Wireless Personal Communications (2005) 32: 339–356
DOI: 10.1007/s11277-005-0751-2

Enhancing Transport Layer Capability in HAPS–Satellite Integrated Architecture

C.E. PALAZZI[1,2], C. ROSETI[3], M. LUGLIO[3], M. GERLA[2], M.Y. SANADIDI[2] and J. STEPANEK[2]

[1]*Dipartimento di Scienze dell'Informazione, Università di Bologna, Via Mura Anteo Zamboni 7, 40127 Bologna, Italy*
[2]*Computer Science Department, University of California Los Angeles, Boelter Hall, Los Angeles, CA 90095, U.S.A.*
E-mail: cpalazzi@csr.unibo.it, gerla@cs.ucla.edu, medy@ucla.edu, stepanek@cs.ucla.edu
[3]*Dipartimento di Ingegneria Elettronica, Università di Roma Tor Vergata, Via del Politecnico 1, 00133 Rome, Italy*
E-mail: roseti@ing.uniroma2.it, luglio@uniroma2.it

Abstract. The use of HAPS/UAV to enhance telecommunication capabilities has been proposed as an effective solution to support hot spot communications in limited areas. To ensure communication capabilities even in case of emergency (earthquake, power blackout, chemical/nuclear disaster, terrorist attack), when terrestrial fixed and mobile infrastructures are damaged or become unavailable, the access to satellites represents a reliable solution with worldwide coverage, even though it may suffer from shadowing impairment, especially in an urban environment. In this paper we approach an innovative and more challenging architecture foreseeing HAPS/UAV connected to the satellite in order to enlarge coverage and to allow interconnection with very remote locations. In this scenario, we have analysed TCP-based applications proposing some innovative techniques, both at protocol and at architectural level, to improve performance. In particular, we propose the use of a PEP technique, namely splitting, to speed up window growth in spite of high latency, combined with TCP Westwood as a very efficient algorithm particularly suitable and well performing over satellite links.

Keywords: HAPS, UAV, TCP, satellite, splitting, TCP Westwood

Abbreviations: ABSE, Adaptive Bandwidth Share Estimation; ARQ, Automatic Repeat Request; ERE, Eligible Rate Estimate; HAPS, High-Altitude Platform Station; RTT, Round-Trip Time; TCP, Transmission Control Protocol; UAV, Unmanned Aerial Vehicle

1. Introduction

To ensure telecommunication capabilities in emergency scenario requires the use of challenging architectures. The concept of using unmanned objects [1, 11, 23, 24], flying or stationary (HAPS/UAV) at relatively low altitudes has been previously introduced and proven effective for backup or capacity upgrade in high traffic areas (hot spots). The satellite is intrinsically suitable to provide service in such a scenario, as when used with HAPS/UAV, the combined architecture is capable of enhancing hot spot coverage with reasonable latency. While the HAPS/UAV provide short-range wireless connectivity at high rates even with small user terminals, the satellite can ensure large bandwidth for very long range connectivity, to reach remote headquarters/command posts all over the world.

In this paper, we revisit the performance of the protocols developed for Hot Spot scenarios when used in this novel, rather unconventional environment. With the HAPS/UAV + satellite connection we expect much higher loss rates on the ground-to-UAV link than in conventional Hot Spots because of distance, obstacles and UAV motion. To characterise such a scenario. classical channel models developed for satellite environments can be suitably applied [7, 18]. The exploitation of path diversity, both at theoretical and at simulation level, has also been investigated. The corresponding channel models [10, 12, 13, 17, 20] are suitable for a more complex scenario involving multiple HAPS/UAVs. Moreover, the satellite link has a large propagation delay and may introduce random loss of its own. Since much of the applications (e.g., web traffic, area maps, image files, emergency reports, etc.) will run on TCP, it is important to evaluate the performance of TCP in this environment. We propose to study two different ways of maintaining TCP connections:

1. End-to-end connections, from ground user to Internet server. We will evaluate different TCP protocol choices including the legacy TCP New Reno and the newly proposed TCP Westwood.
2. Proxy server on board of the UAV. The idea is to split the TCP connection on the HAPS/UAV and thus reduce the problem into two much easier sub-problems, i.e., a very lossy link with short propagation delay; and a more reliable (or at least, more predictable) link with very large propagation delay. In this case, different TCP "flavors" may be needed for the different link characteristics.

A system with multiple HAPS/UAVs in the sky to provide more extensive coverage can additionally be identified. This latter scenario is clearly much more complex as it also introduces the possibility of HAPS/UAV to HAPS/UAV communications (possibly on a separate radio channel). Moreover, users in an "urban canyon" can handoff to another HAPS/UAV if the first HAPS/UAV goes beyond a building out of sight. The handoff must be smooth in order to prevent disruption of ongoing sessions (e.g., video conference among emergency teams). In this paper, we will limit ourselves to single HAPS/UAV scenario. The results can be easily extended, however, to multiple HAPS/UAVs flying platforms.

2. System Architecture

We consider an urban scenario with mobile users (pedestrians and cars) connected by a very efficient communication infrastructure comprising the cellular system as well as a Mesh Network with Hot Spots placed in strategic locations (busy street crossings, tall buildings, parks, shopping malls, airport lounges, etc.) to access multimedia services. In particular, data services (mainly, access to the Internet from mobile users), if the trend is confirmed, will see a dramatic growth in the next few years, owing to the introduction of new mobile services such as location-based resource discovery (e.g., nearest drugstore), navigation support, web access, etc. In this environment, which is becoming increasingly dependant on communications, in case of emergency situation when power goes out and shortly thereafter all the Hot Spots and Cellular Base Stations are shut off, communications come to a standstill until power and telecommunication systems are restored. In addition, communications are most needed to control vehicular traffic and to allow users to "navigate" their way (or alternatively, be remotely guided) out of the traffic congestion. At the same time, repair crews, police and medical teams need efficient communications to coordinate their work. A similar emergency scenario in an urban area occurs when there is a chemical or nuclear disaster caused by human error, plant break down, act of war, or terrorist attack. Again in such situations the communication infrastructure will have been impaired while the need for efficient communications remains critical.

In the depicted scenario, to deploy in a short time a system composed of several HAPS/UAVs to establish an emergency telecommunication infrastructure is very cost effective and relatively easy. Due to their unmanned nature, HAPS/UAVs can be deployed very rapidly and flown even in environments potentially dangerous to humans, as those polluted by chemical fumes or nuclear radiations. Assuming that the on-board radio is the same as the ground Hot Spot radio, once in place, the HAPS/UAV will act like a Hot Spot to the customers on the ground permitting them to communicate among one another but also to access the Internet, or a remote command post, via satellite.

