Enhancing Wireless Internet Links for Multimedia Services

George C. Polyzos and George Xylomenos
{xgeorge,polyzos}@cs.ucsd.edu
Center for Wireless Communications
and
University of California, San Diego,
Department of Computer Science & Engineering,
La Jolla, California, 92093-0114, U.S.A.

***Abstract*— We describe a novel link layer protocol architecture that aims to enhance the performance of Internet protocols over wireless links, in particular in order to effectively support interactive multimedia applications with varying Quality-of-Service requirements. The degraded performance of these links on the Internet and the inadequacy of existing approaches in overcoming these problems in a protocol independent manner motivate our solution. Our design provides multiple services and performance feedback to higher layers, thus supporting adaptive protocols and applications. In addition, it serves as the basis for Internet evolution towards Quality-of-Service provision, since it can be used to support relevant mechanisms at higher layers, regardless of the underlying hardware.**

I. Introduction

During the past few years, wireless and mobile communications have been evolving and expanding at a rapid pace. A multitude of competing wireless technologies and service models are currently available to the public. As cellular telephony is evolving worldwide towards fully digital systems, the opportunity arises to extend the reach of advanced digital multimedia services to mobile users. Some recent approaches make widespread use of satellite networks, either providing mainly mobile telephony services and Internet-style communications as an overlay, or IP service directly. On the other hand, Wireless Local Area Networks (WLANs) that can transparently link wireless (and possibly mobile) hosts to the Internet are becoming available and inexpensive, while work is underway for WLAN standardization that will eventually enable interoperability among vendors.

Despite their differences, all these systems share common characteristics that set them apart from wired networks and traditional (GEO) satellite-based networks. In contrast to satellite links, cellular and WLAN links exhibit low propagation delays, while even (recent) satellite proposals are based on LEO and MEO satellites with considerably lower (than GEO) propagation delays. However, the most important commonality among them and difference with wired links, is a much higher error rate and packet loss rate and, furthermore, the variability of link characteristics over time, many times a rapid one. In addition, many of these systems (in particular the cellular telephony based ones that are the most prevalent today), suffer from severe bandwidth limitations. Furthermore, although some of these systems might provide ample bandwidth, at least by the standards of traditional, non-multimedia applications, they will certainly be the bottleneck for multimedia applications, especially given the rapid increases in bandwidth and reliability offered by modern fiber optic based wired networks.

At the same time, mobility, in combination with cost considerations, will demand that users effectively employ a set of systems with wide bandwidth capacities. Envisioned future *hierarchical cellular* systems will provide ubiquitous connectivity worldwide by combining different technologies (satellite, cellular telephony, WLANs) to support cells of varying size and communication characteristics. Users will roam between systems, choosing the one to use at each location and point in time based on availability, cost and performance. Roaming between systems based on different technologies requires *vertical handoffs*, in addition to the usual *horizontal handoffs* that take place between cells belonging to the same system. While horizontal handoffs are already a problem for Internet protocols since they cause pauses in communications, vertical handoffs will be even more of a challenge since they will change the long term characteristics of the communications path.

As these emerging wireless systems gain popularity, there is increasing pressure to seamlessly integrate them with the Internet. Superficially, this is an easy task as Internet protocols are designed to accommodate diverse network technologies, making only minimal assumptions about their capabilities. In practice, the design of these protocols has been influenced by assumptions that hold for wired but not wireless networks, thus causing their performance to suffer when employed over wireless links. As an example, a WLAN that suffers only a 2% IP packet loss causes TCP throughout to drop to only 47% of its value in the absence of losses, for local area transfers [2]. The corresponding figure for wide area transfers is only 23%. Even worse, a cellular system that suffers a 2% loss over its short, speech optimized, frames, would suffer IP packet loss of 64%, thus reducing TCP throughput to nearly zero [6].

We believe that these problems present to the networking community both a challenge and an opportunity. The challenge is to overcome the limitations of existing protocols when employed over wireless media. The opportunity is to prohibit similar media dependencies from causing trouble in the future, thus setting the scene for Internet protocol evolution. In this paper we propose enhancements to the link layer that accomplish these goals and at the same time ease the evolution of the Internet towards *Quality of Service* (QoS) support. Our architecture is applicable to any kind of wireless link and parts of it would be important even for wired links since it is instrumental for making QoS support available to higher layers and applications.

