure 7.10. First, each user can observe his/her download and upload rate. Furthermore, each user can also see the current rate limit that the PacketShaper is using to shape his/her traffic. Thus the user can know whether his/her slowdown is due to shaping at the access link or elsewhere on the Internet. Finally, each user is informed of his/her current purchase, current charge, time left on the current purchase, and tokens left. The information on the user interface is updated once every 3 seconds[30].

[30] The frequency is limited by the rate the information can be polled from the PacketShaper.

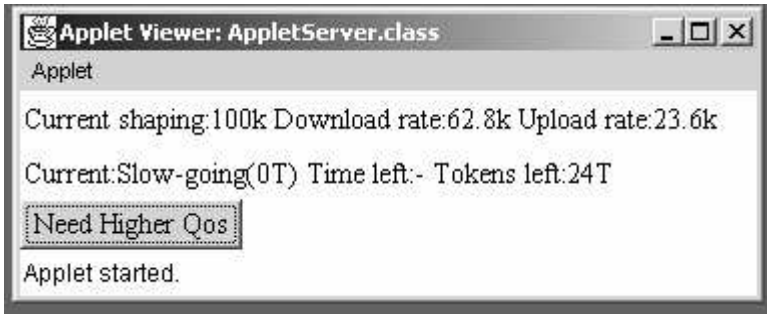

Figure 7.10: Initial user interface of the second prototype.

Users need to press a "Need Higher QoS" button to make purchases. After pressing the button, users will see the price for the RESPONSIVE and can make purchases (see Figure 7.11). To help users notice the RESPONSIVE price, colors listed in Table 7.9 are used to help indicate the RESPONSIVE price level. The button will reappear after one minute if users have not made a purchase. There were several reasons for using the button. First, we wanted to measure the user response time between looking at the RESPONSIVE price and making a purchase. Second, we wanted to hide the price so that users do not need to constantly think about it[31]. Third, we did not want to use intrusive mechanisms, like beeps or flashes, to inform users whenever the price has changed. It is much simpler for users to request the RESPONSIVE price when they want to make a purchase.

[31] Based on the first experiments, some users indicated that it is stressful to have prices changing in a background window.

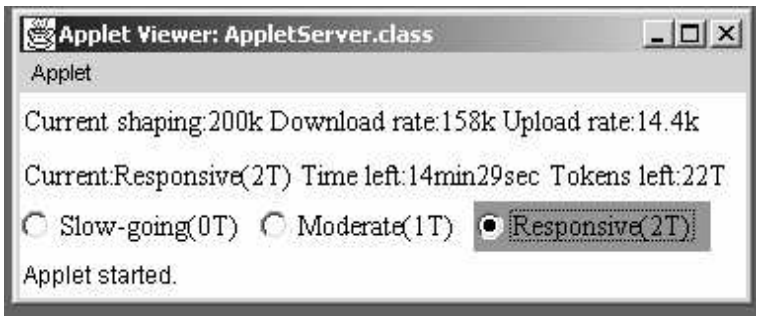

Figure 7.11: User interface of the second prototype after pressing the "Need Higher QoS" button.

Table 7.9: Colors used to indicate the RESPONSIVE prices.

Responsive Price (Tokens/15Min)	Color
2	GREEN
4	YELLOW
6	RED

For convenience, after a purchase has expired, a pop-up window will appear to inform users of the change in the RESPONSIVE price and to allow them to make another purchase (see Figure 7.12). The pop-up window will disappear after one minute if users have not already closed it.

Figure 7.12: Pop-up window in the second prototype after a purchase has expired.

7.2.2 Evaluation

For the second prototype, we had eight users, once again students and professors, who agreed to participate. We had fewer users because it was closer to the end of the semester and many people were busy. We asked them to use the service for a few days to

get familiar with the congestion pricing scheme and a week for us to collect detailed data. All eight users completed the experiment.

The goal of the experiment was to understand if varying prices can entice users to change their behavior. Since we were offering users QoSs with known average performance, we decided not to worry about adjusting prices according to access link load and instead change them using a heuristic. We set the price of the RESPONSIVE so that 50% of the time, it would be 2 tokens/15min, 25% of the time, it would be 4 tokens/min, and 25% of the time, it would be 6 tokens/15min. However, we did not inform users that the prices are set artificially. With the above setup, we found that we can easily entice users to choose a lower QoS, MODERATE, by increasing the price of a higher QoS, RESPONSIVE (see Table 7.10). During the experiment, heavy users of bandwidth were more sensitive to prices while light users, with lots of tokens available, selected the RESPONSIVE most of the time. Thus by changing prices, we could easily entice the heavy users to select a lower QoS that has more traffic smoothing.

Table 7.10: Percentage purchasing the RESPONSIVE at different prices.

Price of the RESPONSIVE (Tokens/15Min)	Number of Samples	% Purchase the RESPONSIVE
2	118	55.1%
4	47	23.4%
6	65	13.8%

Using surveys, users did like this new scheme more and would be willing to use it. They stated that they would use it if their DSL or cable modem providers offer similar pricing scheme. They liked having to interact with the service at most once every 15 minutes and only have to select from three levels of QoS. From the surveys, users indicated that they do look at the RESPONSIVE price, mainly by its color indicator, and

the tokens left when making purchases. From the usage data, most of the users judiciously used their tokens, knowing that unused tokens will disappear at the end of a day, so as to just have a few tokens left each day. Users also mentioned that 24 tokens a day and the price levels do place a reasonable, but not burdensome, constraint on them. See Appendix B for the survey questions and responses.

There are also a few other benefits of using traffic smoothing. First, users are more willing to use the SLOW-GOING when out of tokens because eventually they would be able to download large files. Second, with the gradual ramp up of traffic smoothing, users can still use the SLOW-GOING to run background jobs when they are not in their offices to make purchases.