The HAPS/UAV cruise through the ‘urban canyons’ acting as repeaters and propagating the received signal to a wireless receiver on the ground, exploiting paths that involve other HAPS/UAV, or the GEO satellite connected to a remote gateway located far away from the disaster area [21]. In this way, shadowing impairment may be mitigated. The satellite antenna installation represents a significant weight and cost factor in the HAPS/UAV implementation. An alternative realistic scenario involves a mobile ground satellite station, for example a truck equipped with Wi-Fi and satellite transceivers, as depicted in Figure 1. In this case, HAPS/UAV can be relieved of satellite transmitter overhead. HAPS/UAV can thus be very simple, requiring very little maintenance and rare upgrades, while the latest technology innovation can be easily implemented in the “on-wheels” satellite station.

This solution greatly enhances “prompt response” in case of emergency. Both trucks and lightweight UAVs can be deployed literally within minutes after the accident. UAVs can fly at elevations of a few thousand feet, well below commercial air lanes. Heavy HAPS (with satellite gear) on the other hand would require much more elaborate and time consuming launching and navigation procedures. In addition, the higher altitude with respect to UAV implies slightly different performance in terms of link budget and delay. On the negative side, there is an additional link on the end-to-end path (the link from UAV to satellite truck). Moreover, since the UAVs in the urban emergency environment fly much lower than the HAPS in the operational

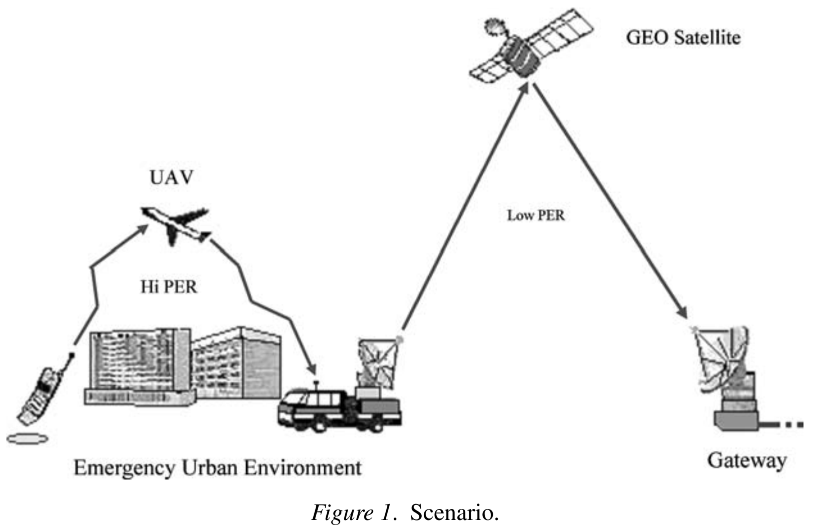


Figure 1. Scenario.

environment, it may be necessary to deploy multiple trucks, or a ‘constellation’ of multiple UAVs to provide adequate UAV-to-Truck connectivity.

The eventual presence of several flying units can greatly enlarge the area covered by the proposed architectures and the exploitation of path diversity can improve availability in an urban environment. Indeed, this two-level “satellite empowered” architecture includes the advantages of very small mobile terminal technology, the ability of handling high data rates, and the capability to establish very long-range connections.

In the following subsection, we include a more detailed description of the HAPS/UAVs features.

2.1. High-Altitude Platform Stations (HAPS)

The proposed solution for an emergency urban environment demands highly efficient broadband and multimedia services. Ideally, the goal is to build a wireless network able to efficiently cover a wide area with low propagation delays and negligible multi-path fading. In this context, radio communications could be based on higher frequency bands looking for larger bandwidths. Unfortunately, the use of this spectrum of frequencies implies line-of-sight propagation between the base station and each customer terminal since local obstructions cause problems.

The concept of High-Altitude Platform Stations (HAPS) is not new. What is quite new is the willingness to utilise such platforms as a structure to locate a communication payload. The capability to offer connectivity over large areas utilizing a device located at a few kilometers (in near space), represents a trade off solution between very distant satellites and terrestrial repeaters. In fact, HAPS do not require complex launch, can be easily moved from one position to another, and can be easily repaired and re-launched. Moreover, free space losses and propagation delay are not critical at all. The flying objects usable for this scope can be classified as stationary (blimps) or moving, both usually unmanned (even though someone has proposed manning such vehicles). The balloons have the advantage that they appear fixed with respect to an Earth observer, but the mechanical stabilization is extremely critical. On the other hand, moving platforms need to be tracked from the ground even though they actually fly over limited areas. The coverage area of a platform typically has a diameter up to 200 km subdivided in cells of 1–10 km.

HAPS located in the stratosphere at 17–22 km can carry communication relay payloads and operate in a quasi-stationary position. The payload may include a complete base-station and a satellite transponder. Therefore, HAPS can combine the best features of both terrestrial and satellite systems leading to a powerful integrated network. A single HAPS can replace a large number of terrestrial resources covering a wide area by providing a flexible cellular frequency re-use structure with a reduction in service costs. Basically, HAPS are aircrafts or airships (essentially balloons, termed “aerostats”) and can be manned or unmanned; in the latter case, autonomous operations can be coupled with remote control from a control ground station. Throughout the evolution of HAPS, three types of vehicles have been distinguished:

1. *Unmanned Airships*. These can range from small expendable balloons flying at modest altitudes to large airships operating at about 21 km altitude. Typically, the balloons are solar-powered and filled with helium.
2. *Unmanned Aircrafts*. This type of HAPS is an unmanned solar-powered plane, which can fly against the wind or in a roughly circular tight path to maintain a position as stable

as possible. These platforms are power-limited and need to store sufficient energy for station-keeping throughout the night. A fuelled unmanned version of these aircrafts, flying generally at modest altitudes, is specifically known as Unmanned Aerial Vehicle (UAV).

3. *Manned Aircraft*. These are represented by the conventional fuelled aircraft.

The advantages of HAPS communications can be summarised as follows:

Rapid deployment. Unlike satellites, the HAPS-based services can be designed, implemented and deployed relatively quickly. This presents obvious advantages in terms of rapid restoration of communication resources, after a disaster or in emergency scenarios.

Relatively low cost upgrading of the platform. The possibility of effortlessly landing and re-launching permits easy upgrades or reconfigurations of the payload, allowing a high degree of "future- proofing".

Broadband capability. Potentially, a large number of users can be served exploiting mm-wave frequencies with their large bandwidth allocations and offering line-of-sight links over many areas.

Flexibility to respond to traffic demands. HAPS can provide an efficient resource allocation through flexible and responsive frequency re-use patterns and adaptable cell sizes.

