

Cross-Layer Design Tutorial

Frank Aune - faune@stud.ntnu.no

Norwegian University of Science and Technology,
Dept. of Electronics and Telecommunications,
Trondheim, Norway.

Published under Creative Commons License.

26.11.2004

Acknowledgement

I would like to thank the following people for their valued feedback and cooperation: Vegard Hassel, Anna Kim and Hend Kouba.

Contents

1	Introduction	1
1.1	Historical Background	2
1.2	The Layered Design of the OSI Reference Model	3
1.3	Real-Time Applications	4
1.4	Adapt and Evolve the IP and its Control Protocols	4
1.4.1	Quality of Service (QoS) and Routing Tables	5
1.4.2	Type Of Service (TOS) bits	5
1.4.3	Alternative Control Protocols	5
1.5	Wireline versus Wireless Networks	6
1.6	Summary	6
2	Cross-Layer Design - Collaboration Across Borders	8
2.1	Cross Layer Design Definition	9
3	Cross-Layer Design: Evolution or Revolution?	10
3.1	The Evolutionary Approach to CLD	10
3.2	The Revolutionary Approach to CLD	11
4	Cross-Layer Design Examples and Implementations	12
4.1	Evolutionary Cross-Layer Design Examples	12
4.1.1	Example 1: Layer Triggers	12
4.1.2	Example 2: EventHelix' Protocol Design	13
4.1.3	Example 3: MobileMan	15
4.1.4	Example 4: Joint Source Channel Coding	17
4.1.5	Example 5: Designing a Mobile Broadband Wireless Access Network	18
4.2	Revolutionary Cross-Layer Design Examples	19
4.2.1	Example 6: Wireless Sensor Networks	19
4.2.2	Example 7: JSCC Shannon Mappings	20
5	Cross Layer Design Skepticism	22
6	Conclusion and Future Work	24
7	Appendices	26
7.1	Appendix A: OSI quick reference model overview	26
7.2	Appendix B: TOS-bits	27

Abbreviations

Alphabetical overview of abbreviations

3G - 3rd generation mobile network (UMTS)
4G - 4th generation mobile network (in planning)
BER - Bit Error Rate
CDMA - Code Division Multiple Access
CLD - Cross Layer Design
ECN - Explicit Congestion Control
FIFO - First In First Out
IEEE - Institute of Electrical and Electronics Engineers
IEEE 802.11 - Based on CSMA/CA
IP - Internet Protocol
ISO - International Standardization Organization
JSCC - Joint Source Channel Coding
LAN - Local Area Network
LL - Link Layer
MAC - Medium Access Control
MANET - Mobile Ad Hoc Network
MIMO - Multiple Input Multiple Output
MUD - MultiUser Diversity
OFDM - Orthogonal Frequency Division Multiplexing
OSI - Open Systems Interconnection
QoS - Quality of Service
RC - Rate Controller
RTP - Real Time Protocol
SSCC - Single Source Channel Coding
TCP/IP - Transfer Control Protocol / Internet Protocol
UDP - User Datagram Protocol
UMTS - Universal Mobile Telecommunications System
VoIP - Voice over IP
WLAN - Wireless Local Area Network
WSN - Wireless Sensor Networks

Abstract

Cross-Layer Design has recently become the new hype in wireless communication systems. This tutorial is to examine the current research activities in the area of Cross-Layer Design. The evolutionary and revolutionary approach to the Cross-Layer Design principle are studied. Examples such a 4G Mobile Broadband Wireless Access system and system-wide Cross-Layer Design enabled network stack are shown in the evolutionary approach; whereas Wireless Sensor Networks was discussed in the revolutionary approach case. Like any other new trend in science and engineering, preliminary precaution should always be exercised. Skeptisms regarding Cross-Layer Design were also included. In conclusion, Cross-Layer Design has a great potential in future wireless communication systems.