7.2.3 Analysis

We used traffic smoothing instead of rate-limiting because our users' usages were dominated by short-duration bursts. However, in a large network with more users, user usages would have both short-duration bursts, like web surfing, and long-duration bursts, like downloads. So a more appropriate congestion pricing scheme would be to combine traffic smoothing with rate-limiting. One approach for combining them is to set a lower QoS with more smoothing and lower rate-limit level than a higher QoS. Using the same simulation setup as in the first experiment, we found that using this approach can effectively reduce access link bursts. In simulations, we let the three levels of QoS, SLOW-GOING, MODERATE, and RESPONSIVE, have the same traffic smoothing as before, but capped the SLOW-GOING at 1M, the MODERATE at 10M, and the RESPONSIVE at 100M (see Table 7.11). We found that the RESPONSIVE can reduce the access link bursts by 7.7% and the MODERATE by 40.7% (see Table 7.12). Thus

when there is no congestion, the price of the RESPONSIVE can be set so low that most of the users can afford it. However, when there is congestion, its price can be increased to encourage users to use the MODERATE, thereby, reducing the access link bursts further by possibly another 33% (7.7% to 40.7%).

Table 7.11: Time at each level when load increases- when combining traffic smoothing with rate-limiting.

Level (Mbps)	0.049	0.07	0.1	0.14	0.2	0.4	0.8	1.6	3.2	6.4	12.8	25.6	51.2	102.4
SLOW-GOING (Sec)	6	6	6	6	6	9	9	9						
MODERATE (Sec)	0	3	3	3	3	6	6	6	6	6	6			
RESPONSIVE (Sec)	0	0	3	0	3	3	3	3	3	3	3	3	3	3

Table 7.12: Effectiveness of different QoSs on reducing access link bursts when combining traffic smoothing with rate-limiting.

Quality	Reduction in Standard Deviation of Bursts at Access Link
SLOW-GOING	68.0%
MODERATE	40.7%
RESPONSIVE	7.7%

7.3 Conclusion

Using two experiments of prototyping, evaluation, and analysis, we found a congestion pricing scheme that is acceptable to users and effective for operators in allocating LAN access link bandwidth. In the first experiment, we offered 12 users three sizes of bandwidth and gave each user certain number of tokens a day. We then adjusted the prices of the three sizes according to load and charged by the minute. We found that this scheme did not work well because user usages were dominated by short bursts. Rate-limiting was not effective given the pattern of short bursts. Furthermore, charging by the

minute was very taxing on user involvement. From analyzing usage pattern, we found that charging users once every 10-15 minutes would be more reasonable and that using traffic smoothing would be very effective for dealing with short bursts. More specifically, if half of the users of a large network can be enticed to have their traffic smoothed, then the burstiness at its access link can be reduced by 20-30%. In the second experiment, we offered 8 users three levels of QoS that differ on degree of traffic smoothing and used a charging granularity of 15 minutes. We found that this scheme is effective because we can easily entice users to select a lower QoS, one with more smoothing, by increasing the price of a higher QoS. It is also acceptable because users only need to make a purchasing decision at most once every 15 minutes. Finally, for usages containing both short-duration bursts (web surfing) and long-duration bursts (downloads), we found through simulations that combining traffic smoothing with rate-limiting can be very effective for congestion pricing[32].

[32] It does not make sense to try this scheme with our user group because their usages contain very few long downloads. Thus we need a group whose usages contain both short-duration bursts and long-duration bursts.

Chapter 8 Conclusion

Congestion pricing, that is, the varying of prices according to load, can efficiently allocate bursty usage of scarce network resources. During peak usages, it can allocate resources according to user valuation by involving users in allocation decisions. However, there is a lack of detailed evaluations with dynamic pricing on user resource demands. Therefore, we conducted user experiments with real systems to understand how to apply dynamic pricing for voice and data traffic. We discovered that there is no obvious congestion pricing scheme that is both acceptable to users and effective for operators. Many user interface and system issues must be considered. However, we also confirmed that a good scheme can potentially be very effective for solving congestion. Our work should be viewed as an initial user study of applying congestion pricing to allocate network resources like voice and data bandwidth.

In Section 8.1, we review the motivations and the challenges of dynamic pricing. In Section 8.2, we summarize our work on voice and data traffic. In Section 8.3, we generalize our findings on applying dynamic pricing to network resources. In Section 8.4, we provide a critique of our work. In Section 8.5, we suggest future research. Finally, in Section 8.6, we conclude with a list of our contributions.

8.1 Motivations and Challenges

Congestion pricing is a resource allocation mechanism that varies prices to affect user demand. It can help operators achieve economic efficiency by allocating resources according to user valuation. It is especially useful for resources whose average utilization

is low, but peak usage is high and unpredictable. It can also benefit users by providing them with an option to obtain good service quality during periods of congestion. However, the main drawback is that it requires user involvement, and too much involvement can become annoying and lead to the loss of the ability to influence user behavior. Simulations studies on congestion pricing reach different conclusions because they strongly depend on the workload models and the user models used. Thus user evaluations with real systems are needed to prove its efficacy. In our user evaluations, we focus on applying dynamic pricing at access points for voice and data traffic.

The goal of congestion pricing research is to determine whether dynamic pricing can be acceptable to users and effective for operators. For voice traffic, user acceptance is easier because they are familiar with paying by the minute when making calls. However, we still need to verify that changing prices during a call can affect user behavior and not cause excessive user grievance. Understanding just how effective congestion pricing can be for voice traffic is more challenging. Operators are really concerned about the tradeoffs when involving thousands of users. However, large-scale user studies are difficult to arrange and simulation studies are strongly dependent on user models. Thus we need a methodology that can make believable the results scaled up from a small-scale user study. For data traffic, user acceptance of dynamic pricing is more difficult because users have not needed to deal directly with prices before. Furthermore, there are many variations of congestion pricing to explore. For example, one can allocate resources based on rate-limiting, quality-of-service, etc. After finding a scheme that users can understand and accept, we still need to ensure that they would actually respond to changing prices. Afterwards, having some users respond to price changes would need to actually be

effective in reducing overall congestion. Thus for congestion pricing, one needs to view the problem from both users' and operators' perspective, keeping in mind both user interface and system issues like acceptance and performance, and consider scaling issues.

8.2 Summary of Work

To evaluate congestion pricing when there are many users, we propose a methodology that combines small-scale user studies with large-scale simulations. First, user studies are conducted to understand user acceptance and user response to price changes. Next, we use these results to model user behavior for performing large-scale simulations. Such simulations allow us to propose rules for managing congestion pricing and estimate tradeoffs between system performance and user satisfaction. The tradeoffs strongly depend on the form and the parameter of the user model. Thus, we verify them by exploiting the user model and the rules for managing congestion pricing to emulate a large-scale service, and re-measuring user reactions to price changes under such setting.

