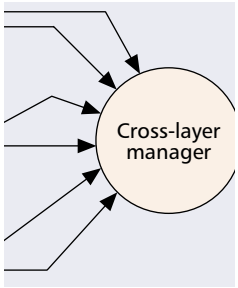
CROSS-LAYER DESIGN IN 4G WIRELESS TERMINALS

GUSTAVO CARNEIRO, INESC PORTO
JOSÉ RUELA AND MANUEL RICARDO, INESC PORTO AND FACULTY OF ENGINEERING, PORTO UNIVERSITY



The classical TCP/IP layered protocol architecture is beginning to show signs of age. In order to cope with problems such as the poor performance of wireless links and mobile terminals, a protocol architecture that considers cross layer interactions seems to be required.

ABSTRACT

The classical TCP/IP layered protocol architecture is beginning to show signs of age. In order to cope with problems such as the poor performance of wireless links and mobile terminals, including the high error rate of wireless network interfaces, power saving requirements, quality of service, and an increasingly dynamic network environment, a protocol architecture that considers cross-layer interactions seems to be required. This article describes a framework for further enhancements of the traditional IP-based protocol stack to meet current and future requirements. Known problems associated with the strictly layered protocol architecture are summarized and classified, and a first solution involving cross-layer design is proposed.

INTRODUCTION

One of the foundations of the Internet has been its protocol architecture. This architecture is characterized by a stack of protocol modules where each protocol solves a specific problem by using the services made available by modules below it, and providing a new service to upper layers. In order to maximize modularity, communication happens mainly between adjacent layers, and is limited to a minimum set of primitives. The traditional protocol stack is composed of the protocol modules TCP over IP over a *link layer* (e.g., Ethernet). Alternatively, RTP/UDP is sometimes used instead of TCP. Figure 1a represents such a stack. The link layer (Ethernet) provides connectivity to other hosts in the same network segment, but not to hosts in different networks. The *network layer*, IP, uses the primitives from the link layer (sending and receiving frames to hosts) to deliver datagrams across multiple networks. Finally, the *transport layer*, TCP, uses the services provided by the network layer (sending and receiving datagrams) to provide a connection-oriented communication service, adding reordering, error recovery, flow control, and congestion control.

The new communications scenarios foreseen in the fourth-generation networks are forcing this stack to also incorporate *quality of service* (QoS) and *mobility* functions. QoS is required to deploy real-time communications over IP networks; with QoS, traditional telephony and data services can be jointly supported by a single IP-based packet-switched network. Mobility functions are also required. Micromobility allows terminals to roam between adjacent wireless access points of the same technology using layer 2 or IP-based solutions. But there is also strong demand to support mobility across networks of different technologies (and administrative domains). Future terminals are expected to be equipped with multiple wireless network interfaces, including 802.11, Universal Mobile Telecommunications System (UMTS), and Bluetooth. Each technology has its own characteristics in terms of coverage, cost, and bandwidth. The *always best connected* concept implicit in 4G requires the terminal to select the best access method available. Thus, mobility support at the IP level becomes very important.

4G terminals need, as usual in wireless terminals, to be portable. Terminal portability places serious constraints on the size of batteries. Therefore, it is imperative to reduce power consumption to a minimum. The protocol stack behavior of such terminals must also take this issue into consideration.

Although far from consolidated, the protocol stack of 4G terminals tends to follow the architecture shown in Fig. 1b. The central MIPv6 (Mobile IPv6) module results from the migration of IPv4 to IPv6, and includes terminal mobility functions. Unlike IPv4, IPv6 accommodates a much larger address space, thus enabling terminals to have public addresses. Using MIPv6, a terminal is able to move to a new access point, a new operator, switch between access interfaces, or use multiple access interfaces simultaneously while maintaining its active connections without user-visible reconfiguration.

However, by retaining the strictly modular architecture, where each layer communicates mainly with its adjacent layers, the protocol stack will be inefficient with respect to performance, QoS, and energy consumption, and may have a negative impact on 4G networking.

