Services or Infrastructure: Why We Need a Network Service Model

Scott Shenker*, David D. Clark†, and Lixia Zhang*

A Position Statement

Abstract—The Internet approach to networking, with its clean separation between underlying network technologies and overlying applications provided by the internetworking layer, has been extremely successful. However, the single class of best-effort service offered by the current Internet architecture is unable to adequately support the service requirements of multimedia applications. We argue that, to remedy this, the Internet should adopt a more general *service model*; that is, the Internet should offer a variety of qualities of service. We present one proposal for a new Internet service model that contains two forms of real-time service and multiple levels of best-effort service. Interoperability between networks is possible only if the Internet and other internetworking approaches, as well as the various supporting subnet technologies, jointly converge on a common service model. Community networks, if they are seen as part of this general communications infrastructure, should join this search for a common service model. The crucial question, which remains largely unanswered, is whether these community networks, as currently envisioned by their sponsors, are merely means to deliver specific services or are part of a more general communications infrastructure.

I The Internet Architecture

The goal of the Internet is to support a wide range of computer-based applications and allow them to operate over a diverse set of network technologies. The centerpiece of the Internet architecture is the internetworking layer, currently instantiated by the IP protocol. Any two underlying network technologies can interoperate, as long as they both support the IP protocol. Moreover, any application which can run on top of the IP protocol is supported by the Internet. Thus, the internetworking layer provides a clean separation between the applications on top, and the network technologies below; this allows for rapid and independent improvements in both networking technologies and in applications. The internetworking layer enables the Internet to function as a general and evolving infrastructure for data communications.

This current Internet architecture delivers a single class of *best-effort* service. The term best-effort means that the network provides no quantitative assurances about the quality of service delivered but instead merely promises to give its "best" effort. All data packets are treated equivalently (typically in a FIFO manner in the switches), resulting in a single class of service provided to all applications. Upon overload, the service can become arbitrarily bad; however, congestion control algorithms (such as those in TCP) prevent congestion collapse by having traffic sources decrease their sending rate when congestion is detected. The current Internet design has been extremely successful in supporting traditional data applications — such as electronic mail, remote terminal, and file transfer — because they all have very elastic service requirements; while they degrade under increasing delays and packet losses, the degradation is relatively graceful.

However, there is an emerging generation of video and voice applications that have very different service requirements. In particular, interactive video and voice applications often have much more stringent delay requirements than the typical data application. Moreover, these *real-time* applications often have an intrinsic data generation rate and do not back off in the presence of congestion. Because of their different behavior under congestion (data applications back off when congestion is present, but real-time applications typically don't), congestion due to real-time applications can prevent data applications from receiving adequate service. Thus, the current Internet architecture does not adequately support real-time applications, since it can neither meet their service requirements nor protect traditional data applications from real-time traffic.

II Evolving the Internet

Spurred by the rapid spread of multimedia applications and other computer-based real-time applications, there are currently several working groups in the Internet Engineering Task Force considering changes to the basic architecture of the Internet. The goal is to transform the

*Palo Alto Research Center, Xerox Corporation, 3333 Coyote Hill Road, Palo Alto, CA 94304-1314, {*shenker, lixia*}@parc.xerox.com*

†Laboratory for Computer Science, Massachusetts Institute of Technology, 545 Technology Square, Cambridge, MA 02139-4307, *ddc@lcs.mit.edu*. Research at MIT was supported by the Advanced Research Projects Agency, monitored by Fort Huachuca under contract DABT63-92-C-0002. The views expressed here do not reflect the position or policy of the U.S. government.

0-7803-2076-X/94/$4.00 © 1994 IEEE

Internet into an infrastructure capable of supporting the entire spectrum of network-based applications, including both real-time applications and the traditional data applications.

The future Internet will be faced with a broad spectrum of application service requirements. Therefore, the cornerstone of the new Internet architecture must be that packets are not always treated equally; in order to make efficient use of bandwidth, the network should deliver better service to those applications which are more performance-sensitive. There are two basic architectural approaches to providing these different qualities of service.