We applied the above methodology to evaluate the effectiveness of congestion pricing for voice traffic using a voice-over-IP gateway service. To attract users to the service, we realized a full-featured service was necessary. We used a four-state FSM to quickly prototype enhanced computer-telephony features, such as incoming call redirection and device handoff. After attracting 100 users, we used the service to conduct experiments for over one year to observe the effects of various pricing policies. During the experiments, each user was given a certain number of free tokens a week and charged a certain token rate a minute. We found that if prices change neither quickly nor frequently, then users can easily be enticed to shorten their sessions after a price increase. Using surveys, we found that users would accept dynamic pricing if given a small

discount. To understand large-scale issues, we used a simple user model to simulate the effects of congestion pricing for voice calls when there are many users using a voice-over-IP gateway service. Using simulations, we proposed a set of rules for operators to most effectively set parameters for managing congestion pricing at scale. By setting the parameters to appropriate values, we estimated that congestion pricing can reduce call blocking rate by 50% or save on provisioning costs by 20%, while only causing users to experience a price change in 4% of their usages. Finally, we verified the user model by re-measuring user reaction to price changes under an emulated large-scale service.

We believe that dynamic pricing can be effective for allocating bandwidth for voice traffic. When applying congestion pricing, we recommend that prices should change neither quickly nor frequently so that users know the cost of extending their calls. From our observations of calling patterns, we see that there are only a few congested periods in a given day, albeit difficult to predict. Nevertheless, resources need to be provisioned for the peak usages. Thus applying congestion pricing during the congested periods can dramatically reduce provisioning while only requiring users to experience price changes occasionally.

For data traffic, we used congestion pricing to allocate bandwidth of a LAN access link. We performed two iterations of prototyping, evaluation, and analysis, and found an acceptable and effective scheme. In the first iteration, we offered 12 users three sizes of bandwidth and gave each user a certain number of tokens a day. The prices of the three sizes are varied according to load, and users are charged by the minute. This scheme did not work well because it was difficult for users to request different sizes to adjust their usages of mostly short duration bursts. In the second iteration, we

experimented with offering users different classes of service, based on traffic smoothing, and charged users only once every 15 minutes. Through experimentation, we found that the scheme can be used to easily entice users to select a lower service class by increasing the price of a higher class. It is acceptable to users because they only need to make a purchasing decision periodically. Furthermore, based on our simulations, if half of the users of a large network can be enticed to have their traffic smoothed during congestion, then the burstiness at its access link can be reduced by 20-30%.

With demand for data traffic growing rapidly, we believe that congestion pricing should be considered as a method for resource allocation because it can more efficiently allocate scarce bandwidth. When applying congestion pricing at access points, we recommend offering users three classes of service based on traffic smoothing and rate-limiting. The quality for each class should be predictable so that users know the services they are purchasing. To minimize user concerns, there should be a cap for the price and users should only have to make a purchasing decision at most once every 10-15 minutes.

8.3 Generalization

Finding an acceptable and effective scheme strongly depends on the resource involved, the cause of congestion, and the desired user responses. For voice traffic, the resource is session-oriented and a typical session lasts on the order of minutes. The first few minutes of a session are important to users, but the later minutes are more optional. Furthermore, for voice traffic, only one service class is required and each session utilizes the same amount of resources. Congestion is difficult to predict: it can occasionally occur unexpectedly, though utilization is low most of the time. For phone calls, users want to call when they need to and call blocking is undesirable. Thus to avoid blocking or to

reduce provisioning, one can use prices to encourage users to use less resources during periods of congestion. Congestion pricing can be applied to network resources, like dialup modems, that have the above characteristics.

For data traffic, the resource is shared by many people. Each user's usage is bursty in time and volume. Users have multiple service requirements. Some users can tolerate more delay or loss than others. At the same time, some users would prefer predictable or higher service quality. Congestion can occur all of a sudden, however, with feedbacks, users can quickly adjust their usages by a large amount. Furthermore, users can tolerate occasional involvement. Thus, we believe that our results at a LAN can also be applied to other scarce bottlenecks like wireless bandwidth.

8.4 Critiques

One criticism of our user studies is that our subjects are mostly students. For voice traffic, we experimented with about 100 dormitory students. Yet, they are from various majors, and certainly not all of them are computer experts. Based on our experience, surveys, and focus groups, we found that occasional price changes during phone calls are acceptable. For data traffic, we only evaluated our system using 10 users in our research group. However, we conducted our study when users have committed to use dynamic pricing to access bandwidth for their everyday work. Under such a situation, we found that users can tolerate dynamic pricing. Thus even though our subjects are not representative of the general public, our studies indicate that congestion pricing can be designed to be acceptable to users.

Another issue with our setup is that we did not charge users with real money. For congestion control purpose, we believe that using free but limited tokens is an effective

constraint. In the real world, one would need to allocate users different amount of tokens. Thus some real money might need to be exchanged for tokens. However, if real money is used, we believe that dynamic pricing would be even more effective because users would care more when prices change.

8.5 Next Steps

To our knowledge, our work is the most extensive evaluation of user reaction to congestion pricing. Nevertheless, more evaluations with more and various user groups are needed. There are still many design decisions to explore for both voice and data traffic. Examples include different user interface designs, different incentive schemes, etc. Regarding pricing policies, it would be interesting to offer users different policies, e.g., flat-rate, per-session dynamic pricing, and per-minute dynamic pricing, and observe which policies they choose. Thus the next step would be to work with a voice operator or an ISP (or even the network administrator of a large academic or corporate campus) to experiment dynamic pricing with more users. With a larger group, one can conduct control experiments to compare dynamic pricing with a static policy like time-of-day pricing. For future user studies, one can utilize the methodology presented in this thesis to verify the estimated tradeoffs of congestion pricing for different user groups or other scarce resources. After investigating congestion pricing at access points, research should be expanded to applying congestion pricing across multiple bottlenecks.

8.6 Contributions

To encapsulate, the main contributions of our congestion pricing investigation are:

- A methodology for scaling the results from a small-scale user study.
- Real implementations and deployments of systems using dynamic pricing.
- Effective schemes of applying dynamic pricing to users for voice and data traffic.
- Measurements of user response and acceptance to dynamic pricing.
- User models based on user experimentations.
- Simulations of the benefits and drawbacks of congestion pricing under large scales.

