IEEE
COMMUNICATIONS
SURVEYS
The Electronic Magazine of
Original Peer-Reviewed Survey Articles

TCP IN WIRED-CUM-WIRELESS ENVIRONMENTS

KOSTAS PENTIKOUSIS, STATE UNIVERSITY OF NEW YORK AT STONY BROOK

ABSTRACT

The Internet has evolved during the last decade, reaching a larger number of users and encompassing several new technologies. New classes of hosts such as mobile devices are gaining popularity, while the transmission media become more heterogeneous. Wireless networks exhibit different characteristics than wired ones. Mobile hosts have different needs and limitations than desktop computers. TCP has served well the wired Internet for almost 20 years, but is not ready for wired-cum-wireless environments. This article presents the challenges that must be met in order to provide reliable transport services to all hosts regardless of the type of network connectivity used. It surveys recently proposed solutions and evaluates them with respect to a wired-cum-wireless environment.

The Transmission Control Protocol (TCP) [1–3] is a reliable, connection-oriented, full-duplex, byte-stream, transport-layer protocol [4–6]. It is an end-to-end protocol [7] that supports flow and congestion control, and is used by many end-user applications, including Web browsers and e-mail clients. In fact, the vast majority of today's Internet traffic uses TCP [8, 9].

TCP was designed for wired networks and has been highly tuned over the years. Although TCP is very efficient on wired networks, it has been shown to perform poorly on wireless networks. As wireless networks connect to the Internet, wired-cum-wireless environments with very distinct characteristics emerge. Such environments are expected to become the norm in the near future. This article presents the challenges that these networks pose and examines recently proposed solutions.

We present the general characteristics of wired-cum-wireless networks and the issues and problems that these pose for TCP. Some of these issues already exist in the wired Internet, but are exacerbated by the interplay between hybrid wired and wireless in the emerging environment. The wireless aspect, in itself, also poses a whole set of new problems. We examine different solutions proposed by researchers in recent years. Finally, the Appendix provides more information on the services that wireless networks offer.

A WIRED-CUM-WIRELESS INTERNET

The Internet has been in constant evolution since the mid 1980s, after the introduction of TCP [1]. Lately though, the Internet has become even more heterogeneous. Today, powerful PCs and workstations coexist with WebTVs [10], wireless phones, and personal digital assistants (PDAs) [11]. Although it is desirable in principle to provide to all hosts the same kind of network services, it is an open question whether it is possible to do so.

Diversification in end-host capabilities limits to some extent the applications that would be running on each category of devices. Differences in computing power, memory size, input and output devices, in addition to mobility and power consumption issues, determine the potential uses of each end-host category. At the same time, end hosts use a far larger spectrum of networking technologies to connect to the Internet, including traditional wires and optical fibers, as well as wireless and infrared media.

Users need reliable transmissions for Web browsing, e-mail, file transfers, and database access, and TCP is the dominant reliable transport protocol on top of which all of these services run [9]. However, TCP was designed [1], and later modified as needed (e.g., [2, 3]), with certain assumptions in mind. For example, segment losses are assumed to be due to congestion, because in wired media transmission errors are relatively rare. This is not true for wireless media where, due to fading channels and user mobility, transmission losses are more frequent. Since certain assumptions behind TCP's highly tuned algorithms do not hold for transmission-heterogeneous media, current TCP implementations do not perform well in such environments [12–15]. In addition to random errors and hand-offs, TCP must also cope with connections that, in certain cases, exhibit limited bandwidth and large round trip times (RTTs). Moreover, for mobile battery-operated devices, power consumption is an important issue that is not addressed by current versions of TCP.

TCP assumes that the underlying network infrastructure has limited service capabilities (i.e., it provides a best-effort, unreliable packet delivery service). The Internet community has preferred so far a more end-to-end approach when

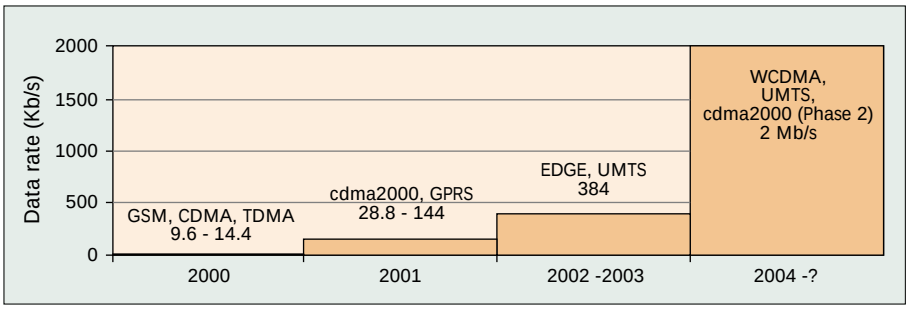

FIGURE 1. *Evolution of cellular networks data rates.*

attempting to provide reliability to applications [7]. With the advances in hardware over the last decade it has become apparent that networks can offer reliable services (e.g., ATM), or at least that this is becoming feasible. Therefore, adding more functionality at the layers below TCP is an active area of research.

END-USER WIRELESS NETWORKS

End-user wireless networks, i.e., networks that are not part of a backbone, can be classified using different criteria. Wireless networks can be classified as local area networks (LANs) or wide area networks (WANs), depending on the service area of the access point (or base station). They can also be classified according to the type of service they offer to the user, as explained in the Appendix.

Wireless LANs — Wireless LANs can provide sufficient bandwidth for office applications but relatively limited mobility: typically the user may roam inside a building or a campus. Wireless LANs have not yet replaced wired LANs, although they are used as an extension to wired LANs. They offer a great service to a number of vertical markets like retail stores, warehouses, and manufacturing plants [16]. There are two main standards: the High Performance Radio Local Area Network (HIPERLAN) standard, and the IEEE 802.11 standard, also known as wireless Ethernet.

A European Telecommunications Standard Institute (ETSI) committee designed the HIPERLAN Type 1 standard (HIPERLAN/1) [17]. HIPERLAN/1 uses a Carrier Sense Multiple Access (CSMA) Medium Access Control (MAC) protocol [6, 18], and can achieve net data rates of up to 20 Mb/s in a 50-meter range, or up to 1 Mb/s in an 800-meter range. HIPERLAN/1 handles mobile hosts that can be moving at up to 36 km/h, and can provide quality of service based on the different categories of data [16]. The standard includes a provision for handoff handling, but does not provide the actual specification for this *per se*. HIPERLAN/1 was designed so that it can offer small delays and is based on small messages that are exchanged relatively frequently. This fact, in combination with the relatively high bandwidth, qualifies HIPERLAN not only for data transfers but for other services as well, such as teleconferencing, video, and medical data transmissions [16].

The original IEEE 802.11 wireless LAN standard [19] was designed to achieve raw bit rates of 1 Mb/s or 2 Mb/s within a 100-meter range. Later, IEEE published two supplements to this standard: 802.11a (40 Mb/s in the 5.8 GHz band) and 802.11b (11 Mb/s in the 2.4 GHz band). IEEE 802.11 uses a CSMA with Collision Avoidance (CSMA/CA) scheme [16, 19] to regulate access to the wireless medium. The maximum mobile host roaming speed in 802.11 is 90 Km/h. Pahlavan *et al.* present the handoff procedure in 802.11 and compare it with other mobile data networks [20]. Most off-the-shelf products, such as the ORiNOCO RG-1000 Residential Gateway, are compliant with IEEE 802.11b and operate in the unlicensed 2.4 GHz band, allowing the user to roam up to 150 meters at 11 Mb/s in open environments, while in closed environments the maximum range at the lowest fallback rate (1 Mb/s) is about 50 m [21].

The term *wide area wireless data networks* (WAWDN) refers to WANs that are dedicated to data traffic [16]. The current generation of WAWDN includes the cellular digital packet data (CDPD) system and the general packet radio service (GPRS), and uses packet switching. WAWDN provide bit rates of one to two orders of magnitude less than wireless LANs. On the other hand, a single base station in such WANs can cover a much larger area, usually some tens of square kilometers. The mobile user can enjoy network services at roaming speeds higher than in the case of wireless LANs. The mobile host, typically a laptop, is equipped with a radio modem, which is used to connect to the network.

Finally, cellular networks that can handle mixed traffic, i.e., both voice and data, can also be used for wireless access. First-generation (analog) cellular networks like AMPS became obsolete due to the limited capacity and services they offer. Second generation cellular networks, like GSM/DCS 1800/PCS 1900, D-AMPS (IS-54), and IS-95 [16, 22, 23], are widely deployed in many countries [24]. Such networks can be used for data transfers, but are not very economical. The user must dial up to achieve connectivity with the network and is usually charged according to airtime spent, not the amount of data transmitted or received. The offered bandwidth is even less than in the case of WAWDN. For example, GSM offers bit rates up to 9.6 kb/s (with GSM Phase 2+ up to 14.4 kb/s) [25], while IS-95 provides rates up to 19.2 kb/s [16]. Note that in the case of a WAWDN (e.g., CDPD), the user is usually charged by the amount of data transferred, so it is economically reasonable for a user to be constantly connected and able to transfer data any time without the need to dial up [26]. Third generation cellular networks (e.g., UMTS [27] and 3G WCDMA [28]), offer significantly higher bit rates (Fig. 1).

The Appendix includes more information on the services and specifications of the aforementioned wireless networks.

THE CASE OF TCP IN THE WIRELESS ENVIRONMENT

We will not delve into the details of TCP [1–3] in this survey. The interested reader is referred to one of several excellent books on TCP, such as [4, 5]. We will, however, discuss the parameters that affect the performance of TCP in a wired-cum-wireless environment.

Limited Bandwidth — As explained previously, wireless wide area networks offer limited data rates. For example, CDPD networks [26] offer a raw bit rate of 19.2 kb/s, which is shared amongst up to 30 users. On the other hand, the current generation of wireless LAN standards offers sufficient bandwidth. For example, the IEEE 802.11b standard offers raw bit rates of up to 11 Mb/s, while HIPERLAN offers up to 20 Mb/s. Third-generation cellular networks promise sufficient bandwidth even for multimedia applications, but will not materialize on a wide scale before 2004.