In order to help solve these issues, this article

 1536-1284/04/$20.00

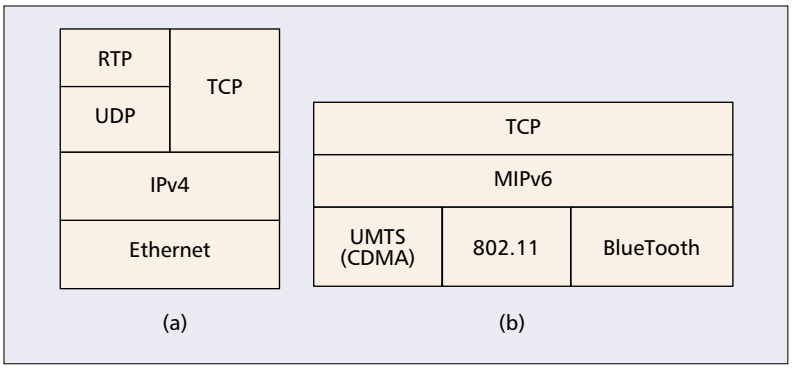


Figure 1. *Internet protocol stack evolution: a) current stack; b) likely future stack.*

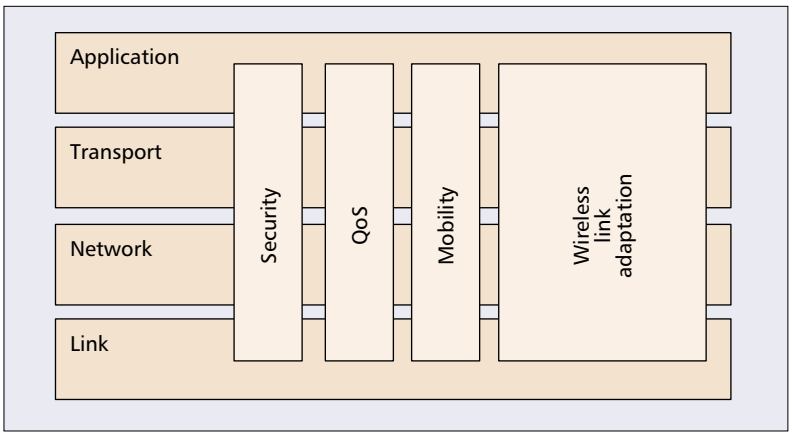


Figure 2. *Cross-layer coordination planes.*

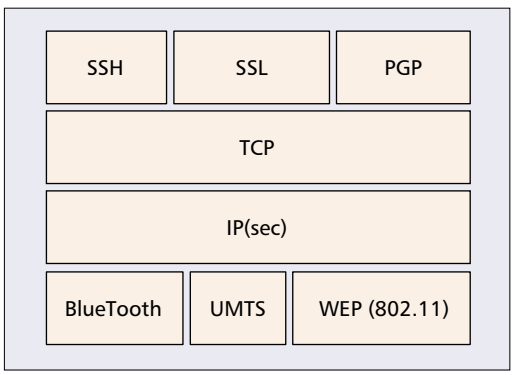


Figure 3. *Security plane.*

reviews known problems and presents a simple framework that is used not only to classify them but also to introduce elements that can be used in defining solutions in the years to come. For this purpose we introduce a set of coordination planes that are transversal to layers, and use them to review and classify problems. We review research work related to cross-layer design. We introduce a simple interlayer coordination model that takes into consideration the problems previously identified. Finally, we conclude the article.

Coordination Planes

In order to classify the problems and show the benefits of cross-layer design, the concept of a *coordination plane* is introduced. A coordination plane is a cross-section view of the protocol stack on which interlayer coordination algorithms can be applied, and it is focused on solving a set of problems of the same kind. Four coordination planes have been identified, as shown in Fig. 2:

- **Security.** The main purpose of this coordination plane is to eliminate multiple layers of encryption, which show up frequently in mobile communication systems.
- **QoS.** The QoS plane is responsible for distributing the QoS requirements and restrictions along the whole protocol stack, and coordinating their efforts.
- **Mobility.** This plane deals with the problems created by mobility scenarios, such as the poor interactions between TCP congestion control, Mobile IP, and layer 2 mobile solutions, as well as mobility across networks.
- **Wireless link adaptation.** This plane is used to represent the cross issues related to a given wireless link layer, and includes bit error rate (BER) and bit rate adaptability. This plane excludes mobility.

Security

Currently, the problem of security, more specifically encryption, can be solved by multiple layers. A scenario where multiple layers of encryption are used simultaneously is becoming common. Multiple layers of encryption may actually improve security, but have other disadvantages, such as the added processing power required, leading to waste of energy, and increased transmission delay. A number of encryption protocols and technologies are currently in use (Fig. 3):

- SSH, SSL, PGP. These protocols provide strong end-to-end encryption at the transport and application layers.
- IPsec, included also in IPv6, is a strong encryption method that can work either end-to-end between hosts, or as a secure tunnel between a host and a subnetwork, or two subnetworks.
- Wired Equivalent Privacy (WEP) is the encryption service provided by 802.11 wireless access network cards. WEP is an weak encryption scheme, since vulnerabilities have been discovered. Moreover, it only covers the access network.
- Bluetooth is another wireless access network technology that provides encryption. No significant security vulnerabilities have been found so far.
- Third-generation cellular systems, such as UMTS, also provide strong encryption techniques; both data and signaling are encrypted between the terminal and the serving radio network controller, wireless link included.