In one approach, the network determines the application type of the traffic source and then delivers the service accordingly. For example, the network could determine that a particular traffic source is a video stream and then deliver a bounded delay service which it deems appropriate for that application. Note that this approach requires the network to know the various application types and their service requirements. We call this the *implicit* service approach, since the service delivered is not explicitly defined to the end applications (i.e., the network does not tell the application what quality of service it will get but merely delivers what it thinks is appropriate).

In the other approach, the network offers a menu of explicitly defined delivery services. Applications choose their desired service from this menu according to their particular needs. In this *explicit* approach, the network need not have any knowledge of applications or their service requirements.

Recall that the key element of the Internet design philosophy is that the internetworking layer (IP) is independent of the network technology below it, and is independent of the applications implemented on top of it. It is extremely important that we retain this crucial property in the new Internet architecture. Thus, the network should not embody knowledge about the service requirements of applications, because that would then limit the set of future applications to those provided for in the initial design. Instead, the network should offer an explicit menu of services, and applications can choose the service from this menu that best suits their needs[1]. We refer to the set of these offered services as the *service model.*

Since applications must request their service explicitly, the Internet service model will become embedded in end user applications. The forces of backwards compatibility are overpowering in the commercial world; commercial network providers cannot easily make changes that obsolete end user applications. Thus, while the implementing network mechanisms may change, and the suite of resident applications may change, the service model of the Internet will likely to remain extremely stable (though extensible[2]). The service model is thus, in some sense, the most fundamental aspect of the Internet architecture.

In addition to defining a new service model, transforming the Internet architecture also involves the introduction of new protocols and significant changes to the basic forwarding mechanism in switches. We do not address those issues here (see [1, 2] for a discussion of one possible set of mechanisms).

III A Proposed Service Model

The Internet Engineering Task Force is in the very early stages of considering the adoption of a new service model. In this section we briefly sketch one proposal for this new Internet service model, which is described much more fully in [2, 4]. We must emphasize that the Internet may not adopt this particular proposal; our intention is merely to provide an example of a candidate service model and the reasoning used to design it. Moreover, we restrict ourselves to the aspects of the service model that are related to delivery of data along a given path; there are aspects of the service model related to routing and other issues that we do not address here.

The central design criterion for the service model is clear; the service model must adequately address the service needs of present and, to the extent we can foresee their needs, future computer-based applications. Thus, it is useful to start our design effort by first categorizing applications according to their service requirements, and then proposing elements of the service model (that is, a service class) which meets their needs.

We can roughly divide applications into two categories: *elastic* applications and *real-time* applications. Elastic applications adjust easily and flexibly to delays in delivery; that is, a packet arriving earlier helps performance and a packet arriving later hurts performance, but there is no set need for a packet to arrive at a certain time. Typical Internet applications are elastic in nature, and they have been well supported by the current Internet's best-effort service. Thus, we propose continuing to use best-effort service to support such elastic applications. However, there is a wide range of per-packet delay sensitivity among elastic applications. In particular, interactive burst transactions (e.g., Telnet and X-protocol)

[1] Note that there are two aspects of the separation provided by the internetworking layer. Separating the underlying network technology from the internetworking layer allows networks to interwork in a general fashion; separating the internetworking layer from applications above allows the network to support a more general set of applications. One can imagine designs which adopted one of these separation principles without the other, leading to either interoperable but service-specific networks, or incompatible networks which support a variety of applications; neither of these are attractive design choices.

[2] While one cannot remove or alter existing services, one can add new services to the service model.

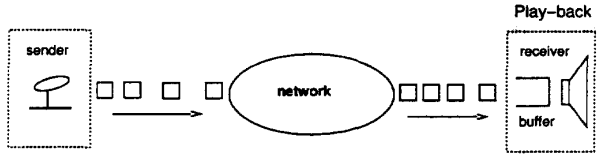

Figure 1: Playback Applications: the receiver plays the data back at the playback point where the playback point is the sum of the generation time and the offset delay.