Low propagation delay. Delays of platforms located in the stratosphere are negligible compared with satellite delay, thus providing remarkable advantages for Internet best effort (TCP) and interactive applications.

3. Protocol Enhancement

For effective bandwidth access in the scenario presented earlier, efficient algorithms must be implemented to guarantee wireless error resilience. In the following subsection, we describe two complementary schemes that have both already demonstrated significant improvements in TCP performance in a wireless environment: TCP Westwood and the splitting technique.

3.1. TCP Westwood

In TCP Westwood (TCPW) [19], the sender continuously monitors ACKs from the receiver and computes its current Eligible Rate Estimate (ERE). This scheme relies on an adaptive estimation technique applied to the ACK stream. The goal of ERE is to estimate the rate a connection is eligible for depending on congestion and bandwidth on the path, and thus achieving high utilization without starving other connections. Research on active network estimation [8] reveals that samples obtained using "packet pairs" often reflects physical capacity, while samples obtained using "packet trains" gives short-term throughput estimates. Not having the luxury to estimate using active probing packets, a TCPW sender carefully chooses sampling intervals and filtering techniques to estimate the eligible rate of a connection. Duplicate acknowledgments (DUPACKs) and delayed ACKs are also properly counted in ERE computation.

In all TCPW implementations, upon packet loss (indicated by three DUPACKs or a time-out), the sender sets the congestion window (cwnd) and the slow start threshold (ssthresh) based on the current ERE. TCPW uses the following algorithm to set cwnd and ssthresh.

```
if (3 DUPACKS are received)
    ssthresh = (ERE * RTTmin) / seg_size;
    if (cwnd > ssthresh) /*congestion avoidance*/
        cwnd = ssthresh;
    endif
endif

if (coarse timeout expires)
    cwnd = 1;
    ssthresh =(ERE * RTTmin) / seg_size;
    if (ssthresh < 2)
        ssthresh = 2;
    endif
endif
```

The EREs are determined using a time-varying coefficient, exponentially weighted moving average (EWMA) filter, which has both adaptive gain and adaptive sampling. Let t_k be the time instant at which the kth ACK is received at the sender. Let s_k be the ERE sample, and $\hat{s}_k$ the filtered estimate of the ERE at time t_k. Let (α_k be the time-varying coefficient at t_k. The "Adaptive Bandwidth Share Estimation" (ABSE) version [26] of the filter is given by:

$$\hat{s}_k = \alpha_k \hat{s}_{k-1} + (1 - \alpha_k) s_k \tag{1}$$

where $\alpha_k = \frac{2\tau_k - \Delta t_k}{2\tau_k + \Delta t_k}$, and τ_k is a filter parameter that determines the filter gain, and varies over time adapting to Round-Trip Time (RTT) and other path conditions. Later, we assume a fixed τ_k and focus on adaptive sampling.

In the filter formula, the ERE sample at time k is $s_k = \frac{\sum_{d_{t_j} > t_k - T_k} d_{t_j}}{T_k}$, where d_{t_j} is the number of bytes that are reported delivered by the jth ACK, and T_k is an interval over which the ERE sample is calculated.

To preserve both efficiency and fairness, TCPW exploits a continuously adaptive sampling interval T: the more severe the congestion, the longer T should be. The time interval T_k associated with the kth received ACK is appropriately chosen between two extremes, as depicted in Figure 2 depending on the network congestion level. The sampling interval, in fact, ranges between $T_{\min}$ and $T_{\max}$, where $T_{\min}$ corresponds to an ACK-pair inter-arrival time, while $T_{\max}$ is set to RTT.

To determine the network congestion level, the ABSE estimator compares the "Achieved Rate" with the instantaneous sending rate which is equal to *cwnd/*RTT$_{\min}$. A measure of the

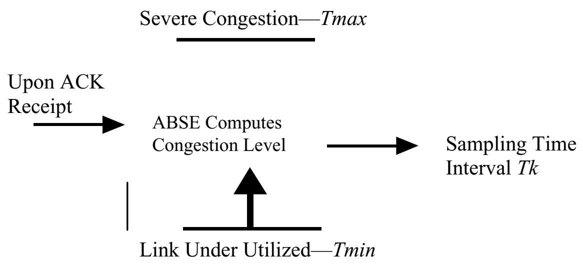

Severe Congestion—*Tmax*
Upon ACK Receipt
ABSE Computes Congestion Level
Sampling Time Interval *Tk*
Link Under Utilized—*Tmin*

Figure 2. Illustration of sampling time interval T_k adapting to network congestion level.

path congestion level is thus obtained. The difference between the instantaneous sending rate and the achieved rate, clearly reflects the bottleneck queue, thus revealing that the path is becoming congested. The larger the difference, the more severe the congestion, and the larger the new value of T_k should be.

When the kth ACK arrives, the estimator first checks the relation between the latest ERE estimate $\hat{s}_{k-1}$ and the current *cwnd* value. When $\hat{s}_{k-1} * \mathrm{RTT}_{\min} \geq cwnd$, indicating a path without congestion, T_k is set to $T_{\min}$. Otherwise, T_k is set to:

$$T_k = \mathrm{RTT} * \frac{cwnd - (\hat{s}_{k-1} * \mathrm{RTT}_{\min})}{cwnd} \tag{2}$$

or upon rearrangement:

$$T_k = \mathrm{RTT} * \frac{\left(\frac{cwnd}{\mathrm{RTT}_{\min}} - \hat{s}_{k-1}\right)}{\frac{cwnd}{\mathrm{RTT}_{\min}}} \tag{3}$$

In (3), $cwnd/\mathrm{RTT}_{\min}$ is the expected sending rate, while $\hat{s}_{k-1}$ is the estimated rate the network allowed. TCPW has been shown to provide higher efficiency while remaining friendly to other flows coexisting with a TCPW flow [25].

3.2. Splitting

Another class of approaches involves additional or enhanced network infrastructure. In this case, intermediaries perform processing on behalf of TCP endpoints to the greater benefit of performance. This basic idea generalises to the so-called “Performance Enhancing Proxy” or PEP [4]. Of the many PEP schemes, one involves segmenting, or “splitting” the TCP connection into segments. Splitting makes use of TCP gateways that maintain multiple TCP connections with both other gateways and end users. In fact, between gateways splitting may use a specialised or optimised transport protocol [15].