Long Round Trip Times — In general, wireless media exhibit longer latency delays than wired media [25, 29–31]. This affects TCP throughput and increases the interactive delays

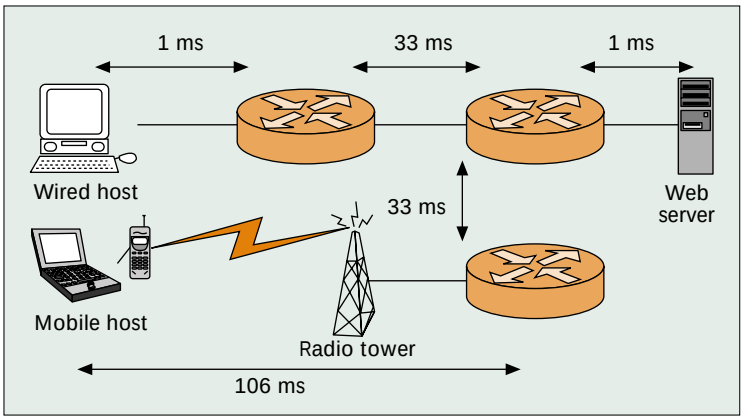

FIGURE 2. *A wired and a mobile host access the same web server.*

perceived by the user. In addition, Lakshman and Madhow [32] show that under certain scenarios TCP Tahoe is "unfair" for connections with longer RTTs. Newer versions of TCP alleviate a number of TCP's inefficiencies, but still increase the congestion window proportionally to the rate of incoming acknowledgements (ACKs). For example, consider the scenario depicted in Fig. 2, where two hosts attempt to access the same Web server. Both hosts are three hops from the Web server. A client connected through an all-wired path is 35 ms "away" from the Web server, while the second client is a mobile host with four times the RTT of the wired host.

Assume now that both clients simultaneously attempt to access the same object on the Web server. Figure 3 illustrates the time diagram for both TCP connections, as well as the evolution of the congestion window for both connections. By the time the connection initiated from the mobile host makes its HTTP request, the wired host has already downloaded part of the object, and its connection has a congestion window of six segments. Because the two connections experience different growth rates in their congestion window, they will achieve different throughput levels. In particular, the connection with the longer RTT (i.e., the mobile host) will experience a much more moderate growth in its congestion window (as illustrated in Fig. 3), and consequently in its sending window, which will translate into smaller bandwidth share, yielding smaller throughput.

Random Losses — Wireless media are more prone to transmission losses, for example due to fading channels, shadowing, etc. TCP was designed with a wired medium (copper cables) in mind, which has bit error rates (BER) on the order of 10^{-6}–10^{-8}. This translates into a segment loss rate of about 1.2–0.012 percent for 1500-byte segments. BER are much higher in the wireless domain, usually on the order of 10^{-3}, and sometimes as high as 10^{-1} [29]. With BER on the order of 10^{-3} the packet loss rate is an order of magnitude more in a wireless environment (approximately 12 percent). Forward error correction (FEC) can be employed to bring the false alarm error rate (FAER) down even to the order of 10^{-6} [29, 33]. However, FEC achieves such low FAER only under certain conditions, and results in a huge expense of bandwidth. Given that bandwidth is very limited with present wireless networks, "strong" FEC is usually not preferred. In addition, FEC cannot solve all problems because terrain type and natural and man-made objects can handicap wireless connectivity altogether.

Random losses are the most prominent problem addressed in the literature, which is why many of the solutions aim at alleviating this deficiency in TCP. The problem with retransmissions does not lie so much in the very fact of sending a segment again. After all, the segment was lost, so the only way to deliver it to the receiver is to resend it. The problem is that a lost segment triggers congestion avoidance mechanisms [3], which essentially make the sender's window much smaller. In this way, a transient error leads TCP to back off too much and not be able to sustain a good throughput level [15].

User Mobility — Wireless networks enable the user to move around. Perkins [34] makes a clear distinction between *portability* and *mobility*. In the first case, the user may use, for example, a laptop that can plug into the network at several different access points. However, this implies that there is some time (practically, the time spent for the transition to another access point) during which the user does not enjoy network services. The term *mobility* means that the user can roam freely and at the same time seamlessly enjoy network services without interruptions. A number of issues related to user mobility led to the creation of the Mobile IP Working Group by the Internet Engineering Task Force (IETF) [35]. Wireless networks are usually designed in a cellular fashion, where each cell covers a particular area. Users inside this area are serviced from a single base station (BS) or access point (AP), a host that is connected to the wired network and offers wireless connectivity to mobile hosts. When a mobile host is moving from one cell to another a procedure named handoff (or handover) must be followed. During a handoff, all necessary information must be transferred between the two base stations so that the mobile host can continue to be connected. Note that Mobile IP is designed with the assumption that mobile nodes do not change BSs more frequently than once per second [36]. Caceres and Iftode were among the first to study the implications of mobility on TCP performance. The consensus of the research community is that it is desirable for reliable transport protocols to be able to differentiate between congestion-related, transmission (or random) losses and motion-related losses.

Short Flows — Web browsing and e-mail account for a large majority of today's Internet traffic [8, 9]. These services usually include the transmission of rather small amounts of data [37]. This means that when the application-layer protocol opens a TCP connection for the transfer, there is a very large probability that the whole transfer is completed while the TCP sender is still in the slow start phase. Therefore, the TCP connection never manages to fully utilize the available bandwidth. Savage *et al.* [38], for example, claim that a 10 KB download under perfect conditions and with infinite bandwidth will not proceed with a throughput faster than 300 kb/s! Indeed, if the client is 35 ms "away" from the server (Fig. 2), the 10 KB transfer will need a 70 ms RTT for connection establishment, and at least another three RTTs for the actual transfer (Fig. 3). This adds up to 280 ms, making the user-perceived throughput 286 kb/s.

Application-layer Protocol Design — Some application-layer protocols make poor use of TCP, leading to considerable performance degradation. Of course, evolution in application-level protocols that are aware of TCP workings allow for improved performance in some cases. For instance, HTTP/1.0 [39], the first version of the Hypertext Transfer Protocol, opens a new TCP connection for the retrieval of each object in an HTML (Hypertext Markup Language) document: a page with three images, for example, involves the establishment of four distinct TCP connections [40, 41]. HTTP/1.1 [42] solves this problem by introducing persistent connections: the first TCP connection that is used to fetch the

base HTML file is also used to fetch the three image objects as well.

Power Consumption — For battery-operated devices like laptops, PDAs, and wireless phones, power consumption is a very important factor. Typically, communicating over a wireless medium consumes more battery power than CPU processing [30]. Of course, TCP was not designed with energy expenditure in mind. However, TCP has been shown in several studies, including [43, 44], to be very efficient in terms of retransmitted segments. TCP manages to have a very low overhead (i.e., it does not waste bandwidth). For example, in [43] TCP is shown to achieve almost perfect goodput (defined there as the ratio of the actually transmitted bytes divided by the number of bytes to be transferred). On the other hand, energy efficiency does not only depend on the amount of avoidable extra data, but also on the total duration of the connection. Prolonged communication times lead to power consumption also [15]. It is important to note that until recently studies did not take power consumption under consideration [45–49]. Although it is for the market to decide, solutions that do take power consumption into account would have a clear-cut advantage.

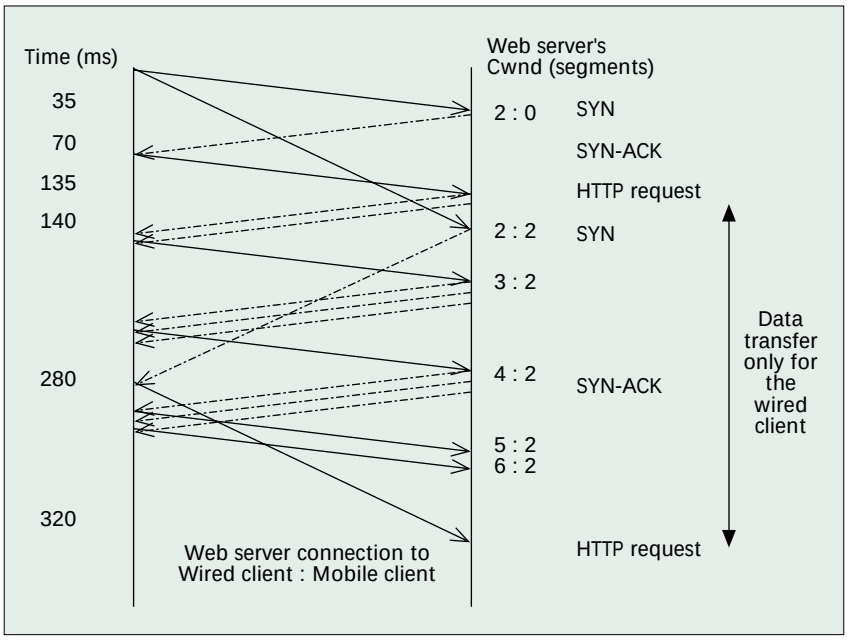

FIGURE 3. *Time diagram of simultaneous HTTP requests from two hosts with different RTTs. TCP initial window (IW) is set to 2 segments, the client uses delayed acknowledgments, and no losses occur. For simplicity the congestion window (cwnd) is presented in terms of number of segments instead of bytes.*

New wireless-oriented protocols, like [31, 47, 50], can be designed with all the above considerations in mind, and therefore perform much better than TCP under a number of scenarios. Although specialized transport protocols, or even protocol stacks like WAP, may provide the framework for the development of wireless browsing, e-mail, and calendar services, users will still require access to all other critical information like databases, file sharing, and other network services. Today this can be done only on top of TCP. Otherwise, specialized new applications must be built for all currently existing applications, which is rather uneconomical, and maybe not even realistic. TCP has an advantage over any new proposal because it has proven to be robust over the years, it is widely deployed, there is a vast store of experience with it, and it eases the task of maintaining compatibility with the rest of the Internet.