II. Related Work

Previous studies have revealed the primary cause of TCP's abysmal performance over wireless links to be its design assumption that *all* losses are due to congestion [4]. Congestion avoidance mechanisms that reduce the sender's transmission rate are thus triggered repeatedly, even for very low error rates, diminishing the effective throughput of TCP and wasting the (scarce) available bandwidth of the wireless link, usually the bottleneck of an end-to-end path. Delays due to mobility induced pauses in communications (during handoffs) are also interpreted as congestion losses, further degrading throughput for mobile systems. Since recovery is traditionally performed at the transport layer of the Internet (or higher), using end-to-end retransmissions, the delay visible to the user due to these losses is considerable, a problem for multimedia applications. End-to-end recovery also exaggerates wireless link problems when the communications path contains more than one wireless link, as retransmissions must repeatedly cross multiple error prone links.

Various approaches are attempting to remedy such problems by modifying the Internet protocols themselves. One scheme enhances transport layer mechanisms so as to avoid handoff induced problems by using mobility indications provided by the network layer [4]. A more comprehensive solution deals with both mobility and high error rates by modifying the transport layer [1] so as to split end-to-end transport connections using as pivot points the routers between the wired and wireless parts of a path. A separate transport protocol, better equiped for dealing with frequent errors, is used over the wireless link. The drawback of this approach is a loss of end-to-end semantics at the transport layer. An alternative approach is to add mechanisms at lower layers that snoop into transport layer segments and retransmit them over the wireless link only, transparently to higher layers [3]. The drawbacks of this scheme include limited gains on the direction from the mobile to the fixed network, and tight coupling with TCP, which makes it the only protocol that can benefit from this solution. These and other end-to-end and local schemes are reviewed in [2] and discussed in [13], while their performance is compared in [2]. This comparative study concludes that local schemes generally exhibit superior performance to end-to-end ones in the test environment.

Numerous *Radio Link Protocols* (RLPs) have been proposed for cellular data services, each enhancing the performance of a specific system using customized link layer mechanisms. Traditional approaches offer complete recovery at the link layer at the expense of greatly varying delays [7]. These delays however can cause TCP to timeout and trigger congestion recovery end-to-end. Another approach tries to avoid such adverse interactions by only offering limited recovery, leaving complete recovery to higher layers [6], if it is required. Although this scheme works well with TCP, it suffers from the same problem as the traditional RLP approaches: it is inflexible since it only offers a single type of service, regardless of higher layer protocol and/or application needs.

III. The Case for a Flexible, Adaptive Link Layer Solution

We believe that the best solution to the problems presented by wireless channels and mobile devices and at the same time a way to introduce QoS-based services in the Internet at the points where it is most critical, is to redesign the link layer architecture. We intend to offer multiple services simultaneously at the link layer, with different QoS operating points provided by service and hardware specific protocol mechanisms. In addition, we will offer feedback in terms of link layer service performance and mobility indications to higher layers, so as to enable QoS-based protocols and applications to provide enhanced services to their users over both wired and wireless parts of the Internet. We present the rationale for these proposals in the next three subsections.

A. The Case for a Link Layer Solution

We claim that the most appropriate solution for wireless link problems lies at the link layer for the following reasons.

1) A link layer solution is a *local* solution, as opposed to transport layer based end-to-end solutions. A local solution has the following advantages:
   - It can be separately optimized for each link, as due to the single underlying link it is possible to know, or discover, the exact link characteristics. Hardware specific details are accessible at the link layer, but hidden from higher layers.
   - Reliability overhead is minimized, since no re-transmissions (for ARQ), parity bits (for FEC), or both (for hybrid schemes) are used over links where this is not necessary, and in particular other wireless or bandwidth limited links on the path.
   - In the case of re-transmission schemes, delay is minimized. In addition, loss detection is faster, due to shorter timeout values and exploitation of link specific characteristics such as in-order transmission [7].
2) A properly designed, i.e. transport layer independent, link layer scheme is a *universal* solution, that can support multiple higher layer protocols and applications:
   - It is compatible with and applicable to all higher-level protocols. In the case of Internet traffic it can be applied not only to TCP, but also to UDP, which is the protocol of choice for real-time and multipoint Internet multimedia applications [8].
   - It can support future higher layers and applications that have not even been conceived yet, by extending the already implemented mechanisms only at links where this is required, in a manner consistent with emerging needs.