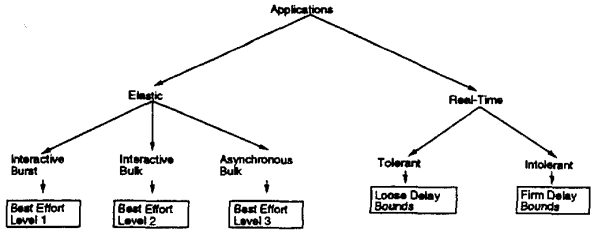

Figure 2: Application Taxonomy: the application classes and the associated service offerings.

are more sensitive to delay than interactive bulk transfers (e.g., FTP), and those are, in turn, more sensitive to delay than asynchronous bulk transfers (e.g., electronic mail and fax). Thus, we propose offering several different classes of best-effort service with different levels of relative delay.

Real-time applications have quite different delay requirements; while the performance of elastic applications degrade gradually with increasing per-packet delays, the performance of real-time applications tends to degrade much more rapidly after the delay reaches some application-specific level. In order to understand this more fully, we consider a prototypical class of real-time applications which we call *playback* applications (see Figure 1). In these applications, the source digitizes some signal and transmits it over the network. The network introduces some delay jitter (i.e., not all packets receive the same delay) and so the receiver buffers the data and plays the signal back at the appropriate moment. This moment is called the playback point and is the generation time plus some essentially fixed offset delay. Data that does not arrive at the receiver before its playback point cannot be played back on time; it is of essentially no use. To choose a reasonable value for the offset delay, an application needs some *a priori* characterization of the maximum delay; this could either be provided by the network in a delay bound, or through the observation of the delays of earlier packets. While our discussion has treated the offset delay as essentially fixed, in reality the application can slowly adjust its offset delay during use as its estimate of the maximal packet delays change. We envision that most future video and voice applications will fit this playback model.

The quality of the "played-back" signal becomes degraded when packet delays exceed the offset delay. Applications vary greatly in their sensitivity to late packets. We can somewhat artificially divide these playback applications into those that are tolerant of occasional dropped or late packets, and those that are not. Intolerant playback applications require an absolutely faithful playing back of the original data, either because the hardware or software is unable to cope with missing data, or because the users are unwilling to risk missing any data. On the other hand, users of tolerant applications, as well as the underlying hardware and software, are prepared to accept occasional losses of data. Most casual telephone conversations are tolerant, since human speech tends to be redundant and, if necessary, users can request that the other participants repeat the missing parts of the conversation.

For intolerant applications, the service model we propose is a firm worst-case bound on delay; this bound must be based on a worst-case analysis (i.e., no assumptions are made about the traffic of other sources) and should not be violated as long as the network switches and links function properly. This service assures that there will be no late packets, so the played-back signal will be identical to the generated signal. Tolerant applications do not need such a reliable bound; for these applications the service model we propose is a loose bound on delay that incorporates predictions about the aggregate traffic load; this bound will occasionally be violated when the predictions are wrong.[3]

Services for both tolerant and intolerant real-time applications involve admission control; before commencing transmission, applications must request service from the network. This service request consists of a traffic descriptor, in which applications specify their traffic load, and a QoS descriptor, in which applications specify the desired quality of service. We envision employing a traffic descriptor that specifies bandwidth and burstiness. After receiving a service request, the network decides whether or not to accept the request based on whether or not it can deliver the desired QoS. In contrast, there is no admission control for the best-effort service classes. Thus, the predominant failure mode for real-time service is that requests can be blocked, and for best-effort service that best-effort packets can be dropped.