TAXONOMY OF SOLUTIONS

The research community has been very active in trying to solve the issues related to TCP performance in the wireless domain. Some researchers have attempted to provide solutions at the data link layer (LL), thereby attempting to hide the "deficiencies" of the wireless channel from TCP. Others have introduced modifications to TCP so that it performs better under the new conditions. Last but not least, there are a number of proposals for new transport protocols, optimized for wireless networks.

LINK-LAYER SOLUTIONS

As noted earlier, the major problem for a reliable transport protocol like TCP arises because of the different nature of the wireless medium. Therefore, it is reasonable to attempt to attack the problem at its root cause. The LL protocol that runs on top of the physical layer has immediate knowledge of dropped frames and thus can respond faster than higher-level protocols. At the same time, the LL protocol has more control over the physical layer protocol. Hence, alleviating the inefficiencies of the wireless medium at the LL provides the transport layer protocol with a dependable communication channel, similar in characteristics to a wired channel. In this way, the transmission media heterogeneity introduced in the network remains transparent to the existing software and hardware infrastructure, and does not necessitate any changes to current TCP implementations

Unfortunately, making a wireless channel look like a wired channel is not an easy task. The LL protocol will have to ensure relatively reliable delivery of packets. This is usually implemented using an automatic repeat request (ARQ) scheme and/or by means of FEC. For example, Mobitex (see Appendix) uses FEC at the physical layer and ARQ at the LL. ARQ works well when the error rates are low. High error rates can lead to a large volume of retransmission and can even cause a complete "black-out" in the connection. On the other hand, FEC is not very well suited for channels that do not have large bandwidth, as is the case for many categories of wireless. FEC can detect and reverse a limited number of bits, but the penalty in bandwidth can be considerable. For example, the raw bit rate for CDPD, which uses FEC, is 19.2 kb/s, while the actual user bit rate is 9–13 kb/s [16]. In addition, the authors in [51] argue that the use of FEC means more power consumption and computation delays per packet. Tabbane [16] also notes that FEC requires fast digital signal processing chips, which do indeed consume additional power.

A question that still remains unresolved is whether the LL protocol should be aware of TCP workings or not. Early studies [14] have shown that a TCP-unaware LL protocol may hide the wireless link errors from TCP, but at the same time lead to worse overall performance. For example, upon the loss of a packet over the wireless link, the LL protocol will retransmit the packet without waiting for a TCP retransmission. However, a TCP-unaware LL protocol will not suppress the duplicate acknowledgements, which can cause a TCP retransmission in addition to the LL retransmission [43]. This duplicated effort leads to worse performance and wastes scarce

resources, *i.e.* wireless bandwidth and battery power. It has been noted that LL retransmission timers must expire faster than TCP's in order to avoid this duplicated effort. Moreover, the LL protocol should try to avoid triggering fast retransmit at the TCP sender, for example by not delivering incoming frames out of order.

TCP-Aware LL Protocols — The most prominent proposal in this category is the snoop protocol [12]. Snoop was designed so that the wired infrastructure of the network would need no changes. A snoop agent must, however, reside at the base station (BS), while the mobile host would need to run the snoop protocol. The BS snoop agent inspects every TCP segment that is sent to or received from the wireless hosts that are roaming in its cell. We will not go into the details of the protocol, but it is useful to present the salient points of its workings.

The snoop agent maintains a cache of unacknowledged data segments destined for the mobile host. When the snoop agent receives a duplicate acknowledgment from the mobile host destined for the fixed host, it retransmits the missing segment and suppresses the duplicate acknowledgment instead of forwarding it to the fixed host. Snoop also uses timeouts to locally retransmit segments, if needed. These timeouts are less coarse than the *de facto* TCP timeout [5], and therefore expire sooner, thus leading to a local retransmission within the time span of a TCP timeout. Snoop utilizes the ability of a LL protocol to respond fast, and at the same time uses the available information to keep TCP "happy" with the existing network connection. However, it should be noted that TCP-level retransmissions do happen, mostly due to timeouts at the sender.

TCP Reno with snoop has been shown to achieve much better performance than both TCP Reno alone and other end-to-end protocols [43]. Moreover, the authors demonstrated that the use of a selective acknowledgements scheme improves performance. Snoop has been tested mostly under scenarios involving an environment composed of a two-hop path with a wireless LAN at the receiver side. The authors also conducted tests in an environment composed of the wired WAN and a wireless LAN: the connection path included a 16-hop route inside a wired WAN, with the 17th hop being the wireless LAN. In this configuration, the proportion of the total RTT that is due to the wireless part of the connection path is smaller than in the two-hop scenario.

One of the key aspects of snoop's superior performance is the ability to respond to losses faster than TCP. However, snoop was designed for small RTTs in the wireless part of the path, so it may not achieve superior performance when these RTTs are large. Indeed, Sinha *et al.* found that TCP NewReno [52] with snoop does not yield better performance in wireless WAN scenarios [31]. As explained in the Appendix, wireless WANs suffer from large RTTs and low data rates, two factors that are important for snoop to work efficiently. Essentially, the proportion of the total connection RTT that is due to the wireless path is very high, so snoop cannot react to losses fast enough to prevent TCP retransmissions: LL and TCP retransmissions interfere, leading to worse performance. Finally, snoop does not perform well when long or frequent disconnections are common [53].

TCP-Unaware LL Protocols — TCP-unaware LL protocols, in principle, help to preserve protocol-layer modularity, and as such can be extended to accommodate transport protocols other than TCP.

TULIP — The Transport Unaware Link Improvement Protocol (TULIP) [51] was designed for half-duplex wireless channels with limited bandwidth. TULIP is not aware of the transport protocol *per se*, but it is aware of the type of service requested (*e.g.*, reliable service for TCP vs. unreliable for UDP). In other words, TULIP requires the network protocol (in this case IP) to indicate if a particular packet requests a reliable packet delivery service or not. Though TULIP is unaware of TCP's workings, it was nevertheless designed with TCP in mind. For example, TULIP attempts to prevent the TCP sender from receiving three duplicate acknowledgements by delivering only in-order incoming frames to IP.

Like snoop, TULIP locally buffers packets and uses an ARQ scheme to attempt to recover from losses on the wireless link, before the TCP sender times out. TULIP's timers rely on a maximum propagation time in the wireless channel. An interesting point is that TULIP does not offer a reliable delivery for TCP acknowledgements. However, the authors do not specify how a TCP-unaware LL protocol is able to differentiate between pure TCP acknowledgements and TCP segments. The developers of TULIP show that, at least over half-duplex radio links, a TCP-uncoupled LL solution is indeed possible and can yield better performance than TCP-aware LL protocols.

Delayed Duplicate Acknowledgments — Delayed duplicate acknowledgements (DDA) [54] is a LL TCP-unaware proposal that attempts to mimic the workings of snoop. The principle is the same as with snoop, *i.e.*, wireless losses are detected at the BS and lost segments are retransmitted locally. The authors specify that each TCP data segment must be encapsulated in a single LL frame, and each TCP ACK should be encapsulated in a single LL ACK. DDA uses different sequence numbers for its frames than TCP does for its segments. The protocol does not attempt to deliver the packets in-order to higher protocols.

A loss is detected at the LL when a LL-duplicate acknowledgement (which encapsulates a TCP ACK) reaches the base station. The BS delays the sending of the duplicate ACKs to the TCP sender by an amount of time *d*. At the same time it retransmits the lost segment locally. Further duplicate ACKs are also delayed. If an ACK arrives indicating that the retransmitted packet has been received, then the duplicate acknowledgments are not sent on to the TCP sender. If the time *d* expires then all duplicate ACKs are released and, probably, the TCP sender goes into fast retransmit. According to the authors, this scheme works well in cases where snoop also works satisfactorily, thus it will not perform well on slow wireless links. The protocol is still in its infancy and a number of issues remain to be resolved, such as the optimal value for *d*, but it seems that under certain conditions it can perform well, almost as well as snoop.

SPLIT CONNECTIONS

Proposals in this category, such as Indirect TCP (I-TCP) [55], came out very early. The basic idea here is that since we have two completely different classes of subnetworks (wired and wireless), we could split each TCP connection into two connections at the point where the two subnetworks meet, *i.e.*, at the base station. The BS keeps one TCP connection with the fixed host, while at the same time it uses another protocol designed for better performance over wireless links for the mobile host. The BS is entitled to acknowledge the segments as soon as it receives them [55]. However, this means that it is possible for the acknowledgement of a particular segment to arrive at the sender before the segment actually reaches the recipient [43]. Obviously this violates TCP semantics.

I-TCP does not handle handoffs efficiently, since the TCP state must be transferred between base stations. According to [56], handoffs may take several hundreds of milliseconds to be completed, thus leading to degraded TCP performance [57]. Moreover, crashes in the base station result in TCP connection termination. I-TCP is also not suitable for cases where the wireless link is not the last part of a connection path, because we could end up splitting a particular connection several times if different combinations of subnetworks are involved, leading to performance degradation.

M-TCP [53] also splits TCP connections but preserves TCP semantics, and does a better job than TCP in handling high BERs, frequent (if short) and prolonged (if not frequent) disconnections due to user roaming, blackouts, etc. However, M-TCP requires a LL protocol to recover from losses in the wireless link. It is interesting to note that Brown and Singh propose another version of M-TCP (Compressed M-TCP) that uses compression to alleviate the problem of limited bandwidth in WWANs: data is compressed at the (wired) sender and decompressed at the (mobile) receiver. Compression can be very effective for certain kinds of data, but typically costs more energy and does not necessarily translate into faster user-perceived response times, especially for devices that have limited processing power.

TCP Modifications

TCP has been undergoing constant modifications and improvements since its earliest days. In this section we discuss, in the context of wired-cum-wireless networks, some of the more recent modifications, though by no means are all of these modifications aimed at the problems of such networks.