In order to avoid excessive processing of datagrams, thus saving power and reducing the delay, interlayer coordination should eliminate encryption features at all layers except one. The problem is deciding which layer should perform encryption, since some data flow, such as signaling, may be originated at intermediate layers.

Choosing, for instance, between IPsec and SSL/SSH/PGP, one has to consider the differences between these protocols. While IPsec provides encryption between hosts, SSL/SSH/PGP

provides encryption between a user and a service. In other words, the encryption is between applications instead of hosts. With SSL/SSH/PGP, the user receives better feedback on and control of the level of security for each connection, while with IPsec everything is done automatically by the operating system. Moreover, there could be multiple users and services communicating between a pair of hosts with different security domains and requirements, but IPsec does not (by default) differentiate between them.

On the other hand, link layer security protocols are either insecure or have local scope. This, in fact, may restrict their usefulness. Ideally, these encryption schemes should only be used to transmit datagrams for which no other encryption scheme is available. However, for some link devices, it may not be possible to turn encryption on and off on a per-packet or per-flow basis.

The bottom line on interlayer security coordination is that only one layer could be configured to perform encryption per data flow. Upper layer protocols seem to be preferred.

Quality of Service

Quality of service is important for 4G networks, especially those with limited resources. Effective QoS provisioning allows simultaneous use of multiple services with different requirements, ensuring that traffic from less demanding applications does not impact negatively on more sensitive traffic.

In order for QoS to effectively work, two conditions must be met:

1) QoS must be treated as an end-to-end issue, and solutions based on differentiated services (DiffServ) or integrated services (IntServ) models are often mentioned in the literature.
2) It must be handled by all the communication layers, since each layer may be required to provide a set of service guarantees.

From the QoS point of view, the protocol stack is composed of upper layer protocols (transport and above), such as application/TCP and application/RTP/UDP, on top of IPQoS. Applications can, in this context, be classified according to the data flows they exchange as elastic or real-time. An elastic flow can be extended in time and transports, for instance, a file by FTP or an HTML page; it is usually supported by TCP, which dynamically adapts the flow rate to the receiver window and network levels of congestion. In a real-time flow, the time of packet arrival at the receiver is relevant: if a packet containing a piece of voice misses a predefined deadline, it may be of no use for the receiver; a real-time flow is usually supported by RTP/UDP protocols. Recently, new types of real-time applications are appearing, are mainly related to audio and video transmission, and can use codecs that dynamically adapt their bit rates to the allocated channel rates. The IPQoS layer includes IP traffic control that implements datagram policing and classification, flow shaping, and scheduling. The link layer may also provide QoS support, by means of transmission priorities or virtual channels.

QoS solutions currently deployed are characterized by restricted communication between layers. Usually, the application is responsible for both creating the transport layer connections (RTP or TCP) and setting up QoS for such connections separately. The IPQoS layer uses QoS requirements to set up IP traffic control, but often this information is not passed to the link layer. Besides, each link layer has its own QoS characteristics.

This leads to the identification of a fundamental problem in currently deployed QoS subsystems: some layers in the protocol stack are not provided with QoS setup information and thus interact poorly with the QoS subsystem, instead of positively contributing to QoS. Some examples of such problems are given.

TCP: Given a constant bit rate allocation to a traffic flow, TCP knows nothing about such allocation. Due to its bandwidth probing and self-synchronizing nature, TCP constantly tries to increase its congestion window until packet losses are detected. One of two situations may occur, depending on whether the sender node is configured to perform policing or reshaping. In the former case, *nonconforming packets* (packets that, if transmitted, would cause the bandwidth reservation to be exceeded) are dropped. This has a negative impact on TCP performance, since it is perceived as a congestion signal, thus causing a reduction of the congestion window. In the latter case, reshaping, the result is also bad. With reshaping, nonconforming packets are queued in a buffer until they become conforming. In this instance, TCP perceives a network that can absorb all traffic it sends with no losses for some time, but the network appears to suddenly begin dropping packets at some point (because the reshaping buffer overflows), so the congestion window is severely reduced. The net result is increased transmission delay for some TCP segments. Moreover, because of the variable transmission delay, the TCP round-trip time (RTT) estimation algorithm does not work very well, with negative impact on TCP congestion control. In order to avoid these problems, there must be coordination between the QoS layer and TCP. In this way, with knowledge of the QoS reservation, TCP could adjust its congestion window in order to avoid sending nonconforming packets.