Chapter 1

Introduction

One of the most visible trends in today's commercial communication market, is the adoption of wireless technology [1]. Today you can purchase just about any electronics device, and it will probably contain some sort of wireless technology. From Bluetooth enabled mobile devices, cameras and printers to Wireless Local Area Network (WLAN) enabled computers, cars and handheld devices. Together with the explosive growth of the Internet, we clearly see a shift in the market towards tighter integration and seamless wireless connections.

Traffic carried by future wireless networks is expected to be a mix of real-time traffic such as voice, multimedia conferences and games, and data traffic such as web browsing, messaging and file transfer. All of these applications will require widely varying and very diverse Quality of Service (QoS) guarantees for the different types of offered traffic, and we are now in the early days of this eventual merging [2].

This report is a tutorial about the recent emerging paradigm shift that is beginning to take place as wireless communications are evolving from circuit switched to packet switched infrastructure. The paradigm shift evolves around a principle known as Cross-Layer Design (CLD), and this tutorial will try to highlight and focus on some of the topics involving CLD and why it can potentially solve some of the problems in wireless communication today.

We start off with some historical background around network stack design, and look at some common problems for networked communications. The introduction of realtime applications and wireless communication pushed the boundaries even further, and we will look at some of the steps taken to accommodate this challenge. We then shift our focus to wireless communications, where the previously implemented solutions are not sufficient to maintain a given Quality of Service (QoS) and thus we are today forced to seek other approaches. After a quick overview of the principles and ideas behind CLD, a number of proposed and implemented CLD systems are described and reviewed. We will also propose that the CLD phenomena is not only related to networked communications, but can also be applied and taken benefit of in scenarios most never thought of. To balance the scale, we also try to focus on some of the negative aspects brought up by critics of a CLD approach. However, as we shall see, a paradigm shift is always painful - both for end-users and commercial actors.

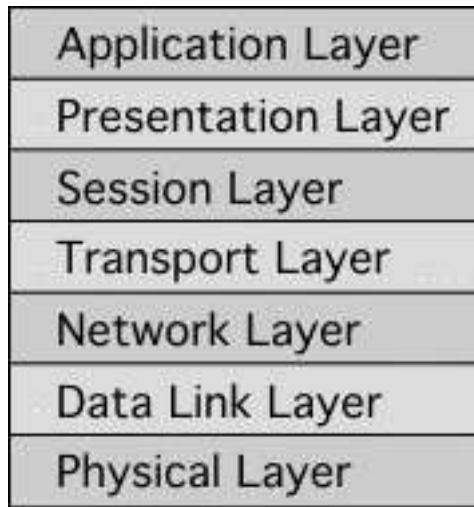


Figure 1.1: The OSI reference model

1.1 Historical Background

The International Organization for Standardization (ISO) began in the early 1980s to work on an open set of protocols that would enable multivendor computers to interact and communicate with one another. This work eventually led to the design of the Open Systems Interconnection (OSI) network stack, which was sought to become *the* building block of all network based communication worldwide. The OSI model was by many considered as the ultimate model for worldwide interoperability, however as time has showed, the OSI model remained just that: A model by which other implementations are compared against.

The predecessor to the Internet, the ARPANet, was developed by the US Defence Dpt. in the 1960's and was based on the Internet Protocol (IP). Specifications and interoperability were created second to allow others to participate in the network. While using the standardized OSI protocols undoubtedly would have been a good idea at that time, the IP implementation was easy, quick and cheap. The IP allowed the Internet to grow rapidly at the grassroots level. As the governments around the world started participating in the Internet growth, it was expected that the IP was to be replaced by the OSI protocols, however this never happened.

The ISO OSI model remains today as a reference model, by which other implementations are compared to and it describes and outlines the different levels of networking protocols and their relationship with each other. The OSI model consist of 7 layers as shown in Figure 1.1. A quick reference overview of the different layer tasks is provided in Appendix A.