TCP SACK — TCP selective acknowledgments options (TCP SACK) [58] were proposed as a means to alleviate TCP's inefficiency in handling multiple drops in a single window of data [15, 59]. The use of selective acknowledgements is optional, and the two communicating parties have to negotiate during the connection setup whether SACK is to be supported.

In contrast with the standard cumulative TCP ACKs, a SACK can convey information to the sender about multiple non-contiguous blocks of successfully transmitted data segments. That is, TCP SACK enables the receiver to inform the sender about segments that were received out of order. Duplicate ACKs indicate the earliest in-order segment that is currently missing, and that other, later segments have been received, but do not specify which ones. Thus, TCP SACK allows the sender to avoid retransmitting segments whose successful delivery at the other end is not evident from the duplicate ACKs that arrive at the sender.

The TCP SACK specification states explicitly that standard congestion control algorithms, including TCP timeouts, are not affected by the proposal. TCP SACK kicks in only when three duplicate ACKs arrive at the sender. At that point, the sender can skip the retransmission of all SACKed segments, and retransmit the first unacknowledged, un-SACKed data segment. Therefore, SACK will not improve performance in cases where the sender window size is not sufficiently large to allow for at least four segments in flight. This problem arises in cases where the bandwidth×delay product is small, or after a number of consecutive segment losses, which cause the congestion window to shrink [44]. After a retransmission timeout occurs, all SACKed segments are considered "un-SACKed," and the sender must retransmit the oldest outstanding segment in the sending window [58].

Although TCP SACK has been shown to achieve better performance than standard TCP under certain scenarios [43, 59], there are a number of cases for which TCP SACK does not offer any significant performance improvement. For example, Balakrishnan *et al*. reported that for certain traces TCP Reno enhanced with SACK avoided only 4 percent of the retransmission timeouts because of the small congestion windows [60]. Particularly for mobile hosts, it has not yet been demonstrated that the SACK-introduced complexity, which implies more power consumption and increased memory requirements, is worth implementing. Even in the case of wired-only networks, different studies have reached differing conclusions, so a consensus has not yet been achieved on the use of SACKs. TCP SACK is not currently part of the TCP specification [1, 3], but seems to be implemented in a number of popular operating systems, including Microsoft Windows 98 [61].

TCP FACK — TCP forward acknowledgment (TCP FACK) [62] attempts to decouple the TCP congestion control algorithm from data recovery. The aim of TCP FACK is to estimate more accurately the amount of transmitted but unacknowledged data in the network, and hence make intelligent decisions about the data that should be (re)transmitted. TCP FACK is designed to complement SACK in achieving superior performance. It introduces a sender variable that keeps track of the "forward-most" data (in the sense of "segment with the highest sequence number") which has arrived at the receiver (*snd.fack*). In addition, the sender must also keep track of the amount of retransmitted data (*retran_data*). Based on these variables and the information stored in standard TCP sender variables, *i.e*. the next sequence number to be sent (*snd.nxt*) [1], the sender is able to accurately estimate the amount of outstanding data in the network (*acwnd*) [62]:

$$acwnd = snd.nxt - snd.fack + retran\_data$$

Using this estimate, TCP FACK can recover from episodes of heavy loss better than TCP Reno with or without SACK. TCP FACK has been shown to perform better than TCP Reno, and TCP Reno with SACK, under certain conditions [62]. However, it was never really tested for wired-cum-wireless environments, and is more or less targeted toward improving TCP's performance when losses are due to congestion rather than random losses.

TCP Santa Cruz — TCP Santa Cruz [63] includes new congestion control and error recovery strategies. It is a protocol that is designed with transmission-media heterogeneity in mind, and is effective at handling asymmetric and/or lossy links, limited bandwidth, and variable delays. For each segment TCP Santa Cruz keeps an entry recording the time the segment was sent by the sender and the time it was received by the receiver. Thus, the sender can calculate the time interval, $S_{j,i}$, between the transmission of segment j and some other, preceding segment i, and the inter-arrival time, $R_{j,i}$, of the corresponding data packets at the receiver. Based on this information, the protocol can deduce whether congestion is building up or if network conditions are improving or remaining steady [63]. TCP Santa Cruz proposes changes in the congestion control, slow start, and retransmission algorithms in order to take advantage of the increased amount of information that is available to the sender. The growth of the congestion window is decoupled from the number of returned ACKs. Moreover, the protocol uses a selective acknowledgements scheme to improve performance for multiple losses in a single window of data.

TCP Santa Cruz performs better than TCP Reno and Vegas under certain simulated scenarios [63], including file transfers over a three-hop path with a bottleneck link, and a

one-hop transfer over an asymmetric link [63, 64]. However, the authors do not address the issue of how to calculate accurately $S_{j,i}$ and $R_{j,i}$ when the receiver uses delayed ACKs. In addition, the relative complexity that is introduced raises power consumption issues. However, these complexities are mostly introduced in the sender, which for most practical purposes could mean limited additional power consumption for mobile hosts.

Changes to Other TCP Mechanics — Many researchers have proposed changes in TCP mechanisms as a means of improving TCP performance. Some of the proposals have been widely accepted and adopted in TCP standards, while others are still under investigation. Additional information can be found in [65, 66].

ACK Generation TCP Strategies — ACKs returning from a receiver play multiple roles in TCP. They are used for reliability purposes (as part of the sliding window algorithm), for increasing the sender's transmission rate, and for congestion control. According to [3], during Slow Start a TCP sender increments its congestion window (cwnd) by at most one sender's maximum segment size (SMSS) for each ACK received that acknowledges new data. For example, a cumulative ACK for two segments and a cumulative ACK for one segment, both acknowledging new data, would cause the same increase in cwnd, possibly one SMSS.

According to the original TCP specification [1], receivers acknowledge every incoming segment. On the other hand, the latest RFC on TCP congestion control [3] states that a TCP receiver "should" [67] use the delayed ACK algorithm. This algorithm [68] gives the receiver the option to delay an ACK if it is for an in-order segment, but must acknowledge every second full-sized segment. Of course, this presupposes a receiver that does not have any data to send back to the sender, and thus does not piggyback ACKs. Thus, if no losses occur the receiver would send a cumulative ACK only every second in-order arriving segment. However, the receiver must send an ACK within 500 ms of the arrival of the first unacknowledged segment. In the event of an out-of-order segment, the receiver must immediately send a duplicate acknowledgement. Most current TCP implementations do indeed implement the delayed ACKs algorithm. Delayed ACKs save network resources and, in the case of a battery-operated wireless host, energy expenditure (though it was not proposed with this aim in mind).

Allman has studied [69] the effect of the delayed ACKs algorithm, as well as other schemes, for wired networks where drops occur only due to congestion. The implications of random losses due to fading wireless channels and the effect of a limited bandwidth/longer RTTs environment have not been studied with respect to the different ACK generation algorithms. Nevertheless, we present Allman's conclusions here because the proposed algorithms may be of some use in the wired-cum-wireless domain.

First, the delayed ACKs algorithm essentially doubles the amount of time a TCP sender spends in Slow Start. This means that an even larger proportion of short flows spend most of the time in Slow Start, without taking full advantage, where it is available, of network bandwidth. Alternative mechanisms studied include acknowledging every segment, Delayed ACKs After Slow Start (DAASS), unlimited byte counting and limited byte counting [69].

Acknowledging every segment is more aggressive than using delayed ACKs, but leads to slightly better performance in terms of throughput [69]. In particular, an improvement of 28 percent has been measured for short flows. On the other hand, acknowledging every segment leads to more segment losses, because the TCP sender is instructed to be more aggressive, which can lead to congestion build-up. Moreover, as was demonstrated in [44], acknowledging every segment leads, at least in a number of scenarios, to worse performance: first, because the sender is encouraged to undertake unsuccessful segment transmissions; and second, because the receiver consumes twice the power and bandwidth for the transmission of ACKs.

DAASS attempts to limit the amount of time spent in Slow Start by employing the ACK-every-segment algorithm while in slow start and using the delayed ACKs algorithm only during congestion avoidance. DAASS also improves TCP throughput, and achieves exactly the same results for short flows as does the acknowledging of every segment mentioned above, since in this case the algorithms are the same. DAASS causes less congestion build-up than acknowledging every segment, but more than the standard TCP delayed ACKs algorithm [69].

Unlimited byte counting (UBC) uses the number of bytes acknowledged as the metric for the expansion of cwnd: cwnd is increased by as many bytes as are acknowledged. This decouples the window expansion from the acknowledgment generation algorithm at the receiver. However, it proves to be too aggressive and may lead to worse performance. On the other hand, limited byte counting (LBC), where the total increase in cwnd is limited to three SMSS, seems to improve the performance, but not as much as acknowledging every segment and DAASS.

As a last word, it is worth noting that other researchers have proposed to lower the TCP retransmission threshold from three duplicate acknowledgments to two. This proposal stems from the fact that for a wide category of networks the bandwidth × delay product is not sufficient to allow for at least four SMSS segments to be unacknowledged. Consequently, the number of outstanding segments does not permit Fast Retransmit to kick in, and losses are detected only by means of timeouts. For example, Lin and Kung reported that 85 percent of the timeouts found in a set of Internet traces were not preceded by a Fast Retransmit [70]. Lowering the retransmission threshold may solve this problem. On the other hand, lowering the retransmission threshold does not necessarily lead to better performance under all circumstances [44].

ACK Pacing — The standard TCP ACK generation mechanisms produce bursty traffic. For example, if the sender is in Slow Start it will send two new segments for each ACK received. Bursty traffic can incur packet losses and lower throughput in the context of a rather unfairer environment. Many researchers, including [71, 72], have proposed to "pace" the transmission of segments by sending them in a spaced-out fashion during the entire RTT. Note that pacing is a rate-based scheme in the sense that the TCP sender sends segments at a predetermined rate, not according to the flow of incoming ACKs. Theoretically, pacing has many merits, including the potential for smaller queuing delays, thus leading to better performance.