Bibliography

[1] Altmann, J.; Daanen, H.; Oliver, H.; Suárez, A.S. **How to market-manage a QoS network**. Proceedings IEEE INFOCOM 2002. Conference on Computer Communications, June 2002.

[2] Altmann, J.; Varaiya, P. **INDEX project: user support for buying QoS with regard to user's preferences**. 1998 Sixth International Workshop on Quality of Service, 1998. p.101-4.

[3] Baeza-Yates, R.; Piquer, J.M.; Poblete, P.V. **The Chilean Internet connection, or I never promised you a rose garden**. Proceedings of INET'93, 1993.

[4] Blake, S.; Black, D.; Carlson, M.; Davies, E.; Wang, Z.; Weiss, W. **An architecture for differentiated services.** Internet Engineering Task Force, Request for Comments 2475, Dec. 1998.

[5] Brownlee, N. **New Zealand experiences with network traffic charging**. Connexions, vol.8, (no.12), Dec. 1994. p.12-19.

[6] Caesar, M.C.; Balaraman, S.; Ghosal, D. **A comparative study of pricing strategies for IP telephony**. IEEE Global Telecommunications Conference, vol.1, 2000. p.344-9.

[7] Chang, X.; Petr, D.W. **A survey of pricing for integrated service networks**. Computer Communications, vol.24, (no.18), Dec. 2001. p.1808-18.

[8] Cocchi R.; Estrin, D.; Shenker, S.; Zhang, L. **A study of priority pricing in multiple service class networks**. Computer Communication Review, vol.21, (no.4), Sept. 1991. p.123-30.

[9] Danielsen, K.; Weiss, M. **User control and IP allocation**. Internet Economics, ed. Lee McKnight and Joseph Bailey, Cambridge, Mass.: MIT Press, 1995.

[10] Edell, R.; Varaiya, P. **Demand for quality-differentiated network services**. Proceedings of the 36th IEEE Conference on Decision and Control, vol.3, 1997. p.2922-7.

[11] Edell, R.; Varaiya, P. **Providing Internet access: what we learn from INDEX**. IEEE Network, vol.13, (no.5), Sept.-Oct. 1999. p.18-25.

[12] Edell, R.J.; McKeown, N.; Varaiya, P.P. **Billing users and pricing for TCP**. IEEE Journal on Selected Areas in Communications, vol.13, (no.7), Sept. 1995. p.1162-75.

[13] Estrin, D.; Zhang, L. **Design considerations for usage accounting and feedback in internetworks**. Computer Communication Review, vol.20, (no.5), Oct. 1990. p.56-66.

[14] Fankhauser, G.; Stiller, B.; Blattner, B. **Arrow: a flexible architecture for an accounting and charging infrastructure in the next-generation Internet**. NETNOMICS: Economic Research & Electronic Networking, vol.1, (no.2), 1999. p.201-23.

[15] Fishburn, P.C.; Odlyzko, A.M. **Dynamic behavior of differential pricing and quality of service options for the Internet**. Decision Support Systems, vol.28, (no.1-2), March 2000. p.123-36.

[16] Fitkov-Norris, E.D.; Khanifar, A. **Dynamic pricing in cellular networks, a mobility model with a provider-oriented approach**. Second International Conference on 3G Mobile Communication Technologies, 2001. p.63-7.

[17] Fulp, E.W.; Ott, M.; Reininger, D.; Reeves, D.S. **Paying for QoS: an optimal distributed algorithm for pricing network resources**. 1998 Sixth International Workshop on Quality of Service, 1998. p.75-84.

[18] Gupta, A.; Stahl, D.O.; Whinston, A.B. **Priority pricing of integrated services networks**. Internet Economics, ed. Lee McKnight and Joseph Bailey, Cambridge, Mass.: MIT Press, 1995.

[19] Handley, M.; Schulzrinne, H.; Schooler, E.; Rosenberg, J. **SIP: session initiation protocol**. Internet Engineering Task Force, Request for Comments 2543, March, 1999.

[20] Henderson, T.; Crowcroft, J.; Bhatti, S. **Congestion pricing paying your way in communication networks**. IEEE Internet Computing, vol.5, (no.5), Sept.-Oct. 2001. p.85-9.

[21] ITU-T Recommendation H.323. **Packet based multimedia communication system**. 1988.

[22] Klausz, F.J.; Croson, D.C.; Croson, R.T.A. **An experimental auction to allocate congested IT resources: the case of the University of Pennsylvania modem pool**. Proceedings of the Thirty-First Hawaii International Conference on System Sciences, vol.6, 1998. p.363-73.

[23] Leland, W.E.; Willinger, W.; Taqqu, M.S.; Wilson, D.V. **On the self-similar nature of Ethernet traffic**. Computer Communication Review, vol.23, (no.4), Oct. 1993. p.183-93.

[24] MacKie-Mason, J.K.; Murphy, L.; Murphy, J. **The role of responsive pricing on the Internet**. Internet Economics, ed. Lee McKnight and Joseph Bailey, Cambridge, Mass.: MIT Press, 1995.

[25] MacKie-Mason, J.K.; Varian, H.R. **Pricing the Internet**. Public access to the Internet, ed. Brian Kahin and James Keller, Englewood Cliffs, N.J.: Prentice-Hall, 1995.

[26] MacKie-Mason, J.K.; Varian, H.R. **Some economics of the Internet**. Networks, infrastructure and the new task for regulation, ed. Werner Sichel, Ann Arbor, Mich.: University of Michigan Press, 1995.

[27] MacKie-Mason, J.K.; Varian, H.R. **Some FAQs about usage-based pricing**. Computer Networks & ISDN Systems, vol.28, (no.1-2), Dec. 1995. p.257-65.

[28] Murphy, L.; Murphy, J.; MacKie-Mason, J.K. **Feedback and efficiency in ATM networks**. 1995 IEEE International Conference on Communications. Converging Technologies for Tomorrow's Applications, vol.2, 1996. p.1045-9.

[29] Neugebauer, R.; McAuley, D. **Congestion prices as feedback signals: an approach to QoS management**. Proceedings of the 9th ACM SIGOPS European Workshop, Sept. 2000. p.91-96.