Naturally, subdividing the connection results in reduced RTT for packets transmitted in each subsection of the original path. But as a consequence, the node hosting the gateway must include an implementation of TCP that forwards packets between connections. So the reduction in response time comes at the expense of memory and processing. Moreover, splitting violates the end-to-end semantics of TCP, that is, packets corrupted while in the gateway’s buffer cannot be recovered through TCP means and thus result in a connection failure. Having noted these consequences, many of the same system-level arguments in favour of splitting on-board satellites apply equally to other nodes along the connection [22]. As another advantage, splitting can allow disparate terminals to communicate through the gateway, that is, it supports TCP terminals without requiring uniform end-to-end TCP. Some hops may use multicast, and thus more effectively exploit the broadcast nature of wireless transmission while still supporting TCP endpoints. Splitting TCP gateways also provides more rapid recovery from errors and improves the overall robustness of the end-to-end connection when several links suffer from shadowing or high error-rates.

A competing approach to connection splitting introduces an ARQ link layer. For example, the IEEE 802.11 MAC protocol used in wireless LANs includes an ACK. This does achieve many of the previously described advantages, but by operating at the link layer, an ARQ mechanism forces all flows to undergo ARQ, even those such as real-time applications like Voice over IP for which ARQ is notably the wrong solution. Beyond this system design consideration, link-level ARQ can introduce delay variations that actually degrade TCP performance unless TCP is further optimised to account for them.

Having considered the benefits of connection splitting, we now more carefully consider the design of the TCP gateway. To effect connection splitting, a TCP forwarding agent receives data and sends acknowledgements on incoming connections while sending and retransmitting data and receiving acknowledgments on outgoing connections. When the forwarding agent receives a valid incoming data packet, it acknowledges this packet and places the packet in memory to be sent on the corresponding outgoing connection. Consistent with the TCP sender, this packet remains in memory until acknowledged by the outgoing connection. Once this happens, the packet may be removed from memory.

Note that an unrestrained sender can quickly overflow the agent's memory space. As a result, the *forwarding agent* must control the rate of incoming senders. To accomplish this, the agent uses TCP's advertised window mechanism. By progressively limiting the size of the window as memory fills, the agent also limits the rate of the sender. Thus, as packets become backlogged on the gateway node, the advertised window decreases, resulting in backpressure on the sender. By limiting the advertised window to half of free memory allocated to that connection, the gateway eliminates the possibility of dropping packets due to buffer overflow. However, when unacknowledged packets consume the total allocated space, the agent will advertise an empty window, thus stopping the sender. In this case, the agent must send an additional acknowledgment when space becomes available in order to restart the sender.

4. Simulation Scenario

Traditional TCP New Reno (TCPNR) [9] was used to baseline the performance characteristics of our architecture. We subsequently compared these results with results obtained using TCPW, a transport protocol designed to handle wireless errors. We used the NS-2 simulator [5] to verify the performance gains obtained with by our proposed architecture. NS-2 is widely used for simulating networks and is especially valued for its models of transport protocols.

4.1. The Simulation Platform

NS-2 provides substantial support for discrete event-driven simulations at various network levels. In fact, a network configuration can be simulated integrating physical, routing, MAC, transport and application layers both on wired and wireless environment (LAN, satellite, etc.). Distributed with source code, NS-2 allows adding new models and functionality for the purpose of investigating different scenarios and new protocols. In fact, to run our tests we enhanced version 2.1b8a of NS-2 by adding some new modules written in C++. In particular, we added code to simulate the behavior of TCPW and the splitting scheme of the HAPS/UAV. Using this tool, we performed an exhaustive set of experiments with the following parameters:

– packet size,
– queue size at nodes and cache size at the proxy (when present),
– location of the proxy (when present),
– propagation delay of each link,
– capacity of each link,
– error rate on each link,

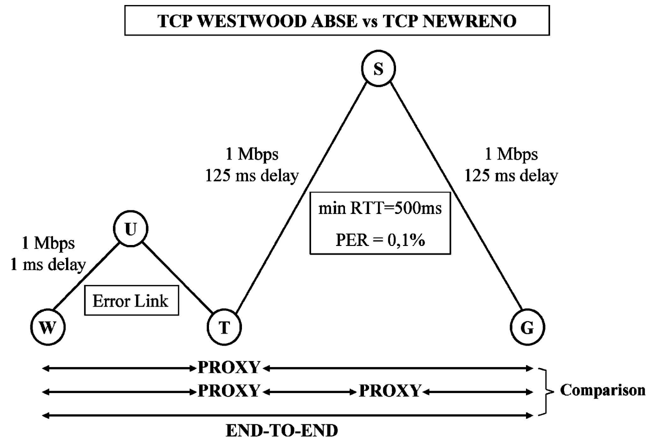


Figure 3. Simulation configuration.

– number of flows present on the channel,
– protocols involved,

4.2. SIMULATION CONFIGURATIONS

Figure 3 depicts the general topology adopted in our simulation corresponding to the architecture of Figure 1. Here, each of the circles in the picture corresponds to one of the actors in the proposed scenario, more specifically:

– W: generic wireless device in the urban area,
– U: HAPS/UAV acting as a signal repeater,
– T: Truck with a satellite station (eventually with proxy on board),
– S: GEO satellite (eventually with proxy on board),
– G: gateway on the ground,

The edges between the various nodes represent wireless connections. Note that the propagation delay between W and T, through U, is small considering the flying altitude of the HAPS/UAV. And the delay between a both between T and S and between S and G is a GEO satellite delay (125 ms). Previous work has characterised errors on wireless links [2, 6]. For our scenarios, we used a uniform Packet Error Rate (PER) of 0.1% between T and G [3, 14, 16]. For links between W and T, considered a wide variety of urban conditions. Shadowing was not considered because we assume the user and UAV always remains within line-of-sight.

The bandwidth available on each link is 1 Mbps and the packet-size is 1500 bytes, thus the end-to-end connection has a pipe capacity of about 42 packets. Each adjacent node has a buffer of 50 packets, while the proxy has a buffer of 200 packets, if not otherwise specified. Each simulation utilised different combinations of transport protocols, various PERs on the ground-UAV-ground wireless links, presence or absence of a proxy on the ground satellite station and on the satellite, diverse dimensioning of the cache in the proxy and alternate direction of the data flow. To summarise, the various alternatives include the following:

– transport protocol:

- TCPNR, TCPW,

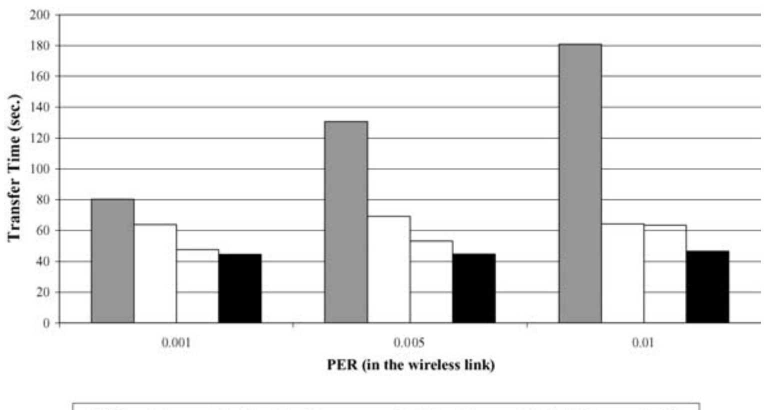
– PER on the link between W and U:
  - 0.1, 0.5, 1.0%,
– proxy on board:
  - on the ground satellite station: split enabled, split disabled,
  - on satellite: split enabled, split disabled,
– cache size on the proxy:
  - 10, 20, 30, ., 240, 250 packets,
– traffic direction:
  - from W to G, from G to W.