However, Aggarwal *et al.* [73] use extensive simulations to show that although pacing can improve fairness and throughput in many cases, it can also worsen performance significantly in many others. Pacing can cause global synchronization, which is known to lead to poor link utilization [74]. As a last word, note that pacing is mainly aimed at dealing with congestion-related losses and, in view of [73], it may not be suitable for wired-cum-wireless environments.

Increase TCP's Initial Congestion Window – Allman *et al.* [75] proposed the increase of the initial congestion window

(IW) to more than one SMSS. Simulation studies have shown that this modification can lead to better performance (up to 25 percent) [75], especially for short flows. Concerns were raised, however, regarding the increased possibility of dropped segments, in particular at congested routers or links. Shepard and Partridge [76] studied the case where a TCP receiver is connected to the Internet over a 9.6 kb/s link, through a router with enough buffer space to accommodate only three segments, each containing 1024 bytes. When the sender uses an IW of four segments it is guaranteed that it will experience a drop in the initial phase of the connection. The study concluded that the performance achieved when IW is four segments is no worse than when it is only one. Nevertheless, the latest RFC defining TCP's congestion control [3] specifies that "the initial value of `cwnd`, MUST be less than or equal to 2×SMSS bytes and MUST NOT be more than two segments." The members of the IETF TCP Implementation Working Group opted for a gradual increase in IW since empirical data from actual networks on the effect of increasing IW were not available at the time.

Increasing the initial congestion window can be advantageous for short flows and connections that exhibit large propagation delays. Note that these modifications are intended only for the initial congestion window: the loss window, *i.e.* the window used after a loss has been detected, and the restart window, which is used when the sender restarts the connection after an idle period, are not affected. The specified value for these windows remains one SMSS.

Explicit Congestion Notification — As pointed out earlier, TCP Congestion Control is mainly governed by the detection of lost or dropped segments. When a sender detects segment losses, it assumes that this is due to congestion and that it should therefore lower its sending rate. This approach is based on the fact that until recently routers did not provide feedback to the sender, and used a rather simple queuing strategy, namely drop tail: when a router runs out of buffer space, the latest incoming segments are dropped. TCP with Explicit Congestion Notification (ECN) calls for more functionality from the routers [77, 78]. The routers should inform the TCP sender of incipient congestion, signaling that the sending rate should be lowered. In this way, the sender is informed on time about congestion buildup and segment drops can be avoided [79].

By explicitly signaling out when congestion is building up, ECN can also be used for wired-cum-wireless environments. If the entire internetworking infrastructure were capable of conveying ECNs, a mobile host could reasonably infer that drops in the absence of ECNs must be due to random losses, and hence that congestion control algorithms should not be employed, but rather that the current sending rate should be sustained. ECN is also power conserving, because the routers do not drop packets to indicate congestion, so only a minimal number of segments are lost [79], except, of course, the ones lost in the wireless path. ECN can also lead to improved overall TCP performance by avoiding retransmissions and timeouts [77–80].

Explicit Loss Notification — In contrast to ECN, Explicit Loss Notification (ELN) was proposed for wireless networks [43]. Instead of implicitly deducing when a drop is due to random losses, it is preferable to be informed explicitly of a loss resulting from errors in a wireless link. If it were possible to explicitly indicate when a particular loss is due to errors induced by the wireless link, then TCP could be modified so as to refrain from going into congestion avoidance. However, such a scheme is very difficult to implement and, to our knowledge, no efficient solution has appeared in the literature.

Fast Retransmits — Caceres and Iftode [57] discuss TCP's performance degradation when handoffs occur as a mobile user roams, *e.g.*, inside a building. The authors have experimented with microcellular networks, *i.e.* wireless LANs with cells of a few meters in diameter. These networks offer good raw bit rate (at the time the article was published, 2 Mb/s for IEEE 802.11; currently, with IEEE 802.11b, it can be up to 11 Mb/s [81]). While bandwidth is not the main issue in these networks, handoffs can have a big impact on TCP throughput, and therefore on the mobile user's experience.

The authors explore situations where the cells are overlapping, adjacent, or otherwise. TCP throughput suffers in all these cases. Even in the case of overlapping cells the throughput is less, though by a mere six percent. The case where the handoff needs one second to complete leads to a period of 2.8 seconds of complete loss of communication at the transport level.

During this period, TCP loses segments in the forward path and acknowledgments in the return path. The absence of acknowledgments means that Fast Retransmit will not kick in, and so the TCP sender must wait for a timeout. The authors propose changes to Mobile IP at the mobile host so that Mobile IP informs TCP when a handoff is completed. Then, the receiver's TCP sends duplicate acknowledgements to initiate Fast Retransmit at the sender TCP. Thus, the sender does not have to wait for a timeout and takes full advantage of Mobile IP's knowledge about handoffs. However, the Fast Retransmits approach aims only at dealing with handoffs and does not accommodate losses due to wireless channel fading.

NEW TRANSPORT PROTOCOLS

As mentioned earlier, new transport protocols can be designed to perform better than TCP in purely wireless networks. However, these protocols are in their infancy and have not yet been tested on a wide scale.

Wireless Transmission Control Protocol — The Wireless Transmission Control Protocol (WTCP) [31] is designed to provide a reliable transport protocol for CDPD [16, 26], and in general for WWANs that exhibit low bandwidth and high latencies. WTCP is used when a mobile host needs to connect through a proxy to access information. Sinha *et al.* [31] note that this is the case for most current deployments of WWANs. Hence, WTCP is a proposal that attempts to solve the problem of transmission-media heterogeneity in a "split connections" fashion.

WTCP uses algorithms similar to those of standard TCP for connection management and flow control. However, it follows a different approach with regard to congestion control. First, WTCP is rate-based with the rate control done at the receiver. That is, the receiver is responsible for setting the appropriate rate for the sender to use. The receiver, upon the arrival of each incoming segment, employs an algorithm [31] to determine whether to ask the sender to increase, decrease, or maintain the current sending rate. This information is conveyed back to the sender through cumulative ACKs. Transmission control in WTCP is governed by the inter-packet delay as measured at the receiver.

Second, WTCP attempts to predict when a segment loss is due to transmission errors or to congestion. In short, while the network is deemed to be uncongested, the receiver keeps statistics for non-congestion-related segment losses. Based on this information, the receiver signals the sender to continue

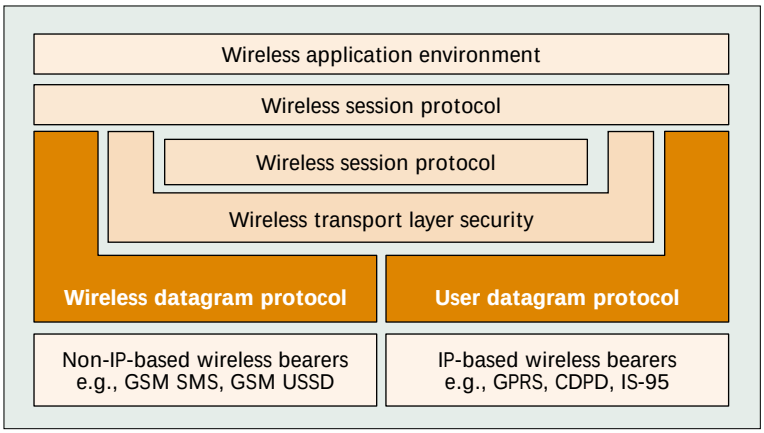

FIGURE 4. *The WAP protocol stack [23].*

transmitting with the same rate if the loss is estimated to be due to transmission errors, or to decrease the sending rate if congestion is deemed to be the cause of these losses. This mechanism, though promising, has not yet been exhaustively tested.

In contrast with TCP, WTCP does not use an ARQ scheme to assure reliability of transfers. The authors believe that one of the main reasons for the sub-optimal performance of TCP in WWANs is inaccurate retransmission timeout estimation. In order to alleviate this inefficiency WTCP employs a scheme with SACKs and probes. The receiver-generated acknowledgements act both as cumulative ACKs and as SACKs. Thus the sender can verify whether certain segments were received and retransmit the missing ones. However, ACKs are not always generated in response to incoming segments. In fact, the sender must specify to the receiver how often to send an ACK. The receiver can also send a SACK in the event of out-of-order segments. If the sender does not receive an ACK during the specified period, it goes into blackout mode [31]. If the sender does not have any more segments to send, and at the same time it has not received a confirmation that all outstanding segments were received, then it will use probe segments to elicit receiver ACKs.

WTCP has been shown [31] to perform better than TCP NewReno [52] and TCP Vegas [82, 83], at least under certain scenarios. Be that as it may, the receiver in WTCP is considerably more complicated than in TCP; this could lead to increased power consumption, since usually the mobile host plays the role of the receiver.

Wave-and-Wait Protocol and TCP-Probing — The Wave-and-Wait Protocol (WWP) [47] and TCP-Probing [48] share some similarities with WTCP, discussed earlier. Like WTCP, WWP has the receiver monitor incoming segments and set an appropriate transmission rate for the sender to use. Both WWP and TCP-Probing use a structured exchange of short, energy-efficient probing segments between sender and receiver to ascertain, in the event of lost segments, whether the loss is due to transient (random or transmission) errors or more persistent conditions (congestion, fading, blackouts, etc.). TCP-Probing then attempts to adjust the sender congestion control mechanism accordingly. WWP's congestion control mechanism, on the other hand, consists of setting an appropriate "wave" level, where a wave is a group of data segments that are transmitted back-to-back.

WIRELESS APPLICATION PROTOCOL

The Wireless Application Protocol (WAP) is an attempt to define an industry-wide specification for developing applications that operate over wireless networks [50]. The WAP Forum [84] decided not to define a specification aimed at a particular set of actual devices, but rather to create a protocol stack and application environment that can allow a broader class of devices to communicate efficiently over wireless networks.

WAP has appeared in the popular press as the "wireless Web." However, this is not strictly correct, not only because other "wireless Web" solutions have been proposed (*e.g.*, NTT DoCoMo's i-mode [85, 86]), but also because it goes against the very philosophy of WAP, at least in its current incarnation. Mann [30] notes that the "WAP user paradigm is one where users are able to make small, specific, focused requests for small chunks (typically less than 1000 bytes), not browse random Web sites."