B. The Case for Multiple QoS Points at the Link Layer

The main point of our proposal, however, is the provision of multiple, parameterized points of QoS simultaneously over a single link. Our rationale for this is the following.

1) Different higher layer protocols and applications have varying requirements and could benefit from different trade-offs among metrics such as throughput, delay, and reliability. In particular, many multimedia applications:
   - Are very bandwidth-intensive, at least for some of their media streams, a characteristic more important on bandwidth starved wireless links.
   - Are possibly error tolerant to some extent, in that they can recover from occasional losses by gracefully degrading the service offered to the user.
   - Have different delay and error sensitivities. This may even be the case between the multiple media streams used by a single application.

   This variability in service requirements among media allows a broad spectrum of error control techniques to be applied. In addition, the need for high bandwidth combined with the possibility of error tolerance and the fact that various media or media components are not independent, introduces a new spectrum of possibilities in the form of traffic priority policies. For example, in some cases it might be preferable to have a high quality audio stream and a variable quality video stream, which might be able to react quickly to a scene change under good channel conditions, but could be even completely blocked (replaced by a frozen frame) in case the bandwidth is momentarily needed for the transfer of a document (with full reliability and low delay).
2) Full ARQ-based reliability at the link layer combined with the use of TCP (or other reliable transport protocols) is inefficient and can be detrimental when the various timers used at the link and transport layers are not compatible. To avoid adverse interactions when enhancing link layer services, it is preferable to offer a QoS level appropriate to each type of higher layer protocol or application. Thus, to support the multiprotocol nature of the existing and future Internet, we need to provide an extensible set of link layer services that cater to varying needs.

Note that our approach would allow physical links to be run at high(er) bit-error rates than with classic RLPs, providing much more bandwidth, and would suggest to avoid the use of time diversity (such as bit interleaving), which is usually detrimental for applications that can provide their own processing and buffering, as it introduces considerable extra delay in the communications path.

C. The Case for Feedback from the Link Layer

There are two types of feedback that would be beneficial to higher layers in the framework of our architecture.

1) Per service performance metrics such as delay, throughput and reliability achieved, updated dynamically as link characteristics vary. Hardware independent metrics can be used both to select the appropriate service for each need without any knowledge of service implementations, and to assess the performance of an end-to-end path before and during a session, so as to adapt higher layer and application mechanisms according to prevailing conditions.
2) Notifications in the form of upcalls to higher layers can be used to inform interested parties of events such as disconnections and resets, that may indicate the beginning and end of handoffs. The link layer, being aware of hardware details, can detect such events fast, while higher layers can filter them into authoritative horizontal or vertical handoff notifications, and act accordingly.

IV. Link Layer Architecture

We are building a family of link layer protocols optimized for wireless links, supporting Internet protocols [9], [12]. These protocols are in turn encapsulated in a generic link layer that co-operates with the remainder of the protocol stack to provide QoS-based services. The following subsections describe our link layer architecture in more detail.

A. Protocol Stack Framework

Figure 1 shows a rough framework for a protocol stack based on our enhanced link layer and its services. Our framework handles applications with varying requirements, that in turn make use of different transport layer protocols, such as TCP and UDP. It is possible (but not required) that higher layers will also provide QoS-based services to the transport layer, for example a *Class Based Queuing* (CBQ) mechanism can be used to ensure sharing of gateway transmission bandwidth in a predefined manner during periods of congestion [5], [11]. Such services can be established end-to-end using RSVP as the setup and negotiation mechanism [15]. The goal of our data link layer is to enhance the service offered to each class according to its requirements, without interfering with higher layer QoS provisions. For example, an application could set up an end-to-end path using RSVP that would guarantee it a specific amount of bandwidth, with CBQ enforcing the guarantees at each gateway. Our link layer would preserve the priorities and link shares decided upon by RSVP/CBQ, and enhance the QoS of the underlying wireless link based on application requirements, for example by performing ARQ-based error recovery.