Using 20 independent simulation runs for each configuration we calculated the average throughput during a period of 230 s as well as the time required to transmit a 5 MB file.

5. Results

We will first consider performance when combining the use of TCPW and a splitting proxy in the presence of different PERs on the link between the wireless device and the terrestrial satellite station T through the airborne HAPS/UAV. Figure 4 shows the time required to transmit a 5 MB file from W to G.

In this case, TCPW demonstrates a significant advantage over TCPNR. In particular, without connection splitting, TCPW requires only from 59.33% (with 0.1% PER) to 34.99% (with 1% PER) of the time required by TCPNR. When an intermediate proxy is introduced at T, the gap between TCPNR and TCPW performance decreases significantly. In this case, TCPW only needs 72.52% (with 0.1% PER) to 64.70% (with 0.5% PER) of the time required by TCPNR. When applied separately, TCPW or splitting both lead to a substantial performance improvement. TCPW generally outperforms traditional transport protocols when employed on wireless links involving long satellite paths, however, its transmission time is further reduced



▨ NewReno_End-to-End	□ NewReno_Split (Proxy in T)
□ Westwood_End-to-End	■ Westwood_Split (Proxy in T)

Figure 4. Time to transmit a 5 MB file from W to G.

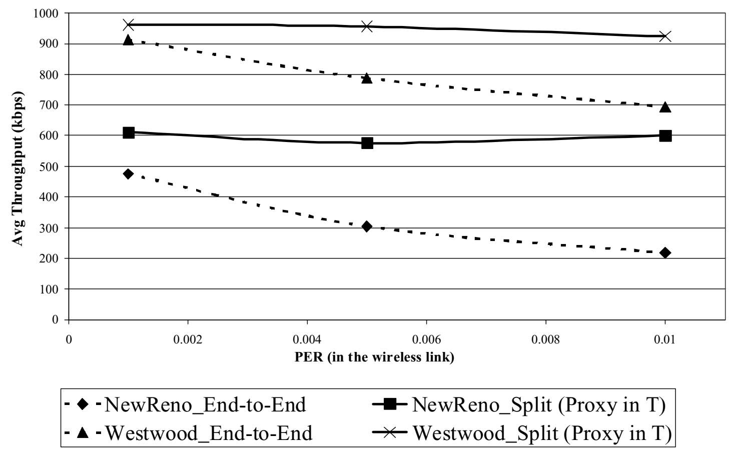


Figure 5. Average throughput over a 230 s transmission from W to G.

with splitting. Finally, splitting shows constant performance for different PER, demonstrating its talent in hiding wireless losses to the endpoints.

In Figure 5 we compare the average throughput achieved by different transport protocols, the eventual utilization of a splitting proxy on node T and various PER on links connecting W and G. For each configuration we ran 20 simulations averaging the number of bytes sent in 230 s to calculate the average throughput achieved. Again, TCPW coupled with a proxy shows the best results, with an average throughput ranging from 923.53 kbit/s (with a PER of 1.0%) to 961.18 kbit/s (with a PER of 0.1%). It appear splitting hides the frequent errors on the shortest wireless link from the rest of the connection: for each transport protocol, the average throughput achieved remains relatively constant regardless of PER.

For sake of completeness, a set of simulations was performed which considered the *reverse* data flow from G to W and the results closely follow the previous set. In this case, performance improves as a result of TCPW dealing with wireless links and the splitting at T protects the connection from the frequent errors present on the links between W and T. The average times required to transmit a 5 MB file from G to W are shown in Figure 6, while Figure 7 illustrates the average throughput attained on a 230 s simulation run.

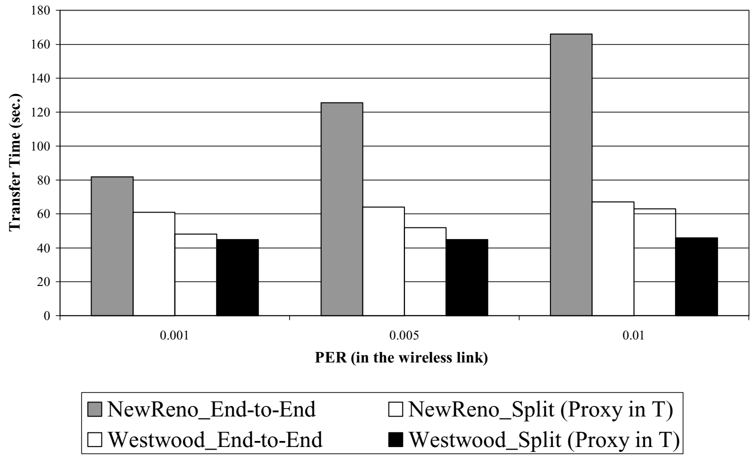


Figure 6. Time to transmit a 5 MB file from G to W.

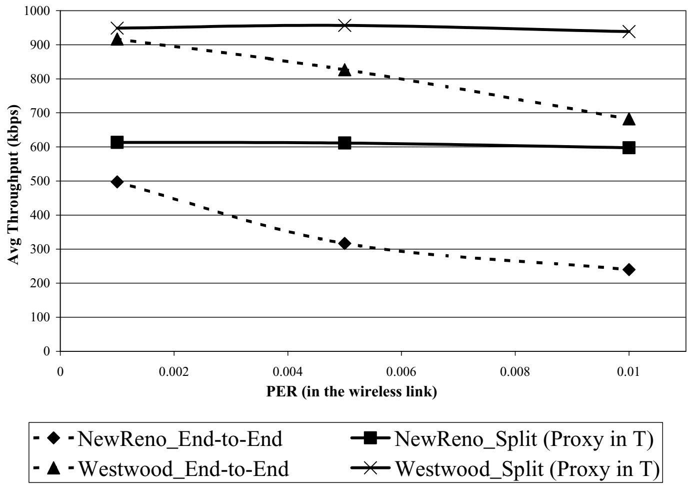


Figure 7. Average throughput over a 230 s transmission from G to W.