RTP: Given an RTP flow with a QoS reservation, it is possible that the link layer is performing automatic repeat request (ARQ) in a way that adversely impacts the obtained QoS. ARQ will often introduce jitter and delay. Unfortunately, real-time flows are very sensitive to delay and jitter, and usually it is preferable to simply drop erroneous packets than to attempt reliably retransmitting them using ARQ techniques. Therefore, some coordination between the QoS and link layers is required to overcome this problem. Using the maximum tolerable delay indicated by the application when it requests the QoS reservation, it should be possible, for instance, to reconfigure the *maximum retry limit* of link layer control (LLC) to prevent introducing excessive delay.

Packets from different flows need radically different treatment (e.g., TCP vs. RTP). This is especially true at the link layer. Therefore, the sending primitives offered by layer 2 must accept additional options to specify the transmission mode. Another approach would be for the link

Packets from different flows need radically different treatment (e.g., TCP vs. RTP). This is especially true at the link layer. Therefore, the sending primitives offered by layer 2 must accept additional options, to specify the transmission mode.

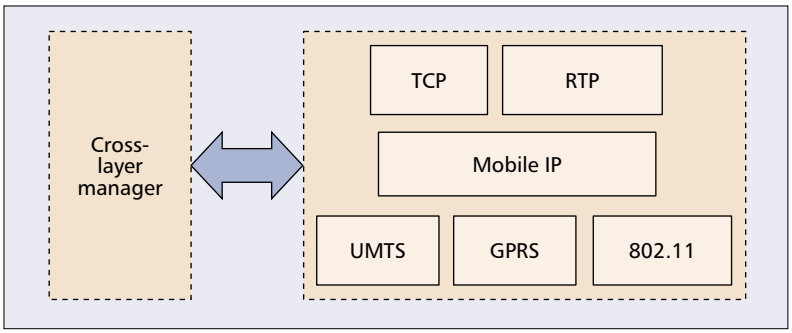


Figure 4. *Mobility plane.*

layer to implement the concept of a *virtual channel* or *context*. In this way, the QoS system creates a QoS context at QoS setup time with the desired attributes. This context is then referenced when transmitting each packet of the flow. This is the approach of some existing access network technologies like UMTS.

One of the greatest challenges in 4G is solving the problems associated with the low duration of terminal batteries. Although the problem is mostly hardware-related, intelligent use of resources by the protocol stack can be effective in reducing the amount of energy spent in data transmission. The general approach to better manage the scarce energy resources of mobile terminals is to take advantage of trade-offs. Usually, two kinds of trade-offs can be considered.

Power vs. delay: Consists in the ability to reduce the transmission power at the cost of an increase in transmission delay. This can be accomplished by constantly monitoring the interference level of the wireless medium, and delaying the packet transmission until the interference level drops below a certain threshold that permits safe transmission with less power [1].

Power vs. bit rate/BER: An alternative trade-off usually available in wireless networks lies in the ability to reduce transmission power by reducing the bit rate, without compromising the BER. Alternatively, it is also possible to reduce transmission power while maintaining the bit rate, but it has a cost in terms of increased BER.

However, these trade-offs cannot be used indiscriminately. The power module must be controlled so that the limits tolerated by each packet flow — in terms of delay, average bit rate, and BER — are not violated. With this type of interaction, energy savings can be achieved without adversely impacting real-time communications. Common 802.11 cards, for instance, can provide:

- Quality link, the general quality of the reception
- Quality level, the signal strength at the receiver
- Quality noise, the silence level (no packet) at the receiver that can be used in cross-layer design

Per-flow power control settings can be integrated into the link context, mentioned earlier.

Mobility

With the introduction of the concept of mobility in IP networks, a whole new set of problems emerged. In fact, the Internet was not designed with mobility in mind. As a consequence, the IP-based protocols do not handle mobility well.

The most glaring case is TCP. In a scenario where a mobile terminal is moving from one access point to another, the following happens to TCP connections:

- The link to the old access point begins to lose power, the BER increases gradually, and TCP begins to drop packets.
- At some point, the old link is lost, and the terminal tries to negotiate access to a new base station.
- Finally, the new link is established, and the TCP connection may resume.

In the scenario described above, there is a period of time when TCP packets are dropped.[1] Some of the consequences that can be identified include the following:

- The congestion window drops to a minimum value during handover, as it should, but is very slow to return to its original value once the link is reestablished. This happens due to the congestion avoidance stage of TCP congestion control [2], where the congestion window grows linearly with time.
- TCP keeps an RTT value that is used as a basis for computing the retransmission timeout (RTO). The RTT is automatically probed by TCP and updated over time. During an intertechnology handover, the new access link may present radically different characteristics, so it makes no sense to keep the old RTT value; it should be simply discarded so that a new, more accurate value is probed.