Although first-generation WAP is more or less targeted at "smart" mobile phones, it does not exclude PDAs, two-way pagers, and other devices. In fact, even more powerful devices, such as laptop computers, can use WAP. WAP assumes that the mobile device has limited processing power and memory, and is connected to a network with rather limited data rates (typically less than 10 kb/s) and high latencies. The network is assumed to be unreliable [30]. Moreover, WAP was designed with power consumption in mind: the target devices are assumed to have limited battery lifetime, so every effort must be made to conserve energy. Other device characteristics are small screen size and limited data entry capabilities.

WAP introduces a new protocol stack (Fig. 4). The WAP Application Environment (WAE) includes a microbrowser specification, the wireless markup language (WML), a scripting language (WMLScript), and a framework for wireless telephony applications [30].

WSP, the Wireless Session Protocol, provides functionality similar to HTTP/1.1 [42], including long-lived sessions. WSP supports reliable connection-oriented session services, which can be suspended and resumed at a later time [50], over WTP (Wireless Transaction Protocol). Note that resuming a WSP session is less costly in terms of signal exchanges for the mobile client than establishing a new one. WSP's ability to resume sessions saves scarce network resources and preserves battery power. WSP also supports unreliable connectionless sessions over WTP or WDP (Wireless Datagram Protocol). Last but not least, WSP allows for capability and content negotiation, asynchronous content push to the client, and multiple asynchronous transactions [50].

WTP adds reliability on top of datagram services supplied by either WDP [84] or UDP [87], offering three classes of service:

- Unreliable one-way requests
- Reliable one-way requests
- Reliable request-reply transactions [23]

It has no explicit connection setup or teardown, and uses unique identifier numbers (similar to segment sequence numbers in TCP), acknowledgements, duplicate removal, and retransmissions to ensure reliable transactions. WTP allows the user to optionally confirm the receipt of every message. Finally, WTP is message-oriented, that is, the basic unit of interchange is an entire message, not a byte stream as in TCP [50]). WTP offers no security mechanisms; optional security, as well as data compression and encryption, is provided by the Wireless Transport Layer Security (WTLS) protocol [84].

WDP is the transport layer protocol of the WAP architecture, providing unreliable datagram service to the higher layers. It aims to give a consistent network service to the higher layers, independent of the data-capable bearer services of the different underlying network types (Fig. 4). WDP must pro-

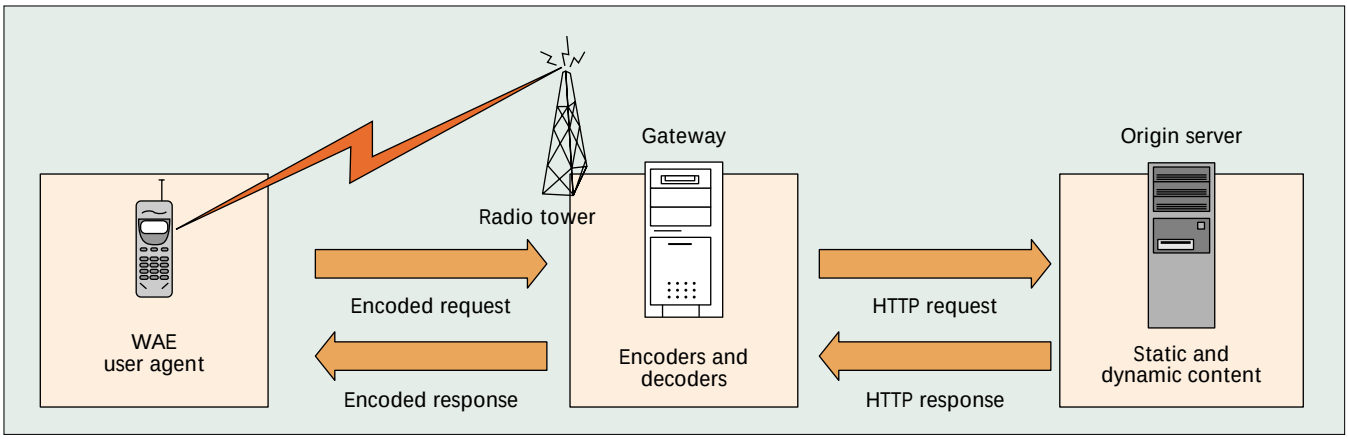

FIGURE 5. *The WAP programming model [50].*

vide a multiplexing/demultiplexing functionality, and basic error detection. In addition, it must handle segmentation and reassembly. In fact, if the bearer network supports IP (*e.g.*, CDPD, GPRS), then UDP is used instead of WDP, because IP handles that. If the underlying network does not use IP for routing, then an adaptation layer is needed in conjunction with WDP [23].

Clearly, WAP follows an approach wherein transfer reliability is added on top of an unreliable datagram service. Hence, much of TCP's functionality is positioned at the presentation and application layers. This choice imposes certain limitations, but using lightweight transport protocols such as UDP and WDP has its merits, *e.g.*, it allows the application developer to choose the exact level of reliability needed by the application.

Another key aspect of WAP is that all connections between a mobile host and an origin server pass through a proxy, the WAP gateway (Fig. 5). The proxy uses the WAP protocol stack to communicate with the wireless device, and standard HTPP/TCP/IP to communicate with the origin server. Thus, the WAP protocol stack is essentially a "split connections" proposal.

A typical scenario involves a user issuing a request on his device. The device will use the WAP protocol stack to communicate with the proxy. The proxy receives the request, translates it into an HTTP request, and uses a standard TCP/IP wired network to connect with the origin server. The WAP gateway is responsible for encoding the responses of the origin server into a compact binary format and conveying it to the mobile device using the WAP protocols. WAP uses binary formats instead of, for example, the standard text formats for HTTP, in order to minimize the amount of data that must be transmitted and limit the processing that needs to be done at the mobile host (see Compressed M-TCP in an earlier section). In this way less power is expended for each request. Furthermore, a WAP gateway typically supports a DNS service in order to accelerate name lookups, and can use caching and distillation to enhance performance [23].

WAP is intended to be as compatible as possible with the existing Internet, given the device limitations mentioned above. Potentially, one could foresee a merging of the WAP stack with TCP/IP. We shall not pursue the details of WAP any further; the interested reader can find a wealth of information in [23, 30, 50] and at the WAP Forum web site [84].

Wireless network	Bandwidth (kb/s)
Motient (formerly ARDIS)	4.8–19.2
Mobitex (by Cingular)	8
CDPD	19.2
Ricochet (by Metricom)	28.8–128
GPRS	Up to 170

Table 1. *Data transer rates provided by wide area wireless data networks [11, 22, 28, 88, 91].*

CONCLUSION

TCP has been shown to perform poorly in wireless environments in terms of achieved throughput due to the factors examined earlier. The consensus of the research community is that the use of "traditional" TCP in wireless environments is not a very attractive choice. While TCP is needed for a wired-cum-wireless Internet, the current version(s) is not adequate for the task. Improving TCP performance under these conditions has been a very active area of research over the last several years. However, researchers have more or less focused on specific, *ad hoc* problems, and there appears to be no general solution so far. Solutions that work extremely well under some conditions perform poorly when these conditions cease to exist. For example, solutions proposed for wireless LANs do not perform well in wireless WANs, and vice versa. Moreover, most of the work in this area used throughput as the main criterion for judging the merit of a proposal. To pass muster, new proposals will have to demonstrate their improved performance not only in terms of throughput but also in terms of power consumption.

ACKNOWLEDGMENTS

The author gratefully acknowledges the invaluable assistance of Professor Hussein Badr in the preparation of this article.

APPENDIX: WIRELESS DATA NETWORKS

Earlier we introduced the various wireless networks and classified them according to coverage area into wireless LANs and WANs. In addition, computer networks, including wireless, can be classified into packet switching and circuit switching networks [6, 16]. This Appendix presents yet another categorization, which is based on the kind of services offered and the user mobility supported.

Full User Mobility — Wide Area Wireless Data Networks (WAWDNs) include systems such as the Cellular Digital Packet Data (CDPD) [26], Mobitex [16, 88], Advanced Radio Data Information System (ARDIS) [89], and the General Packet Radio Service (GPRS) [16]. These packet-switched networks allow the user to roam almost everywhere. For

example, special attention has been given to the ARDIS network[1] so that coverage includes not only open spaces but also building interiors [90]. Moreover, an ARDIS base station can cover an area of 15–25 km in diameter. The user is able to move with relatively high speeds and remain connected, though at rather slow data throughput rates (Table 1).

CDPD was designed to use channels of the Advanced Mobile Phone Service (AMPS), the first generation analog cellular network that do not carry voice traffic. These channels are usually shared with AMPS, *i.e.*, used only when there are no phone calls. However, the network operator can specifically assign channels only for data traffic. As CDPD gains popularity — and AMPS becomes obsolete — more CDPD-dedicated channels can be allocated. CDPD can be deployed at less cost than other wireless wide area networks because it utilizes the existing AMPS hardware and software infrastructure. CDPD is based on a CSMA/CD variant called digital sense multiple access [16] and offers IP-based services [26], which is a great advantage.

Similar to CDPD, GPRS is normally embedded in a GSM network [16]. GPRS uses packet switching and is provisioned in GSM Phase 2+. Its main objective is to provide standard data transfer technologies like TCP/IP with a mobile radio network with significantly higher bit rates than the other systems, in contrast to which it was also designed with office applications in mind. The standard was finalized by ETSI in late 1997, but to many it is a transitional technology towards third-generation cellular networks [16, 90]. GPRS is currently being deployed in many European countries. The reader can find more information about WAWDNs in [16, 23–26, 90]. In addition, [11] provides a rich glossary of terms used in the wireless industry.