[30] Parris, C.; Keshav, S.; Ferrari, D. **A framework for the study of pricing in integrated networks**. Tech. Rept. TR-92-016, International Computer Science Institute, Berkeley, California, 1992.

[31] Paschalidis, I.Ch.; Tsitsiklis, J.N. **Congestion-dependent pricing of network services**. IEEE/ACM Transactions on Networking, vol.8, (no.2), April 2000. p.171-84.

[32] Patek, S.D.; Campos-Nanez, E. **Pricing of dialup services: an example of congestion-dependent pricing in the Internet**. Proceedings of the 39th IEEE Conference on Decision and Control, vol.3, 2000. p.2296-301.

[33] Peha, J.M. **Dynamic pricing as congestion control in ATM networks**. IEEE Global Telecommunications Conference, vol.3, 1997. p.1367-72.

[34] Pindyck, R.S.; Rubinfeld, D.L. **Econometric models and economic forecasts**. 3rd ed. New York: McGraw-Hill, 1991.

[35] Rupp, B.; Edell, R.; Chand, H.; Varaiya, P. **INDEX: a platform for determining how people value the quality of their Internet access**. 1998 Sixth International Workshop on Quality of Service, 1998. p.85-90.

[36] Semret, N.; Liao, R.R.-F.; Campbell, A.T.; Lazar, A.A. **Peering and provisioning of differentiated Internet services**. Proceedings IEEE INFOCOM 2000. Conference on Computer Communications, vol.2, 2000. p.414-20.

[37] Shenker, S. **Service models and pricing policies for an integrated services Internet**. Public access to the Internet, ed. Brian Kahin and James Keller, Englewood Cliffs, N.J.: Prentice-Hall, 1995.

[38] Shenker, S.; Clark, D.; Estrin, D.; Herzog, S. **Pricing in computer networks: reshaping the research agenda**. Computer Communication Review, vol.26, (no.2), April 1996. p.19-43.

[39] Stiller, B.; Fankhauser, G.; Joller, G.; Reichl, P.; Weiler, N. **Open charging and QoS interfaces for IP telephony**. The Internet Summit, San Jose, California, June 1999.

[40] Stiller, B.; Reichl, P.; Leinen, S. **Pricing and cost recovery for Internet services: practical review, classification, and application of relevant models**. NETNOMICS: Economic Research & Electronic Networking, vol.3, (no.2), 2001. p.149-71.

[41] Waldspurger, C.A.; Hogg, T.; Huberman, B.A.; Kephart, J.O.; Stornetta, W.S. **Spawn: a distributed computational economy**. IEEE Transactions on Software Engineering, vol.18, (no.2), Feb. 1992. p.103-17.

[42] Wang, X.; Schulzrinne, H. **An integrated resource negotiation, pricing, and QoS adaptation framework for multimedia applications**. IEEE Journal on Selected Areas in Communications, vol.18, (no.12), Dec. 2000. p.2514-29.

[43] Xiaowei, C.; Mingquan, L.; Zhenming, F. **A model based on congestion pricing for QoS**. Proceedings 2001 International Conference on Computer Networks and Mobile Computing, 2001. p.243-7.

[44] Zhang, L.; Deering, S.; Estrin, D.; Shenker, S.; Zappala, D. **RSVP: a new resource reservation protocol**. IEEE Network, vol.7, (no.5), Sept. 1993. p.8-18.

Appendix A: Survey Questions and Answers for Voice Experiments

Survey 1

Date: 10/10/00

Questions:

1. Have you used the service from a computer and a phone to make outgoing calls? If not, can you tell us why?

2. Would you continue using the service to make outgoing calls? If not, can you tell us why?

3. Any suggestion for improving the service?

4. We would like to sign up more students at Foothill and Stern. Any suggestion on how we should advertise our service? E.g., would having an informational table next to the cafeteria in one of the evenings help?

Sample Answers to Question 1:

“I have used the service and is pretty satisfied with it.”
“I like hand-held phones better.”
“All of my soundcard's output plugs are used up by speakers since I have surround sound so it would be a hassle to unplug my speakers and plug in the microphone.”
“I haven't gotten the chance to go to CompUSA to buy the sound card yet.”

Sample Answers to Question 2:

“The service is great mainly because it is free.”
“The voice quality is actually very good, much better than other Internet-based calling services.”
“Because it has been working well.”
“It's been pretty convenient.”
“If the service wasn't free, I doubt I'd use it.”
“I will continue using the service because I found the reception to be very clear. It sounded like a cell phone.”

Sample Answers to Questions 3:

“The microphone volume threshold of the incoming sound is too low.”
“The main problem with the service, and with every other voice-over-IP service, is the half-second lag time between transmissions.”

Survey 2

Date: 11/03/00

Question:

Currently only half of the users who signed up are using the service. I would like to find out why. If you are not using the service, can you send me an email telling me why?

Sample Answers:

“Because I don't have that many non-local (i.e. already free) calls to make in the Bay Area.”
“I haven't used the service within the last two weeks because I haven't had a reason to call long distance.”

Survey 3

Date: 12/8/00

Question:

According to our record, you have used the computer-telephony service in the past two weeks when congestion pricing was used. Congestion pricing charges users 10*X tokens a minute, where X is the number of people using the service. According to our record, 93% of the minutes got charged 10 tokens a minute and 7% of the minutes got charged 20 tokens a minute. I am wondering if you can send me an email to the following question.

1. For a similar congestion level, do you prefer a flat rate of 15 tokens a minute or congestion pricing of 10*X tokens a minute, where X is the number of people using the service.

Sample Answers:

“I would rather have congestion pricing if that annoying voice wouldn't come on every time it changes. I wish it just wouldn't tell me. Otherwise, I'd rather have flat-rate without the voice.”
“But the only annoying thing about it is that it would stop my conversation just to tell me what the rate is.”
“I prefer congestion pricing. However, during holiday times (Xmas), a flat rate pricing scheme might be better.”
“Because the chance of running into more than 1 people using the system isn't that high.”
“Seems it would be more likely to give a lower rate.”

Survey 4

Date: 12/18/00

Questions

1. We would like to have more dormitory users sign up for the service next Spring semester. Which methods below would you recommend for advertising our service?
    a. put flyers in students' mail boxes.
    b. put posters around the dorms.
    c. hold small info sessions at the dorms.
    d. hold a big info session at Soda Hall (computer science building).
    e. Others...