Each service provided by our link layer employs a mix of mechanisms specifically optimized for the underlying medium. The mechanisms adapt to current link conditions in an attempt to hide local performance variations from higher layers as far as possible. The protocols thus support multiple QoS classes *simultaneously* in order to support various higher layer protocols and applications. The QoS classes provided are meant to satisfy generic requirements rather than fit the specific characteristics of TCP or UDP or particular applications. With this approach, it is possible to optimize the performance of existing protocols and applications without penalizing future ones. Each mechanism/protocol is separated from all others, effectively operating over its own virtual private link, for ease of programming. All services are in turn encapsulated in a generic link layer that presents a common interface to higher layers regardless of underlying links and service implementations.

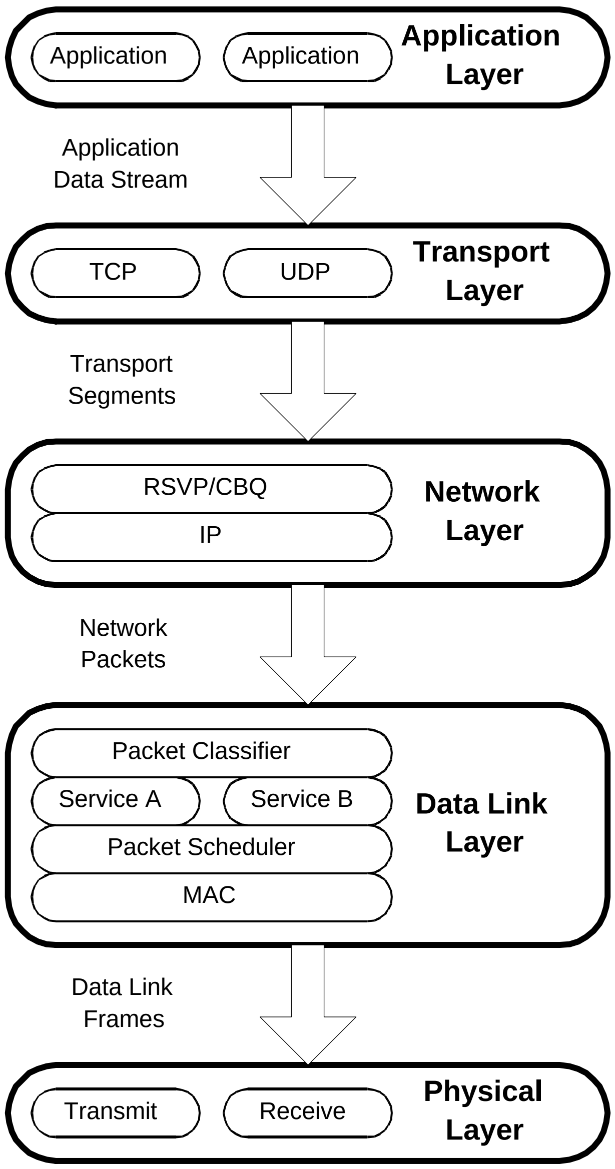

Fig. 1. Link layer based protocol stack framework.

The link layer also includes a *packet classifier* and a *packet scheduler* above and below the service protocols, respectively. The classifier decides which service should receive each network layer packet, while the scheduler chooses the service whose turn is to transmit, and sends its next packet to the MAC layer for transmission on the link. In the reverse direction, packets from the MAC layer are simply distributed to the appropriate services based on link layer header fields, and are then propagated upwards towards the network layer. More details on the internal design of the link layer are presented in the next subsection.

B. *Link Layer Internal Design*

The internal design of the proposed link layer is shown in more detail in Figure 2. The packet classifier decides which service should handle each packet, and sends it there for further