The OSI 7-layer reference model is an attempt to abstract features common to all approaches in data communications, and organize them into layers or modules such that each layer only worries about the layer directly above it and the one directly below it. As we shall see throughout this tutorial, a rigid and strict layer definition turns out to be perhaps the most fundamental identity of

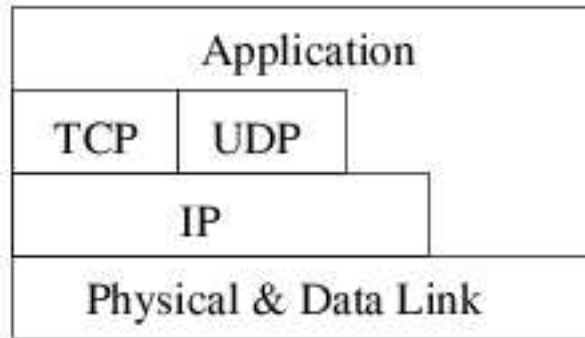


Figure 1.2: TCP/IP and UDP in a network stack

the traditional design approach.

The most fundamental challenge in any network reference design, is how to allocate the available resources among the different network users. The traditional approach to network stack design has always been to treat the different layers as separate entities, and then perform layer specific operations on these entities to achieve an operational network stack with acceptable performance. Since the main goal of the OSI model was to allow multivendor computers to interact and communicate (transmit pure data traffic), the term QoS was not thought off and thus not an issue in the design process. Resource handling and sharing followed a First-In First-Out (FIFO) pattern, or perhaps better described as a "best-effort" service. To minimize congestion in the network, each network entity adapted its transmission rate accordingly [2], and this worked for the most part. The layered network stack approach has been extremely successful over the last three decades, and the design principles have been widely adopted throughout various implementations and applications worldwide.

1.2 The Layered Design of the OSI Reference Model

Transfer Control Protocol/Internet Protocol version 4 (TCP/IPv4) is today the most successful implementation of the OSI reference model, and Figure 1.2 shows how these protocols relate to the layered network stack . Since TCP/IP is loosely based on the layered design of the OSI reference model, it also inherits its potential flaws and weaknesses. First of all, the stack design is highly rigid and strict, and each layer worries only about the layer directly above it or the one directly below it. This results in a non existant collaboration between the different layers, presumeably because no-one at that time saw any need for such a feature. Of course, you can always argue that one of the most important steps in any architectural design is to first identify processes or modules in the system and abstract them by assign roles or requirements. To sum it up; the OSI reference model made perfect sense as a building block at that time. An interesting note, however, is that the the TCP/IP was never ment to be a

”design once - fit all” type of protocol, but as time has moved on, the TCP/IP protocols have nonetheless emerged as the de-facto standard for packet based point-to-point communication today.

1.3 Real-Time Applications

In the last few years, the Internet throughput, usage and reliability have increased almost exponentially. The introduction of broadband wireless mobile ad hoc (manet) and cellular networks, together with increased computational power have opened the door for a new breed of applications to be created, namely real-time multimedia applications. Examples of modern real-time applications are multimedia videoconferences, Voice over IP (VoIP) and various other types of multimedia streaming services. Requirements for real-time communication are [3]:

- Adaptability to dynamically changing network and traffic conditions
- Good performance for large networks and large numbers of connections
- Modest buffer requirements within the network
- High effective capacity utilization
- Low overhead in header bits per packet
- Low processing overhead per packet within the network and at the end system,

These requirements are very difficult to fulfill in any IP based network [3], and when real-time applications were first developed, they quickly revealed some of the more serious quirks of the Internet implementation. Not only do datagrams need to reach its destination in an extremely short period of time, they also need to reach the destination in a more or less the correct order. In a real-time application, data with a timestamp over a certain set value are therefore useless. Calculated ”features” such as packet loss, retransmission, packet congestion, routing problems, reassembly trouble and timeout were highly common, and obviously not well suited for real-time application deployment. Real-time applications in a manet or cellular network, relying on packet switched infrastructure, proved to be problematic - at best. The calculated ”features” just amplified in such environments due to time varying channel conditions. Naturally one of the main goals for the network engineers and researchers became how to improve the real-time application support in a packet switched manet or cellular network infrastructure.

1.4 Adapt and Evolve the IP and its Control Protocols

The financial investments in both the Internet topology and IP have been so huge over the past decades, introducing something ”revolutionary” new is not even an option. The key feature became backwards compability. In this section

we will briefly describe some of the features implemented in IPv4 which tried to address the problems described above with regards to real-time multimedia applications in packet switched networks.