2. We would like to get the existing users to use the service more? Which methods below would make you use the service more?
    a. increase call coverage to allow calls to anywhere in California.
    b. add extra features like voice mail.
    c. Others...

3. How can we get users to use their computers to make phone calls. We found that we really couldn't use prices to encourage users to use their computers instead of phones. If you strongly prefer to use your phone instead of your computer, can you tell us why?
    a. you just prefer to use a phone instead of a computer.
    b. you don't have your computer setup to make phone calls.
    a. you are using other Internet phone services, like Dialpad and Net2Phone, on your computer.
    b. Others...

4. Finally, any other suggestion or comment about the service?

Sample Answers to Question 1:

“Extend the service to a larger area.”
“Give current users an incentive to get other users to join.”
“A mailing list of students would also work.”

Sample Answers to Question 2:

“More call minutes.”
“Allow calls out of state as well.”
“To the rest of the US would be outstanding.”
“Make rates cheaper.”

“Internet calls are still not a good alternative to normal phone calls because of the lag time.”

Sample Answers to Question 3:

“You can walk around and do stuff with the phone.”
“I strongly prefer using the phone because all of the output slots on my sound card are already taken up, it would be a hassle to unplug some of them to plug in the microphone headset each time I need to make a call.”
“Using the computer doesn't work for some reason, even though it's set up and everything.”
“It was a bit too complicated to set it up through NetMeeting.”
“It's difficult to do, and too much of a pain - the phone is much easier.”

Sample Answers to Question 4:

“I have to dial a lot of numbers for one phone call. Anyway to reduce the amount of numbers to be punched in?”
“I learned that when using the service, the quality of the connection is less than normal. It sounds like talking on a cell phone, static-ky.”

Survey 5

Date: 3/6/01

Demographic Questions:

1. What is your gender? (Male (M) or Female (F))

2. What is your major & year? (E.g., EECS/junior)

Communication Pattern Questions:

1. Do you have access to a computer (laptop or desktop) in your room? (Yes (Y) or No (N))
    a. How many emails do you send & receive? (E.g., 10 a day, 100 a week)
    b. How many instant messages do you send & receive? (E.g., 10 a day, 100 a week)
    c. How often do you use other Internet-telephony services, like Dialpad and Net2Phone? (E.g., 10 a day, 100 a week)

2. Do you have access to a cell-phone? (Yes (Y) or No (N))
   If Yes,
    a. What is your average monthly phone bill? (E.g., $25 a month)
    b. What percentage of your calls is personal (family & friends) versus business (banking, etc)? (E.g., 60/40: 60% personal & 40% business)

3. For the phone in your room?
    a. What is your average monthly phone bill? (E.g., $40 a month)
    b. What percentage of your calls is personal (family & friends) versus business (banking, etc)? (E.g., 60/40: 60% personal & 40% business)

ICEBERG Computer Telephony Service Questions:

1. Have you used the service from a phone? (Yes (Y) or No (N))
    a. If Yes, on a scale of 1-5, how do you like the service? (5:very satisfied, 4:satisfied, 3:OK, 2:dissatisfied, 1:very dissatisfied) And why? (...)
    b. If No, why not? (...) And would you use it soon? (Yes (Y) or No (N))

2. Have you used the service from a computer? (Yes (Y) or No (N))
    a. If Yes, on a scale of 1-5, how do you like the service? (5:very satisfied, 4:satisfied, 3:OK, 2:dissatisfied, 1:very dissatisfied) And why? (...)
    b. If No, why not? (...) And would you use it soon? (Yes (Y) or No (N))

Pricing Policies Questions:

1. On a scale of 1-5, how do you like time-of-day pricing that charges less during the off-peak hours and more during the peak hours? (5:like it a lot, 4:like it, 3:OK, 2:dislike it, 1:dislike it a lot) And why? (...)
    a. Compare with a time-of-day pricing of 10 tokens/min from 11pm-7pm and 30 tokens/min from 7pm-11pm, do you prefer:
        i. Flat rate (F) of 15 tokens/min or the time-of-day (T)? (F or T)
        ii. Flat rate (F) of 20 tokens/min or the time-of-day (T)? (F or T)
        iii. Flat rate (F) of 25 tokens/min or the time-of-day (T)? (F or T)

2. On a scale of 1-5, how do you like call-duration based pricing that charges less for a short duration call and more for a long duration call? (5:like it a lot, 4:like it, 3:OK, 2:dislike it, 1:dislike it a lot) And why? (...)
    a. Compare with a call-duration pricing of 5 tokens/min from 1st to 3rd minute, 10 tokens/min from 4th to 10th minute, 20 tokens/min from 11th to 20th minutes, & 30 tokens/min from 21st minute on, do you prefer:
        i. Flat rate (F) of 10 tokens/min or the call-duration (C)? (F or C)
        ii. Flat rate (F) of 15 tokens/min or the call-duration (C)? (F or C)
        iii. Flat rate (F) of 20 tokens/min or the call-duration (C)? (F or C)
        iv. Flat rate (F) of 25 tokens/min or the call-duration (C)? (F or C)

3. On a scale from 1-5, how intrusive is the price announcement in the middle of a call? (5:very intrusive, 4:intrusive, 3:tolerable, 2:slightly intrusive, 1:not intrusive)
    a. If a call costs 10 tokens/min, how much discount would it take so that you would not mind receiving a price announcement once in a while (at most once a minute)? (E.g., never, discount of 1 token/min, discount of 2tokens/min)

4. Any other suggestion or comment?

Survey 6

Date: 5/4/01

Demographic Questions:

1. What is your gender? (Male (M) or Female (F))

2. What is your major & year? (E.g., EECS/junior)

Congestion Pricing Questions:

1. On a scale of 1-5, how do you like congestion pricing that charges more when more people are using a service and less when less people are using it? (5:like it a lot, 4:like it, 3:OK, 2:dislike it, 1:dislike it a lot) And why?