Our service model is based on a rather rough taxonomy of applications (see Figure 2 for both the taxonomy and the associated service classes). Clearly our taxonomy is neither precise nor complete. However, we believe that

[3]The terms "firm" and "loose" only distinguish the reliability of the delay bounds, not their magnitudes.

the service model generated from this taxonomy is general enough to meet the needs of the entire spectrum of future Internet applications. Of course, a period of experimenting and rethinking will be needed before the field reaches — if it ever does — a consensus on the appropriate service model for the Interenet.

IV How Many Service Models?

The Internet is not the only packet-switched network. There are numerous other networks based on IPX and other proprietary protocols. There are also plans for deploying wide-area ATM networks[4]. Should each of these networks have its own independent and distinct service model?

Our answer is an emphatic "no". It is extremely important that all of these networks adopt service models that are compatible. Obviously the most efficient solution would be to have a single network architecture. This is unlikely to happen, for a variety of historical and technical reasons. However, it is essential that these networks have, at base, at least some degree of compatibility in service models. There are two reasons for this. First, if these networks are to interwork then each must be able to support the service model of the others. Otherwise, the networks cannot make any assurances about the end-to-end service delivered along paths which traverse more than one of these networks.

Second, the service model determines the kind of service a device expects to get from a network. The model an application uses to describe its traffic stream will affect the manner in which it transmits packets (which, in the case of rate-adaptive coding, may even effect the coding algorithm being used). The model an application uses to describe the service a traffic stream needs will affect the amount of buffering required at the receiving end host. For a device to be able to attach to a wide variety of networks, the kinds of services available should be roughly comparable. If the service model of the Internet is significantly different than the service model of other networks, then separate devices (or expensive "transformers") will be needed for each network. If the service models are very similar, then the ability of these devices to easily migrate between the various networks will be greatly enhanced.

Our presence at this conference is a plea for those involved in the design of non-Internet networks to enter into a dialogue with those of us involved in rearchitecting the Internet; we must attempt to achieve some degree of compatibility in the network service models. If our proposed Internet service model doesn't suit the needs of other networks, we need to know that now so that we can attempt to accommodate those needs while the Internet service model is still in flux.

[4] To the extent that ATM is seen merely as subnet technology to support IP then ATM should support the Internet service model like any other Internet subnet technology. However, many people envision ATM as providing end-to-end service without an intermediate IP layer, and in that case the relationship between the ATM service model and the Internet service model is more problematic.

V Community Networks: Services or Infrastructure?

What are community networks? The unifying characteristic of community networks is that their main purpose is to traverse the "last-mile" to residences. Because of this feature, these community networks will likely play a central role in determining the future of telecommunications in that they will determine the nature of networking service provided to the general population.

The media is replete with announcements for high bandwidth test-bed community networks, and the capital being devoted to these test-beds is staggering. These networks exhibit a wide variety of technical characteristics and are being built by a diverse set of corporate partnerships. There has been much attention paid to the physical characteristics of these networks, such as the relative merits of the fiber-to-the-home and the fiber-to-the-neighborhood-and-coax-to-the-home designs, but there has been little public discussion of the protocol architecture being deployed on top of these physical infrastructures. In particular, the notion of an internetworking layer with a "service model" has been completely absent from these discussions. This omission merely reflects the fact that, to a large degree, these community networks are not being designed to be part of a general network infrastructure, but are seen as merely a means for the developers of the network to deliver a specified set of services.

One can think of today's national telecommunications system as having three components: telephone networks, cable TV systems, and the Internet (under whose umbrella, for convenience, we will include all other wide-area computer networks). These different components have employed rather different technologies, which reflect their different missions. The telephone network and cable systems were built specifically to provide, respectively, telephone service and cable TV. Their mission was to give the end-user access to a particular service. In contrast, the Internet was built to allow computers to exchange data, and great care was taken in the architecture to separate the network from the uses of the data at the endpoints. Thus, while the telephone and cable systems are seen as providers of a particular service, the Internet has always been designed to be a general communications infrastructure. An amazing variety of network services and applications have spontaneously arisen on top of this in-

frastructure; remember that applications like world-wide-web and Mosaic were not anticipated by the designers of the Internet, but are still adequately supported, because the infrastructure is so general in nature (see [3] for a much fuller discussion of the issues raised in this section, and in particular Chapter 2 in [3] describes the design philosophy of "open data networks" like the Internet).