Cellular networks, like the Global System for Mobile communications (GSM) [25] and IS-95 [22] (commonly referred to as CDMA in the US), can also be used for wireless network connectivity, though in circuit switching fashion. For example, GSM offers transmission rates of up to 9.6 kb/s (14.4 kb/s in GSM Phase 2+). The designers of GSM chose to offer a reliable link layer protocol for data transmission called the *nontransparent mode*. Thus, packets are shielded from corruption using FEC, and an ARQ scheme assures that lost or corrupted packets will be delivered to the receiver [24]. However, as was noted earlier, a reliable TCP-unaware link layer protocol could interfere with TCP and yield worse performance overall. Table 2 presents the available data transfer bit rates for current cellular networks. The forthcoming Enhanced Data GSM Environment (EDGE) will provide speeds up to 560 kb/s, and constitutes yet another transitional step towards the Universal Mobile Telecommunications System (UMTS) [27].

Cellular network	Bandwidth (kb/s)
IS-95 (CDMA)	9.6–14.4
3G CDMA	Up to 2400
GSM	9.6
GSM Phase 2+	14.4, up to 384
EDGE	Up to 560
UMTS	Up to 2048 (microcell)

Table 2. *Data transfer rates provided by cellular networks [11, 22, 28, 91, 94].*

The IS-95 (also known as cdmaOne [11]) cellular network standard uses Direct Spectrum Code Division Multiple Access (DS-CDMA) and was designed to replace AMPS. According to Qualcomm, CDMA offers a ten-fold to fifteen-fold efficiency increase in comparison with AMPS [24]. IS-95 achieves data transmission rates up to 14.4 kb/s. Data services can be used simultaneously with voice traffic. Moreover, due to the advanced power control needed for CDMA [24], a mobile terminal never uses more transmission power than needed. Practical tests have shown that a mobile terminal transmits with less than 1 mW 98 percent of the time. Finally, due to the nature of CDMA, Rayleigh fading and multipath propagation do not degrade the performance significantly [24].

Third-generation cellular networks, like UMTS [27, 28] and 3G CDMA [22], promise much higher data transfer rates. However, these networks have yet to be realized on a large scale. As far as TCP is concerned, higher transfer rates can alleviate some of the inefficiencies discussed earlier. Table 3 presents the profiles defined in the International Mobile Telephone Standard 2000 (IMT-2000). Note that the maximum data rate of 2 Mb/s is available only to slow-moving terminals relatively close to base stations. The "medium multimedia" data rate is available to moving terminals in urban and suburban areas; terminals in rural areas are provided with lower data rates (up to 144 kb/s). Under these restrictions, UMTS will not be able to deliver the high rates necessary for certain applications.

Portable Wireless Data — Wireless LANs offer portability, and even mobility, but in a rather limited area. They offer higher transfer rates than the previous category of wireless networks. Moreover, next-generation wireless LANs will provide even more bandwidth to the user (Table 4).

There has been considerable effort made to create wireless ATM networks [17, 93]. Wireless ATM can provide a reliable network, solving many of the problems described earlier. However, it has yet to reach a final status; there are many issues that still need to be resolved before it can be widely implemented [93].

Ricochet, a service of Metricom [94], offers an alternative solution, providing speeds up to 128 kb/s (Table 1). Users can access the Internet while they are in the coverage area, using a laptop or palmtop computer and a small wireless modem. However, Ricochet does not offer full mobility or handoff handling during online sessions. In other words, if a user tries to access the Internet while in a moving car, the connection may drop.

Service	Bandwidth (kb/s)	Transmission mode
High interactive multimedia	128	Line-switched
High multimedia	2048	Packet-switched
Medium multimedia	384	Packet-switched
Switched data	14.4	Line-switched
Simple messaging	14.4	Packet-switched
Voice	16	Line-switched

Table 3. *Service profiles defined in IMT-2000 [92].*

Fixed Wireless Data (Broadband Wireless) — The term fixed wireless data is used for wireless point-to-point networks that offer services to a location, such as an office or home, through larger customer-premises antennas than seen in the mobile or portable setups. Fixed wireless can

[1] *ARDIS was created in 1990 when Motorola and IBM merged their bi-directional wide area data networks. In 1995 Motorola took over 100 percent of the company. Motient acquired ARDIS in March 1998 [89].*

	Type	Bandwidth (Mb/s)	Distance (m)
HIPERLAN 1	Portable/mobile LAN	20	50–100
HIPERLAN 2	Portable/mobile ATM	24	50–100
HIPERLAN 3	Stationary local loop (ATM)	48	5000
HIPERLAN 4	Point-to-point stationary	155	50–500

Table 4. *ETSI broadband radio access network standards.*

offer faster data throughputs than the preceding categories, equivalent to T1 line speeds. Examples include the Broadband Radio Access Networks [95] and the IEEE 802.16 Broadband Wireless Access standards [96].

REFERENCES

[1] J. B. Postel, "Transmission Control Protocol," RFC 793, Sept. 1981.
[2] V. Jacobson, R. Braden, and D. Borman, "TCP Extensions for High Performance," RFC 1323, May 1992.
[3] M. Allman, V. Paxson, and W. R. Stevens, "TCP Congestion Control," RFC 2581, Apr. 1999.
[4] W. R. Stevens, *TCP/IP Illustrated, Volume 1: The Protocols*, Addison-Wesley, 1994.
[5] G. R. Wright and W. R. Stevens, *TCP/IP Illustrated, Volume 2: The Implementation*, Addison-Wesley, 1995.
[6] L. L. Peterson and B. S. Davie, *Computer Networks, 2nd Edition*, Morgan-Kaufmann, 2000.
[7] H. Saltzer, D. P. Reed, and D. Clark, "End-to-end Arguments in System Design," *ACM Trans. Computing Systems,* vol. 2, no. 4, 1984.
[8] Cooperative Association for Internet Data Analysis (www.caida.org), Apr. 2001.
[9] K. Thompson, G. J. Miller, and R. Wilder, "Wide-area Internet Traffic Patterns and Characteristics," *IEEE Network*, vol. 11, no. 6, Nov. 1997.
[10] Microsoft WebTV Networks, Inc. (www.webtv.com), Apr. 2001.
[11] Wireless Week (www.wirelessweek.com), Apr. 2001.
[12] H. Balakrishnan *et al.*, "Improving TCP/IP Performance Over Wireless Networks," *Proc. ACM MobiCom*, Nov. 1995.
[13] R. Caceres and L. Iftode, "The Effects of Mobility on Reliable Transport Protocols," *Proc. 14th Int'l. Conf. Distributed Computing Systems*, June 1994, pp. 12–20.
[14] A. DeSimone, M. C. Chuah, and O. C. Yue, "Throughput Performance of Transport-Layer Protocols over Wireless LANs," *Proc. GLOBECOM '93*, Dec. 1993.
[15] V. Tsaoussidis *et al.*, "Energy/Throughput Tradeoffs of TCP Error Control Strategies," *Proc. 5th IEEE Symp. Computers and Communications*, France, July 2000.
[16] S. Tabbane, *Handbook of Mobile Radio Networks*, Artech House Mobile Communications Library, Norwood, MA, 2000.
[17] European Telecommunications Standard Institute, ETSI HIPERLAN/1 standard, http://www.etsi.org/technicalactiv/hiperlan1.htm, Apr. 2001.
[18] A. Chandra, V. Gummalla, and J. O. Limb, "Wireless Medium Access Control Protocols," *IEEE Surveys and Tutorials*, vol. 3, no. 2, 2nd Quarter 2000.
[19] IEEE P802.11, The Working Group for Wireless Local Area Networks, http://grouper.ieee.org/groups/802/11/, Apr. 2001.
[20] K. Pahlavan *et al.*, "Handoff in Hybrid Mobile Data Networks," *IEEE Pers. Commun. Mag.*, vol.7, no.2, Apr. 2000.
[21] Lucent Technologies, ORiNOCO RG-1000 Residential Gateway product datasheet, ftp://ftp.orinocowireless.com/pub/docs/ORINOCO/BROCHURES/RG_1000.pdf, Apr. 2001.
[22] CDMA Development Group (www.cdg.org), Apr. 2001.
[23] Y. Lin and I. Chlamtac, *Wireless and Mobile Network Architectures*, John Wiley & Sons, 2001.
[24] R. Bekkers and J. Smits, *Mobile Telecommunications: Standards, Regulation, and Applications*, Artech House Publishers, 1999.
[25] J. Tisal, *GSM Cellular Radio Telephony*, John Wiley & Sons, West Sussex, England, 1998.
[26] J. Agosta and T. Russel, *CDPD: Cellular Digital Packet Data Standards and Technology*, McGraw-Hill, New York, 1996.
[27] UMTS Forum (www.umts-forum.org), Apr. 2001.
[28] Third Generation Partnership Project (www.3gpp.org), Apr. 2001.
[29] W. C. Y. Lee, *Mobile Communications Design Fundamentals, 2nd Edition*, John Wiley and Sons, 1993.
[30] S. Mann, *Programming Applications with the Wireless Application Protocol: The Complete Developer's Guide*, John Wiley & Sons, 1999.
[31] P. Sinha *et al.*, "WTCP: A Reliable Transport Protocol for Wireless Wide-Area Networks," *Proc. ACM MOBICOM '99*, Seattle, Washington, Aug. 1999.
[32] T. V. Lakshman and U. Madhow, "The performance of TCP/IP for networks with high delay×bandwidth products and random loss," *IEEE/ACM Trans. Net.*, vol. 5, no. 3, June 1997.
[33] Rapidly Deployable Radio Network, Field Tests, http://www.ittc.ukans.edu/RDRN/radio/field_test/10151999/, Apr. 2001.
[34] C. Perkins, *Mobile IP Design Principles and Practices*, Addison-Wesley, 1998.
[35] IP Routing for Wireless/Mobile Hosts (Mobile IP), http://www.ietf.org/html.charters/mobileip-charter.html, Apr. 2001.
[36] C. Perkins, "IP Mobility Support," RFC 2002, Oct. 1996.
[37] N. Cardwell, S. Savage, and T. Anderson, "*Modeling the Performance of Short TCP Connections*," Technical Report, Computer Science Department, Washington University, Nov. 1998.
[38] S. Savage, N. Cardwell, and T. Anderson, "The Case for Informed Transport Protocols," *Proc. 7th Wksp. Hot Topics in Operating Systems*, Rio Rico, Arizona, Mar. 1999.
[39] T. Berners-Lee, R. Fielding, and H. Frystyk, "Hypertext Transfer Protocol - HTTP/1.0," RFC 1945, May 1996.
[40] S. E. Spero, Analysis of HTTP Performance problems, http://www.w3.org/Protocols/HTTP/1.0/HTTPPerformance.html, July 1994.
[41] J. Touch, J. Heidemann, and K. Obraczka, "Analysis of HTTP Performance," ISI Research Report ISI/RR-98-463, Aug. 1998.
[42] R. Fielding *et al.*, "Hypertext Transfer Protocol - HTTP 1.1," RFC 2616, June 1999.
[43] H. Balakrishnan, *et al*., "A Comparison of Mechanisms for Improving TCP Performance over Wireless Links," IEEE/ACM Trans. Net., Dec. 1997.
[44] K. Pentikousis, "Error Modeling for TCP Performance Evaluation," Master's Thesis, SUNY at Stony Brook, May 2000.
[45] A. Chockalingam, M. Zorzi, and V. Tralli, "Wireless TCP Performance with Link Layer FEC/ARQ," Proc. IEEE ICC'99, June 1999.
[46] P. Havinga and G. Smit, "Energy-efficient Wireless Networking for Multimedia Applications," *Wireless Communications and Mobile Computing*, Wiley, 2000.
[47] V. Tsaoussidis, H. Badr, and R. Verma, "Wave and Wait Protocol (WWP): Low Energy, High Throughput for Mobile IP-Devices," *Proc. 8th IEEE Conf. Networks*, Sept. 2000.
[48] V. Tsaoussidis and H. Badr, "TCP-Probing: Toward an Error Control Schema with Energy and Throughput Performance Gains," *Proc. 8th IEEE Conf. Network Protocols*, Japan, Nov. 2000.
[49] M. Zorzi and R. Rao, "Is TCP Energy Efficient?," *Proc. MoMUC '99*, San Diego, California, 1999.
[50] Wireless Application Protocol Forum Ltd., *Official Wireless Application Protocol: The Complete Standard with Searchable CD-ROM*, John Wiley & Sons, 1999.
[51] C. Parsa and J. J. Garcia-Luna-Aceves, "Improving TCP Performance over Wireless Networks at the Link Layer," *Mobile Networks and Applications*, 1999.
[52] S. Floyd and T. Henderson, "The NewReno Modification to TCP's Fast Recovery Algorithm," RFC 2582, Apr. 1999.
[53] K. Brown and S. Singh, "M-TCP: TCP for Mobile Cellular Networks," *Computer Communication Review*, vol. 27, no. 5, Oct. 1997.
[54] N. H. Vaidya *et al.*, "Delayed Duplicate Acknowledgements: A