2. What if congestion pricing can reduce the chance that your call might be blocked because all the phone lines are busy? (5:like it a lot, 4:like it, 3:OK, 2:dislike it, 1:dislike it a lot)

3. What if congestion pricing can make it cheaper for you to use the service because the service provider can support more users with the same number of phone lines? (5:like it a lot, 4:like it, 3:OK, 2:dislike it, 1:dislike it a lot)

4. If the average rate under congestion pricing is 12.5 tokens/min (80% of the time:10 tokens/min, 15% of the time:20 tokens/min, and 5% of the time:30 tokens/min), do you prefer:
    a. Congestion pricing (C) with an average rate of 12.5 tokens/min or a flat-rate (F) of 12.5 tokens/min? (C or F)
    b. Congestion pricing (C) with an average rate of 12.5 tokens/min or a flat-rate (F) of 15 tokens/min? (C or F)
    c. Congestion pricing (C) with an average rate of 12.5 tokens/min or a flat-rate (F) of 20 tokens/min? (C or F)
    d. Congestion pricing (C) with an average rate of 12.5 tokens/min or a flat-rate (F) of 25 tokens/min? (C or F)

5. If the prices under congestion pricing can change (increase or decrease) from one minute to the next, would a price INCREASE affect your behavior? Why or why not?

6. If the prices under congestion pricing can change (increase or decrease) from one minute to the next, would a price DECREASE affect your behavior? Why or why not?

7. If the prices under congestion pricing will ONLY INCREASE during a call, would a price increase affect your behavior? Why or why not?

8. If each price change (increase or decrease) under congestion pricing will last at least three minutes, would a price INCREASE affect your behavior? Why or why not?

9. If each price change (increase or decreases) under congestion pricing will last at least three minutes, would a price DECREASE affect your behavior? Why or why not?

Quality-Based Pricing Questions:

1. On a scale of 1-5, how do you like quality-based pricing where in the beginning of a call, you can choose between a high-quality-connection at a higher price or a low-quality-connection at a lower price? (5:like it a lot, 4:like it, 3:OK, 2:dislike it, 1:dislike it a lot) And why?

2. For those who have used Internet-telephony services like Dialpad and Net2Phone, if the high-quality-connection is the telephone quality and the low-quality-connection in the Internet-telephony quality, do you prefer:
    a. The high-quality-connection (H) at 30 tokens/min or the low-quality-connection (L) at 25 tokens/min? (H or L)
    b. The high quality-connection (H) at 30 tokens/min or the low-quality-connection (L) at 20 tokens/min? (H or L)
    c. The high-quality-connection (H) at 30 tokens/min or the low-quality-connection (L) at 15 tokens/min? (H or L)
    d. The high-quality-connection (H) at 30 tokens/min or the low-quality-connection (L) at 10 tokens/min? (H or L)

1. Any other suggestion or comment about the service or the pricing experiments?

Survey 7

Date: 10/22/01

Demographic Questions:

1. What is your gender? (Male (M) or Female (F))

2. What is your major & year? (E.g., EECS/junior)

Stated Preference Questions:

1. On a scale of 1-5, how do you like congestion pricing that charges more when more people are using a service and less when less people are using it? (5:like it a lot, 4:like it, 3:OK, 2:dislike it, 1:dislike it a lot) And why?

2. What if congestion pricing can reduce the chance that your call might be blocked because all the phone lines are busy? (5:like it a lot, 4:like it, 3:OK, 2:dislike it, 1:dislike it a lot)

3. What if congestion pricing can make it cheaper for you to use the service because the service provider can support more users with the same number of phone lines? (5:like it a lot, 4:like it, 3:OK, 2:dislike it, 1:dislike it a lot)

4. What if there is a less disruptive way of indicating the current price? For example, 1 beep for 10 tokens, 2 beeps for 20 tokens, etc. (5:like it a lot, 4:like it, 3:OK, 2:dislike it, 1:dislike it a lot)

5. If the average rate under congestion pricing is 12.5 tokens/min (80% of the time:10 tokens/min, 15% of the time:20 tokens/min, and 5% of the time:30 tokens/min), do you prefer:
    a. Congestion pricing (C) with an average rate of 12.5 tokens/min or a flat-rate (F) of 12.5 tokens/min? (C or F)
    b. Congestion pricing (C) with an average rate of 12.5 tokens/min or a flat-rate (F) of 15 tokens/min? (C or F)
    c. Congestion pricing (C) with an average rate of 12.5 tokens/min or a flat-rate (F) of 20 tokens/min? (C or F)
    d. Congestion pricing (C) with an average rate of 12.5 tokens/min or a flat-rate (F) of 25 tokens/min? (C or F)

6. If the prices under congestion pricing can change (increase or decrease) from one minute to the next, would a price INCREASE affect your behavior? Why or why not?

7. If the prices under congestion pricing can change (increase or decrease) from one minute to the next, would a price DECREASE affect your behavior? Why or why not?

8. If each price change (increase or decrease) under congestion pricing will last at least three minutes, would a price INCREASE affect your behavior? Why or why not?

9. If each price change (increase or decreases) under congestion pricing will last at least three minutes, would a price DECREASE affect your behavior? Why or why not?

Any other suggestion or comment about the service or the pricing experiments?

Appendix B: Survey Questions and Answers for Data Experiments

Survey 1

Date: 3/5/02

Bandwidth Rate-Limiting Questions:

Please use the following scale to answer the following questions.
(1:dislike it a lot, 2:dislike it, 3:OK, 4:like it, 5:like it a lot)

1. From a scale of 1-5, how do you like the existing best-effort service (no bandwidth rate-limiting, no prices, but no bandwidth guarantees)?

2. From a scale of 1-5, how do you like a service where you need to go through a user interface to request additional bandwidth but can obtain guaranteed bandwidth? Why this rating?

3. Currently, there are three choices of bandwidth selection. Is three OK or should there be more or less choices?

4. Currently, the three bandwidth choices are 150K, 5M, and 10M. Is the middle choice OK or should it be higher or lower?

Congestion Pricing Questions:

5. From a scale of 1-5, how do you like per-session congestion pricing where prices for the different amount of bandwidth can change each session? Why this rating?

6. From a scale of 1-5, how do you like per-minute congestion pricing where the price charged for bandwidth can vary each minute? Why this rating?

7. Did having limited number of tokens affect your bandwidth purchase decisions?

8. Did having limited number of tokens affect the durations of your purchases?

9. Is charging by the minute OK or do you prefer a smaller or a longer charging interval? And why?

10. During the week that per-minute congestion pricing is used, do you feel that prices were changing too frequently?

11. From a scale of 1-5, how do you like the following changes to congestion pricing?
    a. No charge and no rate-limiting when no congestion.
    b. Users can specify simple rules for making purchases. E.g., if reaching a limit, have more than X tokens, and price is less than Y, then upgrade.
    c. Users can specify the timeout period for the idle timer.