Cross-layer design is, once again, an approach to help solve these issues. A possible solution is outlined in Fig. 4.

Central to the protocol stack is the *Mobile IP* layer. It is designed to shield upper layers from the operational details of mobility support. For example, it automatically replaces the care-of address (CoA) for the home address (HA) in the destination field of received packets, and replaces the HA for the CoA in the source field of outgoing datagrams, and the upper layers continue to work transparently in the presence of mobility.

Unfortunately, shielding the upper layers from the knowledge of handover does not prevent them from experiencing its effects. It is important that the Mobile IP layer communicate handover events to the upper layers so that each protocol can decide how to best cope with them. For example, for TCP, and given explicit notifications of handover, the following optimizations could be employed.

Horizontal handover: This occurs when the terminal moves between adjacent access points of the same technology, and is expected to be of short duration. In this case, TCP connections could be frozen. In this way, after the handover is consummated, TCP can resume with the same congestion window value as before. Freezing a TCP connection consists of temporarily blocking the sending application and freezing the RTO timers as if the TCP clock had been stopped. As a result of this simple optimization, the TCP connection can resume much faster after the handover takes place.

Vertical handover: This occurs when the termi-

[1] *This time is always quite large for vertical handovers, whereas for horizontal handovers there are optimizations that can significantly reduce this time.*

nal moves between access points of different technologies. In this case, more caution is required, since the new access link does not have the same properties of the old one. In this situation, it makes more sense to revert the TCP congestion control algorithm to the slow start stage, quickly[2] probe the new link bandwidth, and throw away the previous RTT estimate in order to obtain a new value that is not influenced by past history, as well as discard the old slow start threshold.

Of course, for any of this to work, it is essential that the link layer(s) inform the Mobile IP layer of handovers, preferably before (and after) such handovers take place.

The QoS plane would also benefit from receiving handover notifications, especially if using a per-flow QoS model with explicit a priori reservations. With such a QoS model, it would be useful to tear down existing QoS reservations as soon as the handover begins, and set up new reservations from the new access router as soon as the handover is completed. Failing to do that, the QoS protocol will probably still adjust to the new path, but this process can take more time. If fast handover [3] (FHO) Mobile IP extensions are used, a further step could also be used. FHO works by letting the mobile node attach to the new access router before leaving the current one. This is usually described as "make before break," and allows a smoother and faster handover. During this process, the two access routers exchange FHO messages (Handover Initiate and Handover Acknowledge). In theory, it is feasible to attach a description of the existing QoS flows to the Handover Initiate message so that the new access router may start preparing its resources.

Wireless Link Adaptation

While the standard TCP/IP stack has worked well for wired links, it suffers from bad performance when used over wireless links. When compared with wired links, wireless links in general have lower bandwidths available, higher transmission delays, and higher BERs, and suffer from channel fading.

There is not much that can be done at the protocol stack level to work around the first two problems, and users learn to live with these limitations. Unfortunately, transport protocols suffer severely from the consequences of the last two problems. Again, the widely used TCP protocol is the primary example of this situation.

On one hand, erroneous datagrams are automatically dropped by the link layer, while TCP always interprets losses as a congestion signal. Thus, the TCP congestion control algorithm decreases (usually by half) the congestion window and enters the congestion avoidance state, where the congestion window grows linearly. The net result is a significant reduction in effective performance, which the user does not understand, knowing the advertised bandwidth of the network interface card.

On the other hand, wireless links often experience channel fading effects, consisting of fluctuation of the channel capacity over time. We can distinguish between slow and fast fading according to its duration, which is usually related to the speed of a mobile node's movement. Although fast channel fading has little impact on the performance of TCP, when slow channel fading occurs several consecutive TCP packets are dropped [4]. Thus, TCP congestion control will inevitably and quickly lower the congestion window to its minimum value. Unfortunately, due to the *multiplicative-decrease additive increase* property of congestion avoidance [2], the congestion window is very slow to return to its original value before fading occurred. Once more, the net result is a decrease in effective throughput and, consequently, underutilization of radio resources.