1.4.1 Quality of Service (QoS) and Routing Tables

As of IPv4 QoS and routing tables in the network routers were implemented, and different types of traffic were given different priority. Normal data traffic was delayed or even dropped in favour of real-time traffic. Although QoS is not expected to solve the problem on its own, it has received a great deal of attention in the last few years, and it will become a key design component of any communication system.

1.4.2 Type Of Service (TOS) bits

These bit flags are called Type of Service bits because they enable the application transmitting the data to tell the network what type of network service the application requires. When any one of these bit flags is set in the IPv4 header, routers may handle the datagram differently than packets with no TOS bits set. Each of the four bits has a different purpose and only one of the TOS bits may be set at any time, so combinations are not allowed. The different service classes available are:

- Minimum delay
- Maximum throughput
- Maximum reliability
- Minimum cost

Please refer to appendix B for a brief explanation of the different classes.

1.4.3 Alternative Control Protocols

Transfer Control Protocol (TCP)

The most widely used transport protocol today is TCP, however a number of features makes it unsuitable for real-time applications. TCP is a point-to-point protocol that sets up a connection between two endpoints using handshake, and as a result it can not be used in a multicast environment. TCP includes mechanisms for retransmission of lost segments, which would naturally arrive out of order. However, lost segments are usually not usable in any real-time application. Also TCP contains no mechanism for associating timing information with segments. All of these features makes TCP better suited for pure point-to-point data transmissions.

User Datagram Protocol (UDP)

Another widely used transport protocol is UDP, although UDP does just about as little as a transport protocol can. Aside from the multiplexing and demultiplexing function and some light error checking, it adds nothing to IP. In fact, if

the application developer chooses UDP instead of TCP, the application is talking almost directly with IP. Although commonly done today, running real-time multimedia applications over UDP is controversial to say the least. UDP lacks any form of congestion control and timing information. Congestion control is needed to prevent the network from entering a congested state in which very little useful work is done. If everyone were to start streaming high bit-rate video without using any congestion control, there would be so much packet overflow at routers that no one would see anything. Thus, the lack of congestion control in UDP is a potentially serious problem. Many researchers have proposed new mechanisms to force all sources, including UDP sources, to perform adaptive congestion control [4], however the problem remains to be solved.

Real-time Transport Protocol (RTP)

RTP resolve many of the shortcomings of TCP and UDP when used together with real-time applications. It is designed to provide end-to-end network transport functions for applications transmitting real-time data, such as audio, video, or simulation data, over multicast or unicast network services. RTP provides services such as payload type identification, sequence numbering, time-stamping, congestion control, reconstruction and delivery monitoring to real-time applications making it highly suited for real-time traffic.

DIFserve

Where TCP is ideal for pure point-to-point data traffic and RTP is ideal for real-time traffic, DIFserve tries to be a multipurpose control protocol which uses the TOS-bit in the IP header to identify and mark different types of packets, and thus being able to provide different QoS classes for different type of data.

1.5 Wireline versus Wireless Networks

Some of the characteristics of a wireless system are quite different than those of a wireline system, because a wireless system have limited frequencies and channels available for transmission. **Small-scale channel variations** due to fading, scattering and multipath can change the channelstate from "good" to "bad" within a few milliseconds, while **large-scale channel variations** are usually slow and depends on user location and interference levels from the surroundings [5]. This is the reason why designing Mobile Adhoc NETworks (MANET) or cellular networks supporting data and real-time traffic using a packet switched approach is very difficult. Of course small- and large-scale channel variations also exist for circuit switched networks, but because users monopolize their respective time-slots regardless of usage this is not suited for bursty communication.

1.6 Summary

There is no denying today that the Internet has been a tremendous success, mainly determined by a layered architecture. A natural extension of this trend has been to apply a similar layered architecture when dealing with manet and

cellular networks [6]. As pointed out in this tutorial so far, there are several key issues with the IP and its transport protocols which makes them challenging to use in manet and cellular environments. As suggested by [7] short term fast fading effects should be fixed locally within a layer, while long term fading effects needs to be adressed to the application layer. This goes to show that a strict layered design is perhaps not flexible enough to be used in such dynamic environments.