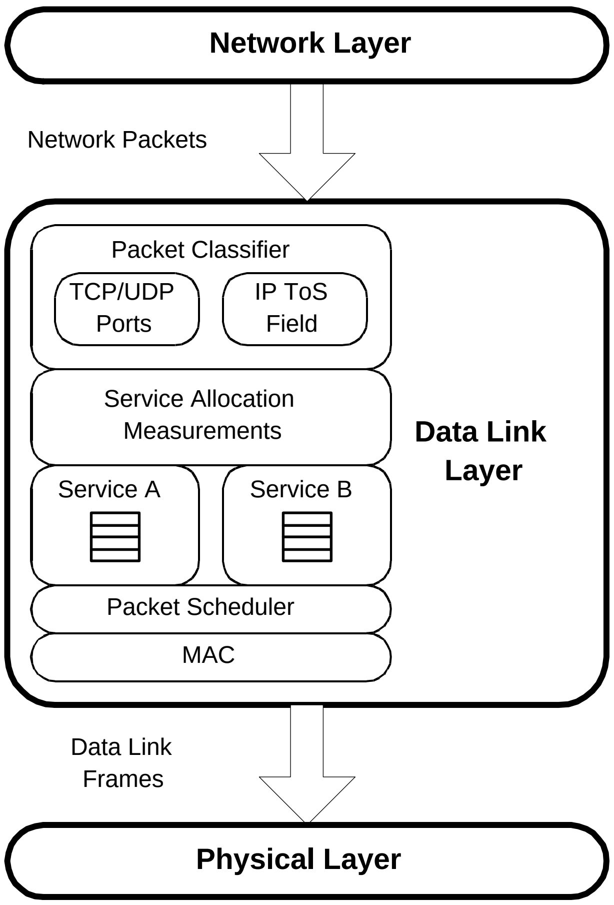

Fig. 2. Internal design of the link layer.

processing. Static rules can be set up to match IPv4 *Type of Service* (ToS) bits, IPv6 flow IDs, or protocol/port pairs (denoting applications that use well known ports), with underlying services. If QoS-based higher layers are used, the classifier may be able to set up rules for classification based on explicit requests made dynamically by, for example, RSVP. The appropriate service can be selected by examining the service performance metrics exported by the link layer, thus making the higher layers independent of the particular services provided and their implementations. A separate module inserted below the classifier measures how much share of the bandwidth was allocated to each service by examining the packets and their sizes on exit from the classifier over a (short) time interval. These link shares are enforced over the next time interval by the packet scheduler so as to preserve the sharing dictated by higher layers.

After each service processes a packet handed to it by the classifier in an appropriate manner, it places it in its own FIFO queue. The packet scheduler then selects the packet at the head of the queue corresponding to the service whose turn is to transmit a packet, encapsulates it into a generic link layer frame, and sends it to the MAC layer for transmission. The scheduler uses the measurements made in the previous time interval in order to decide which service to select for the next transmission. If a service needs to retransmit packets (ARQ) or encode them (FEQ), or both, it simply places the appropriate packets in its

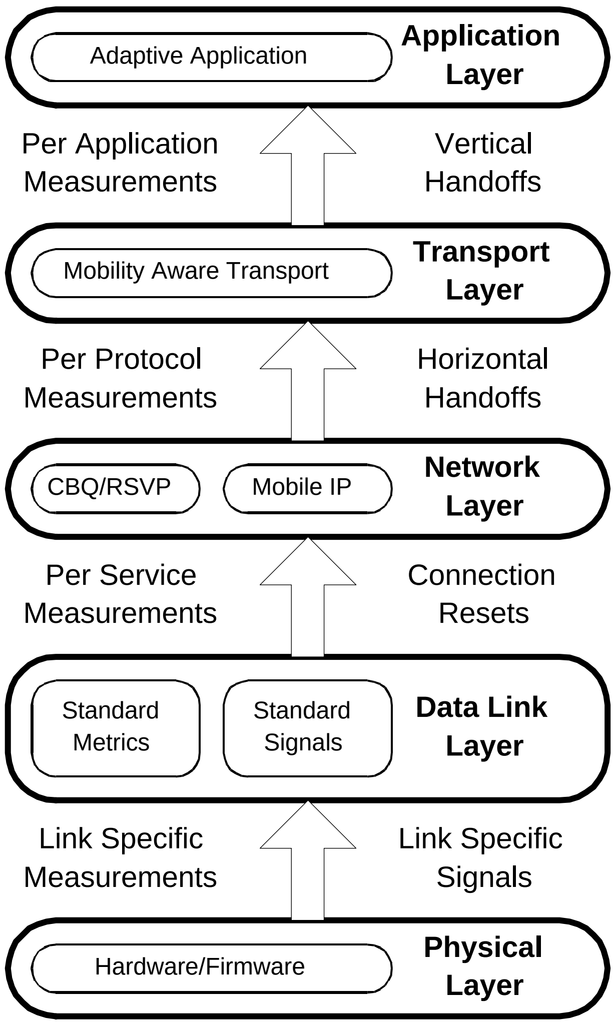

Fig. 3. Feedback propagation through the protocol stack.

queue, so that if a service inflates its data stream with recovery information, it does not affect link sharing, as the scheduler ignores such details. The scheduler and the classifier are thus independent of the services and can be reused, while service implementations are customized to underlying link characteristics.