TCP-unaware Approach to Improve Performance of TCP over Wireless," Technical Report 99-003, Computer Science Department, Texas A&M University, Feb. 1999.
[55] A. Bakre and B. R. Badrinath, "I-TCP: Indirect TCP for Mobile Hosts," *Proc. 15th Int'l. Conf. Distributed Computing Systems (IDCS)*, May 1995.
[56] A. Bakre and B. R. Badrinath, "Handoff and system support for Indirect TCP/IP," *Proc. 2nd USENIX Symp. Mobile and Location-Independent Computing*, Apr. 1995.
[57] R. Caceres and L. Iftode, "Improving the Performance of Reliable Transport Protocols in Mobile Computing Environments," *IEEE JSAC*, vol. 13, no. 5, June 1995.
[58] M. Mathis *et al.*, TCP Selective Acknowledgment Options, RFC 2018, Apr. 1996.
[59] K. Fall and S. Floyd, "Simulation-based Comparisons of Tahoe, Reno, and SACK TCP," *ACM Computer Communication Review*, vol. 26, no. 3, July 1996.
[60] H. Balakrishnan *et al.*, "TCP Behavior of A Busy Internet Server: Analysis and improvements," *Proc. IEEE INFOCOM 1998*, Mar. 1998.
[61] Pittsburgh Supercomputing Center, Experimental TCP Selective Acknowledgment Implementations, http://www.psc.edu/networking/all_sack.html, Apr. 2001.
[62] M. Mathis and J. Mahdavi, "Forward Acknowledgement: Refining TCP Congestion Control," *Proc. ACM SIGCOMM*, Aug. 1996.
[63] C. Parsa and J.J. Garcia-Luna-Aceves, "Improving TCP Congestion Control over Internets with Heterogeneous Transmission Media," *Proc. IEEE ICNP '99*, Toronto, Oct. 1999.
[64] T. V. Lakshman, U. Madhow, and B. Suter, "Window-based Error Recovery and Flow Control with A Slow Acknowledgement Channel: A Study of TCP/IP Performance," *Proc. IEEE INFOCOM 1997*, Apr. 1997.
[65] G. Montenegro *et al.*, Long Thin Networks, RFC 2757, Jan. 2000.
[66] Performance Implications of Link Characteristics (PILC), http://www.ietf.org/html.charters/pilc-charter.html, April 2001.
[67] S. Bradner, "Key Words for Use in RFCs to Indicate Requirement Levels," BCP 14, RFC 2119, Mar. 1997.
[68] R. Braden, "Requirements for Internet Hosts - Communication Levels," STD 3, RFC 1122, Oct. 1989.
[69] M. Allman, "On the Generation and Use of TCP Acknowledgements," *ACM Computer Communications Review*, vol. 28, no. 5, Oct. 1998.
[70] D. Lin and H. Hung, "TCP Fast Recovery Strategies: Analysis and Improvements," *Proc. IEEE INFOCOM 1998*, Apr. 1998.
[71] J. Hoe, "Startup Dynamics of TCP's Congestion Control and Avoidance Schemes," Master's Thesis, MIT, 1995.
[72] M. Mathis *et al.*, The Rate-Halving Algorithm for TCP Congestion Control, http://www.psc.edu/networking/rate_halving.html, Apr. 2001.
[73] A. Aggarwal, S. Savage, and T. Anderson, "Understanding the Performance of TCP Pacing," *Proc. IEEE INFOCOM 2000*, Tel-Aviv, Israel, Mar. 2000.
[74] B. Braden *et al.*, Recommendations on Queue Management and Congestion Avoidance, RFC 2309, Apr. 1998.
[75] M. Allman, S. Floyd, and C. Partridge, "Increasing TCP's Initial Window," RFC 2414, Sept. 1998.
[76] T. Shepard and C. Partridge, "When TCP Starts Up With Four Packets into Only Three Buffers," RFC 2415, Sept. 1998.
[77] S. Floyd, "TCP and Explicit Congestion Notification," *ACM Computer Communication Review*, vol. 24, no. 5, Oct. 1994.
[78] K. K. Ramakrishnan, S. Floyd, and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP," Internet Draft, http://www.ietf.org/internet-drafts/draft-ietf-tsvwg-ecn-03.txt, March 2001.
[79] K. Pentikousis, H. Badr, and B. Kharmah, "TCP with ECN: Performance Gains for Large TCP transfers," SBCS-TR-2000/01, Department of Computer Science, SUNY at Stony Brook, Mar. 2001.
[80] J. H. Slim and U. Ahmed, "Performance Evaluation of Explicit Congestion Notification (ECN) in IP Networks," RFC 2884, July 2000.
[81] M. Avery, "Putting 802.11b to the Test," *Network World Fusion*, http://www.nwfusion.com/reviews/2001/0205rev.html, Feb. 2001.
[82] L. Brakmo, S. O'Malley, and L. Peterson, "TCP Vegas: New techniques for congestion detection and avoidance," *Proc. SIGCOMM '94 Symp.*, Aug. 1994.
[83] L. Brakmo and L. Peterson, "TCP Vegas: End to End Congestion Avoidance on a Global Internet," *IEEE JSAC*, vol. 13, no. 8, Oct. 1995.
[84] Wireless Application Protocol Forum (www.wapforum.org), Apr.2001.
[85] NTT DoCoMo, All about I-mode, http://www.nttdocomo.com/i/index.html, Apr. 2001.
[86] Eurotechnology, Frequently Asked Questions about NTT-DoCoMo's i-mode, http://www.eurotechnology.com/imode/faq.html, Apr. 2001.
[87] J. B. Postel, "User Datagram Protocol," RFC 768, Aug. 1980.
[88] Cingular Wireless (www.cingular.com), Apr. 2001.
[89] Motient Corporation (www.motient.com), Apr. 2001.
[90] A. K. Salkintzis, "A Survey of Mobile Data Networks," *IEEE Commun. Surveys and Tutorials*, vol. 2, no. 3, 3rd Quarter 1999.
[91] Wireless Data University (www.wirelessu.com), Apr.2001.
[92] Siemens, UMTS by Siemens - Lexicon, http://www.siemens.nl/umts/lexicon, Apr. 2001.
[93] B. Walke, *Mobile Radio Networks*, John Wiley & Sons, 1999.
[94] Metricom (www.metricom.com), Apr. 2001.
[95] European Telecommunications Standard Institute, Broadband Radio Access Networks, http://www.etsi.org/bran, Apr. 2001.
[96] IEEE P802.16, The Working Group for Broadband Wireless Access, http://grouper.ieee.org/groups/802/16/, Apr. 2001.

BIOGRAPHY

KOSTAS PENTIKOUSIS (kostas@cs.sunysb.edu) is pursuing a Ph.D. in Computer Science at the State University of New York at Stony Brook. His research interests lie in the area of computer networks, including transport and application-layer protocols, mobile computing, wireless communications, and network management. He received a B.Sc. (honors) from Aristotle University of Thessaloniki, and a M.Sc. from SUNY at Stony Brook, both in Computer Science.