Cross-layer protocol interactions, when used appropriately, can lead to increased network efficiency and better QoS support. A Cross-Layer Design (CLD) is particularly important for any network using wireless technologies, since the state of the physical medium can significantly vary over time. Perhaps information exchange between different layers can even optimize the network throughput.

The next part of this tutorial will focus extensively on the CLD approach for solving the previously described problems, using both theoretical and practical implementations of CLD in manet and cellular environments.

Chapter 2

Cross-Layer Design - Collaboration Across Borders

As mentioned previously, the traditional way of designing a wireless manet or cellular network architecture, has been to identify each process or module and then assign them roles or requirements. Since each process or module has been treated separately, this approach has in many ways caused the research communities to split into different groups, where each group focus their resources on solving "their" problem the best possible way. What other research communities are doing, is not really important, as long as the job is done. This is of course a bit exaggerrated, but none the less illustrates the problem in an efficient manner.

As shown in Figure 2.1, the CLD approach to network architecture is located where the three communities intersect. Listed below are some of the fields the different research communities traditionally have focused on solving:

Wireless networking:

- Architecture: connection versus connectionless
- Energy efficient analysis of manets
- Scaling laws of large scale networks
- Traffic theory
- Protocols

Signal processing:

- Increasing spectral efficiency (bits/s/Hz)
- Reducing Bit Error Rate (BER)
- Reducing the transmission energy
- Detection and estimation algorithms for multi-access

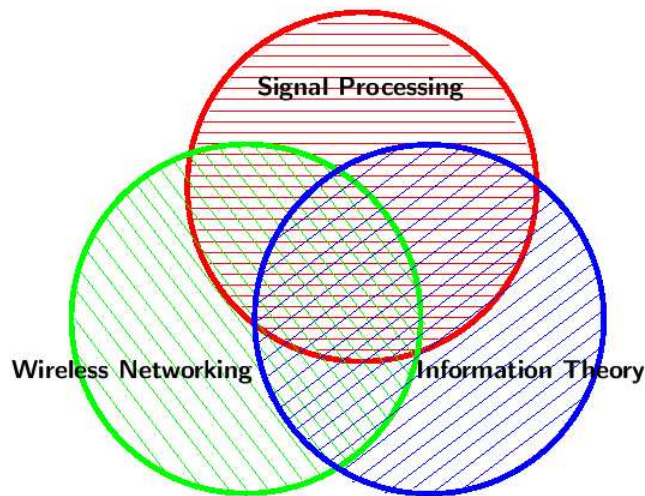


Figure 2.1: Collaboration across borders

Information theory:

- Developing capacity limits
- Designing efficient source coding and channel coding algorithms

The whole idea behind CLD is to combine the resources available in the different communities, and create a network which can be highly adoptive and QoS-efficient by sharing state information between different processes or modules in the system.

2.1 Cross Layer Design Definition

To fully optimize wireless broadband networks, both the challenges from the physical medium and the QoS-demands from the applications have to be taken into account. Rate, power and coding at the physical layer can be adapted to meet the requirements of the applications given the current channel and network conditions. Knowledge has to be shared between (all) layers to obtain the highest possible adaptivity.

Chapter 3

Cross-Layer Design: Evolution or Revolution?

The CLD approach to network stack design is historically a big shift in how one designs a communication system. Not only does the applications, protocols and hardware need to be reimplemented to be able to support the new extensions, but the whole concept of CLD challenges everything engineers and researchers know about network protocols, layers, stack design and system construction. While CLD might seem revolutionary instead of evolutionary at first, there are implementations available today which tries to incorporate some of the key elements of the CLD philosophy into existing protocols and layers.