C. Link Layer Interface

To integrate our protocols into a next generation protocol stack, they all have the same interface to higher layers, thus hiding their link specific mechanisms. Higher layers supporting QoS classes can then use the most appropriate link layer services without media specific knowledge. To make dynamic service selection possible and support adaptive higher layers and applications, the protocols export information on link performance metrics and provide notifications about events that could signify medium term performance variations. Thus, in addition to the service requests that flow downwards through the protocol stack, information also flows upwards as shown in Figure 3. The hardware and firmware of the physical layer may give some link specific feedback to the link layer, which processes it along with its own measurements of recent performance metrics for each service, and propagates such information upwards in a link independent manner.

For each service *separately* the link layer tracks and reports performance metrics such as reliability, delay and throughput, updating them regularly by combining current and past measurements. These metrics are normalized so as to reflect the performance of each service as if it were using the link exclusively, making them independent of the actual link shares received by each service in the past. The exported link performance metrics will allow higher layers to estimate end-to-end path capabilities and performance, enabling them in turn to offer flexible QoS classes to their users. Even if only some links on a path offer such advanced characteristics, suitable higher layers can compose more flexible end-to-end services using service substitutions [10] negotiated by a service establishment protocol such as RSVP. The measurements can be used directly by any higher layer, or modified by intermediate layers to account for their own mechanisms (such as error recovery).

In parallel, the link layer notifies interested parties in higher layers of events, such as disconnections and resets at the link level, which it can discover either using its own control mechanisms or by receiving explicit control signals from the physical layer. These events are propagated in a standardized manner through the protocol stack, so that higher layers can further refine them and/or adapt their operation accordingly. For example, a disconnect notification and a connect notification from two different links shows to mobile IP that a handoff has taken place and routing tables should be modified. Depending on the links used, mobile IP distinguishes between horizontal and vertical handoffs. Horizontal handoffs can be dealt with at the transport layer, for example by freezing its timers until the handoff completes to avoid false loss indications, while vertical handoffs from system to system that change the long term behavior of the path are propagated to applications, which could modify their data streams to compensate, for example by switching to a different video resolution. The actual characteristics of the new link can be discovered by looking at the metrics exported by the link layer, and services can then be renegotiated appropriately.

D. Deployment Considerations

Deployment of the proposed link layer architecture is easy, as it is transparent to higher and lower layers and only requires local modifications. Both endpoints of a single wireless link are usually under the control of a single administrative entity, so both endpoints could update their link layers at the same time. Although the services provided are generic (each one trades-off reliability, delay and throughput in a different manner [12]) they can be used to efficiently support existing protocols (such as TCP) and multimedia applications (such as video and/or audio over UDP), making them attractive to equipment vendors and service providers that could deploy them with the incentive of gaining a competitive advantage on the market over their competitors. Improvements in end-to-end TCP service using enhanced link layers have been demonstrated previously [2]. Our protocols in turn will extend existing work to non TCP based applications and mixed protocol and application environments.

When in place, our link layer architecture will also ease the deployment of QoS based higher layers, since it was designed to interoperate with network layer QoS mechanisms, it provides additional information for service setup on end-to-end paths, and can be used to detect service disruptions and changes due to horizontal and vertical handoffs. The encapsulation of specific services into a generic link layer eases deployment over diverse wireless links. Furthermore, the framework discussed previously can be adapted for wired links as well, in order to provide feedback in the form of performance metrics in a standardized manner over each link of an end-to-end path. It is unclear whether multiple services and upward signaling would be needed for wired links.

E. Experimental Testbed

Our implementation effort is based on a testbed that includes both desktop and portable PCs running the Linux OS. These systems are interconnected using a wired LAN for control, while a wireless LAN is used for experimentation. IS-95 cellular data links will be added when the service becomes available in our area. For the operating system chosen, we use devices on a virtual file system to export link metrics, signals to deliver notifications (upcalls) to registered parties, and (extended) socket options to set ToS values directly from applications, so as to ease classification of packets at each link (classifiers are configured statically in this case, matching each ToS pattern to a specific service). Linux, besides being publicly available in source code form and quite widely used, incorporates many useful characteristics such as fine granularity timers.