The results confirm the performance advantage of using splitting. Moreover, since proxies along the path generate ACKs faster than a distant receiver (with ≈500 ms RTT), the congestion window at the sender increases faster, thus speeding up the transmission rate. To better illustrate how the presence of proxies along the path improve overall performance, we present in Figure 8 the sending window of the last TCP segment along the path. Specifically, we utilised TCPNR with a 1.0% PER on the links between W and G. We then compared the pure end-to-end case with a proxy at T and with two proxies placed at T and S. In each this case, we computed the sending window of the TCP connection terminated at G, thus observing the instantaneous throughput at the receiver. With a pure end-to-end connection we naturally considered TCP between W and G, while including one proxy we have taken into account the connection between T and G and, finally, with two splits we have measured the sending values between S and G.

Without splitting, errors heavily impact TCP performance. In this case, the sending window increases for a small period before a wireless loss halves its value. This behavior, continuously

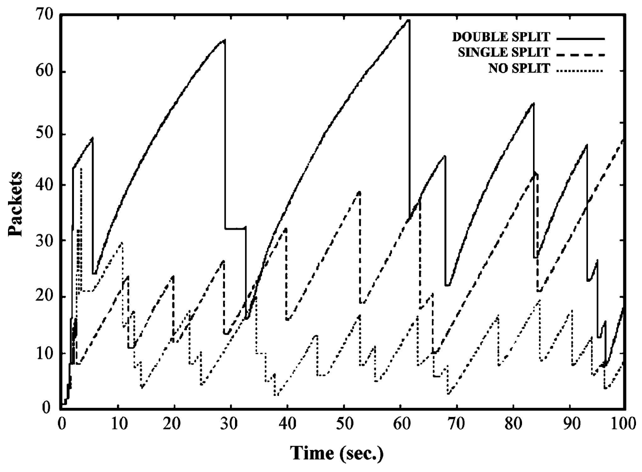


Figure 8. Sending windows at the last sending/forwarding hop with no split, single split at T, or double split at T and S.

repeated, results in a noticeable underutilization of the available bandwidth. Conversely, if the connection is split at T into two parts, the high error rate between W and T remains hidden from the rest of the link, thus resulting in higher sending rates for longer periods. Wireless losses, in fact, are in this case rapidly recovered along the short link between W and T, and T can store enough packets to efficiently utilise the long, but still quite reliable, link between T and G. Finally, when the end-to-end connection is split twice, the presence of the proxy on the satellite S shortens the duration of the send-acknowledge feedback-loop between T and G subdividing it into two cycles: the first one between T and S and the second one connecting S to G.

Using shorter connections accelerates the sending window growth and thus creates a pipelining phenomenon that increases overall throughput. Figure 9 compares split with end-to-end connections. In this scenario, a 5 MB file is downloaded from W by G. In this case, TCPNR reaps the most benefit from splitting by achieving about 91% of the capacity in case of double split and thereby achieving performance equivalent to TCPW.

We now consider optimizing the size of the buffer of a proxy at T. Figures 10 and 11 consider a configuration with 1.0% PER between W and T and compare the average throughput for TCPW and TCPNR with and without splitting. Clearly, an small amount of packets stored in the proxy substantially reduces throughput. All the configurations analysed in our simulations experience a steep performance degradation if we set the cache size to a value lower than the double of the channel capacity (84 packets). Under this threshold, in fact, since the advertised window sent back by the proxy corresponds to half of the free memory in cache, TCP wastes a considerable amount of available bandwidth. Indeed, with an undersized cache capacity, the proxy quickly runs out of data to transmit. The congestion window might be large enough to permit further transmissions, but the window goes unused since the undersized cache does not allow any prefetching and the proxy has exhausted the few packets stored.

In fact, a value twice the pipe size is a lower bound for the dimensions of the cache, while larger values clearly help with better utilization of the available bandwidth. Utilizing the proxy's buffer undeniably protects reliable links from being impacted by transient burst errors that temporarily slow down packets traversing adjacent error-prone edges. For completeness, we also report the average throughput achieved by TCPW and TCPNR without splitting.

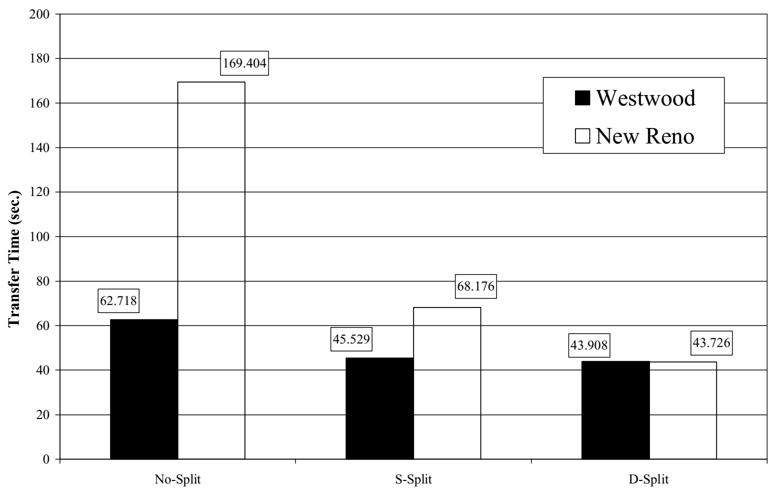


Figure 9. Average time to transmit a 5 MB file from W to G with no split, single split at T, or double split at T and S.

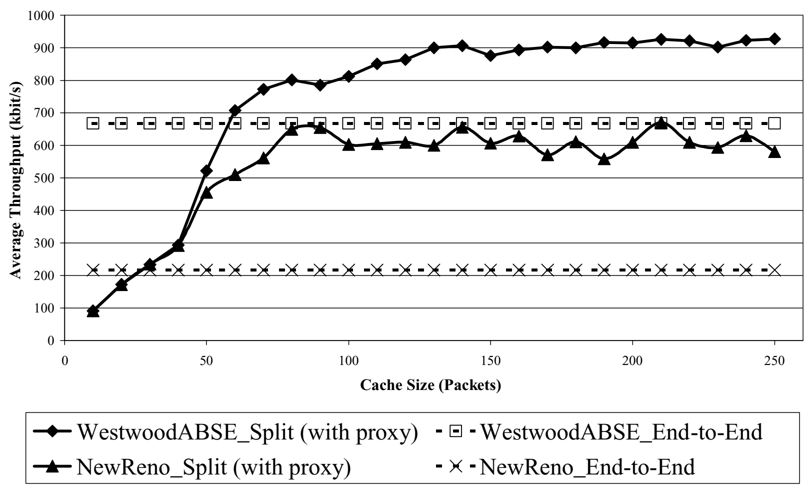


Figure 10. Performance achieved per proxy cache size. Transmissions from W to G, with a single proxy on T.