3.1 The Evolutionary Approach to CLD

An evolutionary approach to CLD always seeks to extend the existing layered structure, in order to maintain compability. Note however, that there is a big difference between the evolutionary basic and the evolutionary system-wide approach which range from [6]:

- simple, yet effective, solutions which extends parts of the strict layering structure
- to system-wide CLD where stackwide layer interdependencies are designed and implemented to optimize overall network performance.

Most CLDs today are evolutionary, because compability with existing systems and networks is extremely important both for end users and commercial actors. The simple, yet effective, CLD solutions are by far the most common today. The reason for this is quite simple: Since an evolutionary CLD is always bound by its original strict layered structure, an extension of this will also always be limited. In most cases however, there is no need for an over-complex CLD implementation to be able to solve the problem at hand. Most often only two or perhaps three layers need to share information, and thus its also reasonable to extend the original strict layered structure.

But what if you were to extend the CLD implementation to include all layers? Is it even possible to keep such an implementation reasonably sane, given that

any evolutionary CLD is bound by its original implementation? In example 3 we take a look at the MobileMan-project which tries doing exactly this.

3.2 The Revolutionary Approach to CLD

A revolutionary approach to CLD, or any design for that matter, is not bound by an existing implementation, and as such does not need to compromise to maintain compability. Where an evolutionary CLD approach prioritize compability first and performance later, a revolutionary design does the opposite. There are many reasons, the main being compability and economy, why a revolutionary approach is not favoured, not only in CLD but pretty much any research or engineering task. How will you communicate with the rest of the world if your system is no longer able to speak with other IP networks? If this problem remains unsolved, you will probably have a hard time getting commercial parties interested in your product. A long time-to-market delay also makes it difficult justifying research on revolutionary ideas.

Since most CLDs today are evolutionary, its hard to find examples of a revolutionary CLD. A revolutionary approach can however be applied to highly specific problems where backwards compability is not important. In example 6 we take a look at the Wireless Sensor Network (WSN) environment, where this approach can be applied. In example 7 we take a look at Shannon mappings, which also is CLD in a revolutionary sence. There are probably other examples where a revolutionary approach can be applied. These results and ideas might give us a new understanding of a specific problem and maybe applied to existing evolutionary implementations as well. History has shown many times, that keeping an open mind about new and revolutionary ideas is always important.

Chapter 4

Cross-Layer Design Examples and Implementations

In this section we will look at some of the proposed and implemented systems using CLD today. Some are still on the drawing-board, while others are being used today in real world scenarios.

4.1 Evolutionary Cross-Layer Design Examples

4.1.1 Example 1: Layer Triggers

Layer triggers are predefined signals which are used to notify special events between protocols. The Explicit Congestion Notification (ECN) bit described in this example is such a layer trigger. Another example is something called L2 triggers, which are internal stack signals added between the Logical Link (LL) and IP layer. L2 triggers allow changes in the wireless link to be efficiently detected and shared among these two layers, thus the name L2. One can even argue that the TOS-bit described in section 1.4.2 is a layer trigger variant. Layer Triggers are the most basic CLD implementation, mainly because:

- It is cheap and quick to implement
- It has quantifiable performance improvements as shown in [8] and [9]
- You can maintain compatibility by extending on the strict layered structure

Because of this, layer triggers are today extensively used in both wired and wireless networks [6]. While not present in most deployments of the TCP, a modern TCP implementation today contains the ECN-bit. This was first proposed and discussed in 1994 by [10], and have since become a reality. The ECN-bit is used to notify the receiver whenever packet congestion occurs in the network. In the TCP header, the ECN bit is always set to zero by the application. If a network router detects packet congestion, it will flip the ECN bit to one. When the marked package eventually reaches its destination, the

receiver sees the ECN bit set and can adopt its transmission rate accordingly to prevent further congestion.

All packet loss in a wireless network using non-ECN enabled TCP will always be interpreted as congestion related, even though this might not always be true. The very nature of a wireless link might produce packet loss even when there is no packet congestion. This leads to redundant retransmissions and loss of throughput [2].

The authors of [8] showed analytically that in a single user environment, *if packets are marked based solely on congestion information, there is no significant degradation of TCP performance due to the time varying nature of the wireless channel as compared to wired networks.* [9] also concluded that an improvement in wireless throughput performance can be achieved using quite similar techniques.