12. Comments or suggestions on congestion pricing, user interface, experimental setup, etc?

Sample Answers to Question 2:

"4, prefer to make it explicit that I want better than best-effort service."
"4, if it gives me *additional* bandwidth, that's naturally an advantage."
"1, just gets in the way."
"2, most important, I cannot get used to it. Every time, I feel about congestion, then I recalled that I need to go and buy more bandwidth. That adds extra efforts. Further, by the time that I purchased the bandwidth, I probably have been limited by 1 min or so already. And the congestion probably will be almost gone by the time I got the b/w."
"I hate to have to interact with the interface to request bandwidth."
"3, for a few reasons:
- It takes time for me to request bandwidth, which forces short transactions such as if I need to check a web page take much longer.
- It's inconvenient to have to remember to request bandwidth.
- It's difficult or impossible to request bandwidth for certain uses. For example, if I want to use my machine as a web server when I'm not at the machine the user must suffer low bandwidth.
However: I see your work as developing two things (1) a congestion pricing mechanism and (2) an agent that can send requests on behalf of the user. A more powerful agent that could adapt policy to user specifications could fix all of these problems."

Sample Answers to Question 3:

"The choices are good enough."
"Three is reasonable, more would be confusing."
"I can't distinguish between 5 and 10 meg for my usage, and sometimes choose randomly."
"2 choices--cheap and expensive!"

Sample Answers to Question 4:

"I think they are pretty good, actually."

Sample Answers to Question 5:

"2, wasted money when not using or when thinking."

Sample Answers to Question 6:

"4, easy to use bandwidth on demand."

Sample Answers to Question 7:

"YES!"
"Yes, I attempted to conserve (although I never came close to running out)."

Sample Answers to Question 8:

"Yes, to a certain extent."

Sample Answers to Question 9:

"The fewer interactions with the system the better."
"Per-minute is OK, reasonable balance between frequent UI interactions and cost."
"I would prefer longer interval. As I said, the bandwidth interval for me is more than 1-5 mins. That is, I have to go and buy bandwidth almost every time."
"Charging by the minute is perfect for me, as most of my requests are short and I don't want to be charged for bandwidth I don't use."
"2-3 minutes would be better. I usually want to buy bandwidth before say, a print job, which requires bandwidth from my computer. Often times, 1 minute is too small for me to switch windows, look at the PS file, and then click print."

Sample Answers to Question 11:

"4, and it will be REALLY cool if we can automate it."
"I can adjust it to match w/ my traffic patterns."
"Definitely an improvement, but might be inconvenient to learn the policy language."

Sample Answers to Question 12:

"One big suggestion is that if the tool can automatically detect the congestion and buy the b/w for me, it will be fantastic. I really feel awkward to go buy the bandwidth every time when I re-compile the paper (on Coeus file server), then buy the b/w before I can refresh the PS file display on my machine."
"User involvement in the order of once a minute is infeasible for me. Having me specify rules would be good."
"Perhaps making the above policies more obvious to the users."
"Having some meters in the UI that would indicate how busy the network is."
"Better awareness of accounting actions and how the tokens are being spent."
"Overall, it's less convenient than free access to bandwidth (of course) but very usable to me. I would definitely use this system if PacBell used it for my DSL line at home and would limit my usage to save $$$."
"Allow me to pick from a set of "rules" for automatically paying for bandwidth."
"Emit a beep every time I come close to 90%. I should have an option of choosing between popping up a window, or emitting a beep, or both."

“Allow me to buy bandwidth for a variable amount of time. Sometimes, I want to use bandwidth, and I click on the button. But before I do the preparation for the high bandwidth transfer, the minute runs out. Or, in the middle of a five minutes high bandwidth usage, I might have a 2 minutes low bandwidth usage. I should not have to go back to that window and click again.”

Survey 2

Date: 4/30/02

Questions:

Please use the following scale to answer the following questions.
(1:dislike it a lot, 2:dislike it, 3:OK, 4:like it, 5:like it a lot)

1. From a scale of 1-5, how do you like the overall bandwidth allocation scheme where you need to request different QoSs (in terms of delay) at most once every 15 minutes?

2. From a scale of 1-5, how do you like it when the QoS is in terms of delay instead of peak bandwidth (like in the first experiment back in March)?

3. From a scale of 1-5, how do you like it when the charging granularity is once every 15 minutes instead of once every minute?

4. Should any of the 3 QoSs (RESPONSIVE, MODERATE, SLOW-GOING) be changed? (E.g., SLOW-GOING slower, MODERATE slower, RESPONSIVE more responsive, etc.)

5. Should there be more or less than 3 price levels (RED, YELLOW, GREEN) for the RESPONSIVE?

6. Did having only 24 tokens a day affect your purchasing decision? Should there be more or less tokens?

7. Any other comment about the user interface, experiments, etc?

Sample Answers to Question 1:

”4, meaning I like it better than the previous experiment.”
“2, on the other hand, if I have slow connection at home, I may want to use it at the time of emergency. It will be more useful in that environment.”

Sample Answers to Question 2:

“3, I don't have very strong feelings about it. I think mostly it is because that I always use "moderate" to save token, so that I don't really enjoy much for the "responsive".”

Sample Answers to Question 3:

“5, that help saves lots of click.”

“4, interacting on a 1-minute granularity is fine if I need network access only a few times during the day, 15-minute granularity is better if I use the network more frequently.”

Sample Answers to Question 4:

”Almost Ok.”
“Don't think so.”
“I think they are fine the way they are.”
”4, I think the QoS are good.”

Sample Answers to Question 5:

”No, 3 is fine.”
”3 seems fine.”
“I can handle 3, 2 would be good too.”
“No, I think 3 levels are very good.”

Sample Answers to Question 6:

”SURE!”
“I think its about right.”
“I think 24 is fine.”
“I think it depends on how much each user uses, they can buy more or less based on their own demands.”

Sample Answers to Question 7:

”I found myself requesting the "responsive" mode when I needed it.”
“I liked that I only had to interact with the system every 15 minutes, instead of 1 minute.”
“These experiments seem less annoying than your previous set.”