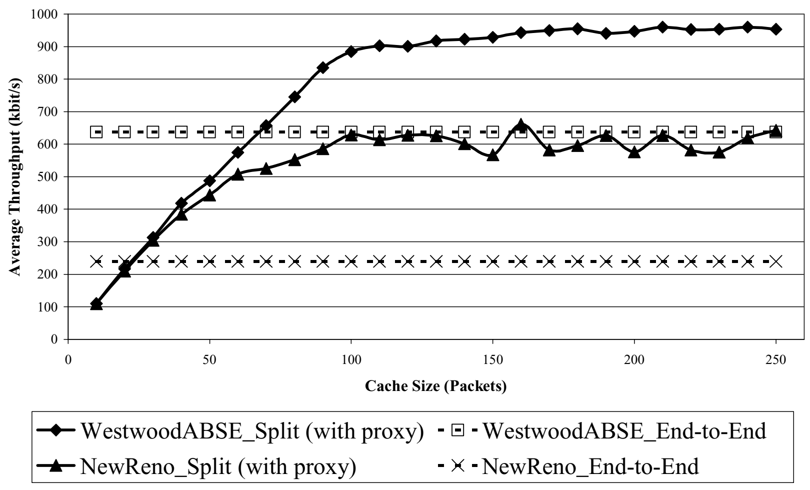


Figure 11. Performance achieved per proxy cache size. Transmissions from G to W, with a single proxy on T.

6. Conclusions

Guaranteeing telecommunication services in emergency scenarios and providing large bandwidth without reducing performance requires both a well-balanced architecture combined with efficient protocols. In this paper, we proposed the combined use of HAPS/UAVs and satellites as an innovative and challenging architecture able to meet the requirements when terrestrial infrastructures are unavailable. In addition, we evaluated use of TCP Westwood and TCP connection splitting at the ground station and satellite link as a means of improving efficiency.

The simulation results confirm the utility of TCP Westwood in error-prone environments. They also detail the benefits of connection splitting. As expected, TCP New Reno benefits the most from splitting. In fact, with two proxies, one on the ground and one on the satellite, the performance of TCP New Reno equals that of TCP Westwood.

References

1. D. Avagnina, F. Dovis, A. Ghiglione, and P. Mulassano, "Wireless Networks Based on High-Altitude Platforms for the Provision of Integrated Navigation/Communication Services", *IEEE Communications Magazine*, Vol. 40, No. 2, pp. 119–125, 2002.
2. H. Balakrishnan, V.N. Padmanabhan, S. Sehan, and R.H. Katz, 'A comparison of Mechanism for Improving TCP Performance Over Wireless Links', *IEEE/ACM Transactions on Networking*, Vol. 5, No. 6, pp. 756–769, 1997.
3. L. Baldantoni, H. Lundqvist, and G. Karlsson, "Adaptive End-to-End FEC for Improving TCP Performance Over Wireless Links", in *ICC 2004*.
4. J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby, 'RFC 3135: Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations', IETF RFC 3135, June 2001.
5. L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, and H. Yu, "Advances in Network Simulation", *IEEE Computer*, Vol. 33, No. 5, pp. 59–67, 2000. Expanded version available as USC TR 99-702b at http://www.isi.edu/~johnh/PAPERS/Bajaj99a.html.
6. R. Càceres and L. Iftode, "The Effects of Mobility on Reliable Transport Protocols", in *Proceedings of the 14th International Conference on Distributed Computing Systems*, pp. 12–20, 1994.
7. E. Corazza and F. Vatalaro, "A Statistical Model for Land Mobile Satellite Channels and its Application to Nongeostationary Orbit Systems", *IEEE Transactions on Vehicular Technolgy*, Vol. VT-43, pp. 738–741, 1994.
8. C. Dovrolis, P. Ramanathan, and D. Moore, "What do Packet Dispersion Techniques Measure?", in *IEEE Infocom '01*, Anchorage, Alaska, 2001.
9. S. Floyd and T. Henderson, 'The NewReno Modification to TCP's Fast Recovery Algorithm', IETF RFC 2582, April 1999.
10. F.P. Fontán, M.A. Vázquez, S. Buonomo, E. Kubista, and A. Paraboni, "A methodology for the Characterisation of Environmental Effects on Global Navigation Satellite System (GNSS) Propagation", *International Journal of Satellite Communications*, Vol. 16, pp. 1–22, 1998.
11. S. Karapantazis and F.-N. Pavlidou, "Broadband from Heaven", *IEE Communications Engineer*, Vol. 2, No. 2, 2004.
12. P. Loreti and M. Luglio, "Satellite Diversity: A Technique to Improve Link Performance and Availability for Multicoverage Constellations", in *3rd Generation Mobile Communication Systems, UMTS and IMT-2000*. Springer Verlag, Berlin, 2001a.
13. P. Loreti and M. Luglio, "A generalized N-State Model to Characterize Satellite Diversity for Arbitrary Number of Satellites in Case of Uncorrelated Channels", *IEEE Communications Letters*, Vol. 5, No. 11, pp. 447–449, 2001b.
14. M. Luby, L. Vicisano, J. Gemmell, L. Rizzo, M. Handley, and J. Crowcroft, 'Forward Error Correction (FEC) Building Block', IETF RFC 3452, December 2002.
15. M. Luglio, M.Y. Sanadidi, M. Gerla, and J. Stepanek, "On-board Satellite "Split TCP" Proxy", *IEEE Journal of Selected Areas in Communications*, Special Issue on "Broadband IP Networks via Satellites", Vol. 22, No. 4, pp. 362–370, 2004.
16. H. Lundqvist and G. Karlsson, "TCP with End-to-End Forward Error Correction", in *International Zurich Seminar on Communications (IZS 2004)*, 2004.
17. E. Lutz, "A Markoff Model for Correlated Land Mobile Satellite Channels", *International Journal of Satellite Communications*, Vol. 14, pp. 333–339, 1996.
18. E. Lutz, D. Cygan, M. Dippold, F. Dolainsky, and W. Papke, "The Land Mobile Satellite Channel – Recording, Statistics and Channel Model", *IEEE Transactions on Vehicular Technolgy*, Vol. VT-40, 1991.
19. S. Mascolo, C. Casetti, M. Gerla, M. Sanadidi, and R. Wang, "TCP Westwood: End-to-End Bandwidth Estimation for Efficient Transport Over Wired and Wireless Networks", in *Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking (MOBICOM-01)*, New York, pp. 287–297, 2001.
20. F. Mazzenga and F. Vatalaro, 'Channel Modeling and Performance Evaluation of LEO Systems', in *AP2000 Davos*, Switzerland, 2000.
21. C.E. Palazzi, C. Roseti, M. Luglio, M. Gerla, M.Y. Sanadidi, and J. Stepanek, "Satellite Coverage in Urban Areas Using Unmanned Airborne Vehicles (UAVs)", in *IEEE Vehicular Technology Conference*, Milan, Italy, 2004.