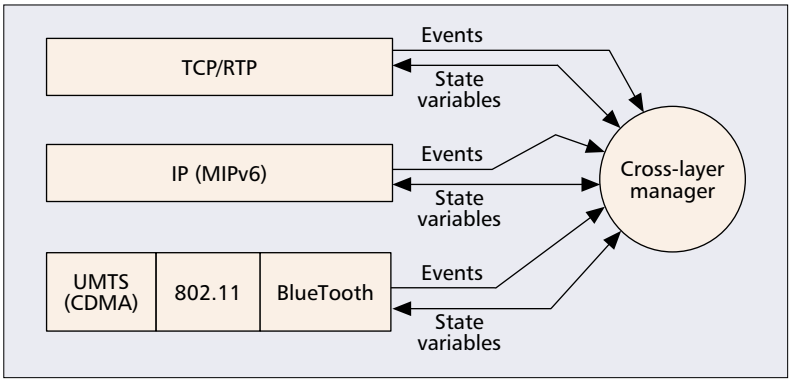


Figure 5. *Interlayer coordination model.*

One way to work around the problem of high BER in wireless links is to resort to link layer ARQ. Although TCP also performs the ARQ function end-to-end, link layer ARQ is always better because of the tighter control loop and reduced overhead. However, some inherent issues interlayer coordination could solve include the following:

- Both TCP and the link layer ARQ compete with each other. For this reason, TCP's ARQ mechanism should be coordinated with layer 2 ARQ. Without this coordination, it is possible that LLC is still trying to retransmit a packet for a long time, and TCP's RTO expires, so TCP decides to retransmit the packet too. Eventually, layer 2 succeeds in transmitting the packet without error, but TCP has another copy queued to be sent.
- The ARQ strategy may be harmful for non-TCP traffic, or at least may need different parameters. Cross-layer design is required to dynamically reconfigure layer 2 ARQ considering the type of traffic being transmitted. Input from the QoS coordination plane is valuable in helping make such management decisions.
- Consider the scenario where a mobile terminal is transmitting two flows, one TCP and one RTP, sharing a single wireless interface. Sometimes, it may happen that layer 2 is trying repeatedly to transmit a TCP segment, and at the same time the real-time flow may be trying to send a datagram with stringent timing constraints. The TCP flow effectively interferes with the real-time one, since it is monopolizing the link layer, thus deteriorating the QoS of other flows. To prevent this situation, a *preemptive* link layer is required. In a preemptive link layer, the ARQ-enabled transmission of any packet can be preempted by another packet with higher priority. This way, we get the benefit of link layer ARQ with a very large long retry limit (LRL)[3] without interfering with real-time flows.

[2] *In spite of its name, slow start is actually a very fast method of probing the link bandwidth, since it increases the congestion window exponentially, as opposed to a linear increase during congestion avoidance.*

Protocol/module	Events	State variables
TCP	Connection initiated Connection terminated Acknowledgement timeout	Congestion window (cwnd) Retransmission timeout (RTO) Round-trip time (RTT) Slow start threshold (SSthreshold)
IP	Handover start Handover end Routing table updated QoS reservation completed QoS Modification	Routing table Security associations List of QoS-enabled flows Per-flow QoS
Link	Link lost Fading start Fading end Corrupt packet received	Capacity (bandwidth), delay, BER Long retry limit Transmission power Noise/interference level

Table 1. *Exposed interfaces for interlayer coordination.*

•When using layer 2 ARQ with TCP, one has to consider the side effects. For example, ARQ introduces an artificial delay in the link, making TCP estimate an inflated value of RTT. The RTT estimated by TCP is used to scale the growth of the congestion window. Moreover, TCP does not work well with an unstable RTT. Therefore, TCP must be kept informed of some details of the layer 2 ARQ so that it can compensate the RTT estimated value.

The work described in [5] further explores these ideas. Regarding the problem of channel fading, cross-layer design is, again, able to provide a solution. In [6] the TCP-Sleep protocol is discussed. The main novelty of this protocol is the addition of a sleep state to TCP. TCP enters the sleep state by indication of the link layer. During that state, TCP stops trying to transmit segments and suspends the congestion window algorithm. When the link layer signals that fading is over, TCP resumes normal operation.

Past Work

In [7] a proposal for adding different levels of error protection depending on application level protocol is presented. It also describes buffering and retransmission in case of error, looking at transport level headers (TCP) to detect errors. Other optimizations for wireless devices include header compression, application-adaptive ARQ, and priority-based scheduling. The Snoop protocol is described in [8]; it basically performs caching and retransmission in case of error, looking at transport level headers (TCP) to detect errors. In [9] multiple layers are proposed to simultaneously process data at the CPU level. Reference [10] attempts to quantify the overall data transmission rate as a function of the wireless channel BER for various values of the MAC LRL; it also compares TCP with MAC layer fragmentation. Reference [11] presents a method for controlling CPU frequency and voltage to save power, triggered by application layer media type. Reference [12] discusses how to employ different types of link layer optimizations, depending on the type of packet being transported; within this work, each layer exports performance measurements and events to the layer above it. Reference [13] discusses cross-layer design for solving common networking problems, including mobility, energy saving, QoS, and ad hoc networks. Reference [14] discusses how cross-layer design techniques can be used to propagate the current conditions of time-varying channels to upper layers in order to improve overall performance.