Will community networks merely be a means to provide a few specified services, or will they be part of a general commmunications infrastructure over which we can develop a wide range of new services in the future? If they are service-specific, then the notion of a general service model, which conceptually separates the functions of the network from the functions of the overlying applications, is irrelevant to community networks. However, if they are intended to be part of a general infrastructure, then they should be designed with a service model, and, as we argued above, that service model should be compatible with the other network service models.

Another difference between the Internet and the telephone and cable systems is that the Internet was expressly designed to provide interworking between heterogeneous network technologies. In contrast, cable TV systems and telephone networks systems have no concern about interworking, since either the service is provided locally (cable) or there is a uniform technology (telephone). If community networks are to be merely local providers of service, then the ability to interwork is not a pressing issue. However, if they are seen as part of an overall interoperable infrastructure then interworking is crucial and so community networks should adopt the idea of a universally accepted internetworking layer.

There is one other issue that depends on the service vs. infrastructure question, and it transcends the detailed technical issues we have considered so far. If community networks are general infrastructure, then even though there may only be a single line into each residence there can still be competition in the arena of the delivered services. That is, the technology is capable of supporting open access to the transport services of the network to multiple providers. However, if community networks are seen as providers of specific services, then it is much more likely that each competing service provider will have to install their own line into the home. Clearly, the overall social welfare is increased if we can avoid duplicating the physical infrastructure while still allowing competition in the services arena; this will require open access to the internetworking layer on the lines going into homes. Unfortunately, it may be more profitable for the builders of community networks to discourage competition in the services arena by controlling the access. It is probably no accident that the only open and general purpose infrastructure in our telecommunications system was developed with heavy government subsidies, while the two components that were developed by the private sector are both closed and service-specific. We should not be surprised if unfettered competition in the community network market leads to closed and service-specific architectures.

The perceived mission of community networks — are they providers of specific services or part of the general communications infrastructure — is thus a central issue. Its resolution will have a profound influence on the protocol architecture used in community networks; the role of the internetworking layer and the nature of the service model will both be greatly affected. Perhaps even more importantly, the choice of mission will determine the essential economic character of the services marketplace.

ACKNOWLEDGMENTS

Many of the ideas expressed here were developed as part of the Integrated Services Internet Project, and we would like to thank our many collaborators for their bounteous insight and collegial spirit: Steve Berson, Bob Braden, Deborah Estrin, Shai Herzog, Sugih Jamin, Danny Mitzel, John Wroclawski, and Daniel Zappala. We would also like to thank Marjory Blumenthal and the rest of the NRENaissance study committee of the National Research Council for innumerable helpful discussions.

References

[1] R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: an Overview," RFC 1633.

[2] D. Clark, S. Shenker, and L. Zhang, "Applications in an Integrated Services Packet Network: Architecture and Mechanism," *Proceedings of SIGCOMM '92*, pp. 14-26, 1992.

[3] Computer Science and Telecommunications Board, National Research Council, "Realizing the Information Future: The Internet and Beyond," National Academy Press, Washington, D.C., 1994.

[4] S. Shenker, D. Clark, and L. Zhang, "A Scheduling Service Model and a Scheduling Architecture for an Integrated Services Packet Network," draft available by anonymous ftp as parcftp.xerox.com:pub/net-research/archfin.ps.

[5] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala, "RSVP: A New Resource ReSerVation Protocol," *IEEE Network Magazine*, September 1993.

[6] L. Zhang, R. Braden, D. Estrin, S. Herzog, and S. Jamin, "Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification," in preparation.