Besides experiments on the wireless testbed, we are using a wireless link emulator for more controlled (and reproducible) tests. The emulator mimics the error behavior of a wireless link from within the kernel. Many error models can be programmed into the emulator, the calibration of which is an ongoing effort in our testbed. Our measurements using the raw services of the wireless LAN, besides giving us ample material for refining the emulator, have also uncovered some interesting effects that are documented separately in more detail [14]. For example, TCP traffic under certain conditions suffers from excessive undetected collisions that dramatically reduce performance, an effect not previously documented since most existing results are based on unidirectional UDP tests. Our results along with other reports in the literature, imply that there exists considerable room for improving application level performance using link layer enhancements.

V. Summary

We have described a link layer based protocol architecture aiming to enhance the performance of multimedia applications over Internet connections that incorporate wireless links. Our architecture accommodates applications and protocols with varying needs by providing multiple services simultaneously at the wireless link, with different performance characteristics. We discussed the problems faced today by such applications and the inadequacies of existing solutions. We then made our case for the main points in our design, and provided a detailed description of our link layer based model. We believe that besides offering enhanced services to higher layers our architecture can also be used as the basis for the evolution of the Internet to Quality-of-Service provision, due to its medium independent QoS-based interface and its feedback to higher layers, attributes important for both wireless and wired links.

References

[1] B.R. Badrinath, A. Bakre, T. Imielinski, and R. Marantz. Handling mobile clients: A case for indirect interaction. In *Proceedings of the 4th Workshop on Workstation Operating Systems*, pages 91–97, October 1993.

[2] H. Balakrishnan, V.N. Padmanabhan, S. Seshan, and R.H. Katz. A comparison of mechanisms for improving TCP performance over wireless links. In *Proceedings of the ACM SIGCOMM '96*, pages 256–267, August 1996.

[3] H. Balakrishnan, S. Seshan, and R.H. Katz. Improving reliable transport and handoff performance in cellular wireless networks. *Wireless Networks*, 1(4):469–481, 1995.

[4] R. Cáceres and L. Iftode. Improving the performance of reliable transport protocols in mobile computing environments. *IEEE Journal on Selected Areas in Communications*, 13(5):850–857, June 1995.

[5] S. Floyd and V. Jacobson. Link-sharing and resource management models for packet networks. *IEEE/ACM Transactions on Networking*, 3(4):365–386, August 1995.

[6] P. Karn. The Qualcomm CDMA digital cellular system. In *Proceedings of the USENIX Mobile and Location-Independent Computing Symposium*, pages 35–39, August 1993.

[7] S. Nanda, R. Ejzak, and B.T. Doshi. A retransmission scheme for circuit-mode data on wireless links. *IEEE Journal on Selected Areas in Communications*, 12(8):1338–1352, October 1994.

[8] J.C. Pasquale, G.C. Polyzos, and G. Xylomenos. The multimedia multicasting problem. *ACM/Springer-Verlag Multimedia Systems*, 6(1):43–59, January 1998.

[9] G.C. Polyzos. A wireless link layer protocol for Internet traffic. In *Proceedings of the International Conference on Telecommunications (ICT '97)*, pages 253–256, April 1997.

[10] S. Shenker and L. Breslau. Two issues in reservation establishment. In *Proceedings of the ACM SIGCOMM '95*, pages 14–26, October 1995.

[11] I. Wakeman, A. Ghosh, J. Crowcroft, V. Jacobson, and S. Floyd. Implementing real time packet forwarding policies using streams. In *Proceedings of the 1995 Usenix Winter Technical Conference*, pages 71–82, January 1995.

[12] G. Xylomenos and G.C. Polyzos. Enhancing wireless Internet links. In *Proceedings of the International Conference on Telecommunications (ICT '98)*, pages 394–398, June 1998.

[13] G. Xylomenos and G.C. Polyzos. Internet wireless link performance. Technical Report CWC-9801, Center for Wireless Communications, University of California, San Diego, January 1998.

[14] G. Xylomenos and G.C. Polyzos. TCP & UDP performance over a 2.4 GHz wireless LAN. Technical Report CWC-9805, Center for Wireless Communications, University of California, San Diego, 1998.

[15] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: A new resource reservation protocol. *IEEE Network*, 7(5):8–18, September 1993.