[3] *The LRL is the medium access control (MAC) layer retransmission limit of 802.11 cards.*

Interlayer Coordination Model

One possible model for interlayer coordination consists of a set of modules (protocols) connected to a central interlayer coordination manager. The modules expose *events* and *state variables* to the manager. Events are notifications sent to the manager, such as *handover begins* or *link lost*. They are used to trigger, or "wake up," the management algorithms. State variables represent entry points to get/set operations that allow the manager to query or modify the internal state of a protocol/module. Figure 5 exemplifies how events and state variables from each module can be used by the coordination manager to implement management algorithms.

Table 1 lists the events and state variables (control points) that need to be exposed by protocols to be used by management algorithms. The problems that have been identified in this document are presented below.

Security Plane

1. *Multiple layers of encryption may improve security, but require more processing power, leading to waste of energy and an increase of transmission delay.* Solution: Disable encryption on all layers except one.

2. *Encryption by upper-layer protocols is preferable to lower-layer ones. Flows generated by intermediate layers, such as signaling, shall use encryption at the highest layer they can.* Solution: For each datagram or data unit, keep a flag indicating if it has already been encrypted by an upper layer protocol.

QoS Plane

1. *TCP does not adjust itself to the QoS reservation, causing reshaping buffers to overflow. As a result, some packets are unnecessarily dropped and retransmitted, wasting energy and increasing delay, or simply delayed in reshaping buffers.* Solution: Control the TCP congestion window using the flow QoS information so that TCP never attempts to transfer more data than permitted by the QoS reservation.

2. *Link layer ARQ may adversely impact the QoS of real-time flows.* Solution: The LRL must be dynamically adjusted to the QoS settings (specifically the maximum delay) of the datagram being transmitted.

3. *Although with some power controlled devices trade-offs of power vs. delay and power vs. bit rate/BER are possible, they cannot be used without knowledge of the QoS treatment a packet expects.* Solution: The transmission power must be dynamically adjusted to the QoS settings of the datagram being transmitted.

Mobility Plane

1. *During a handover, the TCP congestion window goes to a minimum value, but is very slow to recover after the handover is complete.* Solution:

Using handover notifications, freeze the TCP connection and state during handover.

2. *During a vertical handover, TCP does not update its RTT value, even though the terminal might have changed to a new networking technology with a completely different transmission delay, thus making the RTO value suboptimal for the new link type.* Solution: Using handover notifications, freeze the TCP connection and state during vertical handover, but force the RTT to be recomputed when the handover is completed.

3. *Upon handover, it is possible that some QoS flows need to have their reservation updated, but the QoS module is not aware of such handover.* Solution: Use the *Handover end* event notification to trigger QoS refresh for all QoS-enabled flows.

Wireless Link Adaptation Plane

1. *Wireless links suffer from a much higher BER than wired ones. Erroneous or dropped TCP packets cause the congestion window to be reduced, thus decreasing the effective transfer rate of the TCP connection.* Solution: Use link layer ARQ with a suitably large LRL for TCP packets.

2. *Channel fading has a similar effect as a handover from the mobility plane; it causes the TCP congestion window to drop to a minimum value and recovers only very slowly when the fading ends.* Solution: Use *fading start* and *fading end* notifications from the link layer (if available) to temporarily freeze the TCP connection and state.

3. *Link layer ARQ as a means to overcome the BER limitation of wireless links may introduce additional problems:*

- *Competition between link layer ARQ and transport layer (TCP) ARQ.* Solution: Disable TCP retransmission while link layer ARQ is taking place.
- *Increased delay for real-time traffic.* Solution: Dynamically adjust the LRL considering the packet's QoS requirements.
- *Blocking real-time traffic while retransmitting TCP traffic.* Solution: Make the link layer preemptive, adding the ability of real-time flows to interrupt link layer ARQ of TCP datagrams.
- *Confusing the RTT estimation algorithm of TCP.* Solution: Introduce corrective factors into TCP's RTT with information from the link layer.

Conclusion

This article presents an overview of some important problems faced by all-IP wireless mobile terminals, often referred to as fourth-generation terminals. A simple framework for studying and solving these problems with cross-layer design has been presented.