22. J. Stepanek, A. Razdan, A. Nandan, M. Gerla, and M. Luglio, "The Use of a Proxy on Board the Satellite to Improve TCP Performance", in *IEEE Global Telecommunications Conference (Globecom)*, Vol. 21. pp. 2957–2961, 2002.
23. J. Thornton, D. Grace, C. Spillard, T. Konefal, and T. Tozer, "Broadband Communications from a High Altitude Platforms", *IEE Electronics and Communications Engineering Journal*, Vol. 13, No. 3, pp. 138–144, 2001.
24. T.C. Tozer and D. Grace, "High-Altitude Platforms for Wireless Communications", *Electronics & Communication Engineering Journal*, Vol. 13, No. 3, pp. 127–137, 2001.
25. R. Wang, M. Valla, M. Sanadidi, B.K.F. Ng, and M. Gerla, "Efficiency/Friendliness Tradeoffs in TCP Westwood", in *Seventh IEEE Symposium on Computers and Communications*, Taormina, Italy, 2002a.
26. R. Wang, M. Valla, M.Y. Sanadidi, and M. Gerla, "Adaptive Bandwidth Share Estimation in TCP Westwood", in *Proceedings of the IEEE Globecom 2002*, Taipei, Taiwan, ROC, 2002.

Cesare Roseti graduated cum laude in 2003 in Electronic Engineering at University of Rome "Tor Vergata". In 2003 and 2004, he was a visiting student at Computer Science Department of University of California, Los Angeles (UCLA). Since 2004 he is a PhD student at the Electronic Engineering Department and his research interests include satellites communications and transport protocols in heterogeneous (wired/wireless) systems.

Claudio Enrico Palazzi studied computer science at University of Bologna, Campus of Cesena. He has been a student representative in several bodies of University of Bologna and, in particular, from 2000 to 2001 he was part of the Board of Governors. In 2001, he received the Sigillum Magnum of Alma Mater Studiorum University of Bologna. He graduated cum laude in 2002 with a thesis on transport protocols in wireless environments. In 2003, he was the first student enrolled in the Interlink joint PhD program in computer science by which he is currently a PhD student in Computer Science at both University of Bologna and University of California, Los Angeles (UCLA). His research interests include protocol design, implementation and performance analysis for wired/wireless networks.

Michele Luglio received the Laurea degree in electronic engineering from the University of Rome “Tor Vergata”. He received the PhD degree in telecommunications in 1994. From August to December 1992 he worked as visiting staff engineering at Microwave Technology and Systems Division of Comsat Laboratories (Clarksburg, Maryland, USA). He received the Young Scientist Award from ISSSE’95. Since October 1995, he is research and teaching assistant at University of Rome “Tor Vergata” where he works on designing satellite systems for multimedia services both mobile and fixed, in the frame of projects funded by EC, ESA and ASI. He taught signal theory and collaborated in teaching digital signal processing and elements of telecommunications. In 2001 and 2002 he was visiting professor at the Computer Science Department of University of California Los Angeles (UCLA) to teach Satellite Networks class. Now he teaches satellite telecommunications and signals and transmission. He is a member of IEEE.

Mario Gerla received a graduate degree in engineering from the Politecnico di Milano in 1966, and the MS and PhD degrees in engineering from UCLA in 1970 and 1973, respectively. After working for Network Analysis Corporation from 1973 to 1976, he joined the Faculty of the Computer Science Department at UCLA where he is now professor. His research interests cover the performance evaluation, design and control of distributed computer communication systems; high-speed computer networks; wireless LANs; and ad hoc wireless networks. He has worked on the design, implementation and testing of various wireless ad hoc network protocols (channel access, clustering, routing and transport) within the DARPA WAMIS, GloMo projects. Currently, he is leading the ONR MINUTEMAN project at UCLA, and is designing a robust, scalable wireless ad hoc network architecture for unmanned intelligent agents in defense and homeland security scenarios. He is also conducting research on QoS routing, multicasting protocols and TCP transport for the Next-Generation Internet (see www.cs.ucla.edu/NRL for recent publications). He became IEEE Fellow in 2002.

M. Yahya "Medy" Sanadidi was born in Damanhour, Egypt. He received his high school diploma from College Saint Marc, and his BSc in electrical engineering (computer and automatic control section) from the University of Alexandria, Egypt. Dr. Sanadidi received his PhD in computer science from UCLA in 1982. He is currently a research professor at the UCLA Computer Science Department. As co-principal investigator on NSF-sponsored research, he is leading research in modeling and evaluation of high-performance Internet protocols. He teaches undergraduate and graduate courses at UCLA on queuing systems and computer networks. Dr. Sanadidi was a manager and senior consulting engineer at Teradata/AT&T/NCR from 1991 to 1999 and led several groups responsible for performance modeling and analysis, operating systems, and parallel query optimization. From 1984 to 1991, he held the position of computer scientist at Citicorp, where he pursued R&D projects in wireless metropolitan area data communications and other networking technologies. In particular, between 1984 and 1987, he lead the design and prototyping of a wireless MAN for home banking and credit card verification applications. From 1981 to 1983, Dr. Sanadidi was an assistant professor at the Computer Science Department, University of Maryland, College Park, Maryland. There, he taught performance modeling, computer architecture and operating systems, and was principal investigator for NSA-sponsored research on global data communications networks. Dr. Sanadidi has consulted for industrial concerns, has co-authored conference as well as journal papers, and holds two patents in performance modeling. He participated as reviewer and as program committee member of professional conferences. His current research interests are focused on congestion control and adaptive multimedia streaming protocols in heterogeneous (wired/wireless) networks.

James Stepanek received his BS in computer science from Harvey Mudd College in 1994 and his MS in computer science from University of California, Los Angeles (UCLA) in 2001 where he is currently enrolled in the PhD program. He is also currently a member of the technical staff in the Computer Systems Research Department of The Aerospace Corporation. His research interests include wireless and satellite networks.