This framework first classifies known problems in four coordination planes: security, QoS, mobility, and wireless link. Security problems arise from multiple-layer encryption, which causes unnecessary power consumption and processing delay. QoS problems affect flows with QoS requirements, and are caused by lack of information from transport layer congestion control and link layer ARQ. The mobility problems are related to the effects of handover on transport layer connections and QoS signaling. Finally, wireless problems are caused by packet corruption and losses that are perceived by TCP as congestion indications, causing it to have poor performance.

In order to help solve these problems, a simple interlayer coordination model was presented, consisting of a cross-layer manager that receives event notifications from each protocol and performs management algorithms that control the internal state of each protocol to correct their behavior with information from other layers. Future work will include the development of management algorithms.

One possible model for inter-layer coordination consists in a set of modules (protocols) connected to a central inter-layer coordination manager. The modules expose events and state variables to the manager. Events are notifications sent to the manager, such as "handover begins" or "link lost."

References

[1] N. Bambos and S. Kandukuri,"Power-Controlled Multiple Access Schemes for Next-Generation Wireless Packet Networks," *IEEE Wireless Commun.*, June 2002.

[2] V. Jacobson, "Congestion Avoidance and Control," *ACM Comp. Commun. Rev., Proc. Sigcomm '88 Symp.*, Stanford, CA, Aug., 1988, pp. 314–29.

[3] Rajeev Koodli *et al.*, "Fast Handovers for Mobile IPv6," Internet draft, Oct. 10, 2003.

[4] A. Chockalingam, M. Zorzi, and R. Rao, "Performance of TCP on Wireless Fading Links with Memory," *Proc. IEEE ICC '98*, June 1998.

[5] N. Vaidya *et al.*, "Delayed Duplicate Acknowledgments: A TCP-Unaware Approach to Improve Performance of TCP over Wireless," Tech. rep., Comp. Sci. Dept., Texas A &M Univ., 1999.

[6] V. Chandrasekhar, "Improving the Performance of TCP over Lossy Links," Dec. 2001.

[7] Q. Zhang, W. Zhu, and Y.-Q. Zhang, "A Cross-layer QoS-Supporting Framework for Multimedia Delivery over Wireless Internet," *Int'l. Packet Video Wksp.*, 2002.

[8] H. Balakrishnan *et al.*, "Improving TCP/IP Performance over Wireless Networks," *Proc. Mobicom*, Nov. 1995, pp. 2–11.

[9] M. B. Abbott and Larry L. Peterson, "Increasing Network Throughput by Integrating Protocol Layers," *IEEE/ACM Trans. Net.*, 1993, pp. 600–10.

[10] K. F. Dombrowski *et al.*, "Vertical Optimization of Data Transmission for Energy Aware Mobile Devices," *Int'l. Conf. Wireless LANs and Home Networks*, 2001.

[11] W. Yuana *et al.*, "Design and Evaluation of a Cross-Layer Adaptation Framework for Mobile Multimedia Systems," *Proc. SPIE/ACM Multimedia Comp. and Net. Conf.*, Santa Clara, CA, Jan. 2003, pp. 1–13.

[12] G. Xylomenos and G. C. Polyzos, "Link Layer Support for Quality of Service on Wireless Internet Links," *IEEE Pers. Commun.*, vol. 6, no. 5, 1999, pp. 52–60.

[13] A. Goldsmith *et al.*, "A Multilayer Approach to Mobile Networking," *SNRC Project Wksp.*, June 7, 2001

[14] S. Shakkottai, T. S. Rappaport, and P. C. Karlsson "Cross-Layer Design for Wireless Networks," *IEEE Commun. Mag.*, vol. 41, no. 10, Oct. 2003.

Biographies

GUSTAVO CARNEIRO (gjc@inescporto.pt) received a Diploma degree in electrical and computer engineering from Porto University, Portugal, in 2001 and since then has been a researcher at INESC Porto. He actively participated in the European IST ARROWS project and currently is participating in the European DAIDALOS project.

JOSÉ RUELA (jruela@inescporto.pt) received a Diploma degree in electrical engineering from Porto University, Portugal, in 1971, and a Ph.D. degree in electrical engineering from the University of Sussex, United Kingdom, in 1982. He is an associate professor at Porto University, where he gives courses in data communications and computer networks, and is manager of the Telecommunications and Multimedia Unit at INESC Porto, an R&D institute affiliated with Porto University. His main research interests are resource management, QoS, and performance evaluation in high-speed networks.

MANUEL RICARDO (mricardo@inescporto.pt) received a Diploma degree in 1988, an M.S. in 1992, and a Ph.D. in 2000, all in electrical and computer engineering, from Porto University, Portugal. He is an assistant professor in the Faculty of Engineering, Porto University, where he gives courses in mobile communications, and computer networks. He also leads the Communication Networks and Services Area at INESC Porto.