With the ECN bit implementation in TCP, it suddenly became possible to differentiate between congestion related and wireless link related packet loss. This was perhaps one of the first steps towards a CLD approach in manet and cellular network designs. By utilizing the ECN-bit, the lower layers of the network stack are able to tell the higher layers about network status (eg. when congestion is happening) and the higher layers can react accordingly. More importantly, it was shown in [8] and [9] that there was indeed a performance gain to be archived using layer triggers in their CLD approach.

4.1.2 Example 2: EventHelix' Protocol Design

The EventHelix (<http://www.eventhelix.com>) protocol design is provided as an example mainly because it incorporate some common and well-known CLD principles used in CLD protocols, and because it is open source and documentation and sourcecode is available on the EventHelix webpage for anyone to use, review and improve.

EventHelix is a streaming protocol, which uses a standardized interface between the different layers, and thus is able to decoupling the individual protocol layers, as shown in Figure 4.1. This allows for **dynamic** insertion and removal of protocol layers from a stack as shown in Figure 4.2, since the interlayercommunication is standardized. This CLD approach is a variant of the layer trigger approach used in network stacks described in example 1. Communication between the different layers consist of a "transmit" and "handle receive" signal where "transmit" is invoked by the upper layer to transfer data to the lower layer. Equally "handle receive" is invoked by the lower layer to transfer data to the upper layer.

Protocol stacks by default tend to be rigid and inflexible in design, because there are a lot of dependencies between the different layers, and changes in one layer often propagate to other layers. Implementing changes in a deployed protocol is impractical at best or even impossible in many cases. As a direct consequence, protocol layers cannot be dynamically added or removed on the fly. This is a huge limitation, because protocol standards today change and evolve quite frequently. There are several reasons why dynamic adding or removal of a layer in a protocol stack is very useful:

- An application detects a failure in the Physical layer currently in use, and thus is forced to use a different medium for its data transport. The

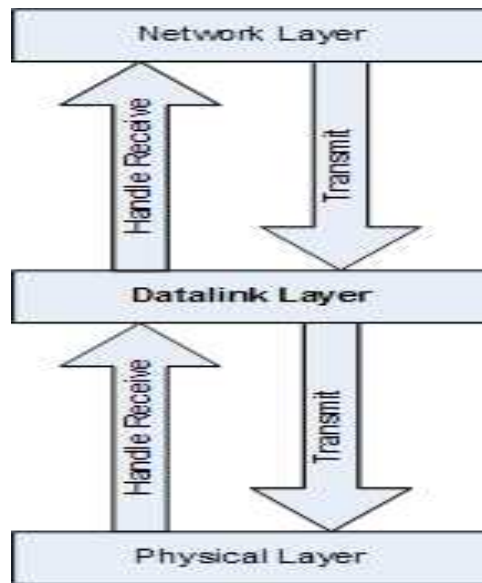


Figure 4.1: EventHelix: Structure of a three layer protocol stack

application is then given the option to change the lower layers of the protocol, while keeping the upper layers intact.

- If a user decides to use encryption, an IPsecurity layer is inserted between the Network layer and Data-Link layer. The application itself does not need to support encryption.
- Inter-layer communication is easy to monitor and/or debug, since a logging layer can be inserted between two layers without interfering the layer communication.
- Loopback and Echoback layers for simple integration and deployment in new environments.

Key features of EventHelix' CLD streaming protocol:

- The implementation of one layer is completely decoupled from the other layers.
- Layers can even be added or removed without the need to change the implementation of the individual layers. Depending on the needs of for example a secure transmission, an IPsecurity layer could be added between the IP and Physical layer without making any changes
- A single layer can interface with any number of upper or lower level layer protocols using the same interface.

EventHelix is under heavy development, but the webpage provides a lot of freely available sourcecode and documentation about their implementation. As we shall see in example 3, the MobileMan reference architecture will also need a newly implemented protocol stack to be able to take advantages of the CLD

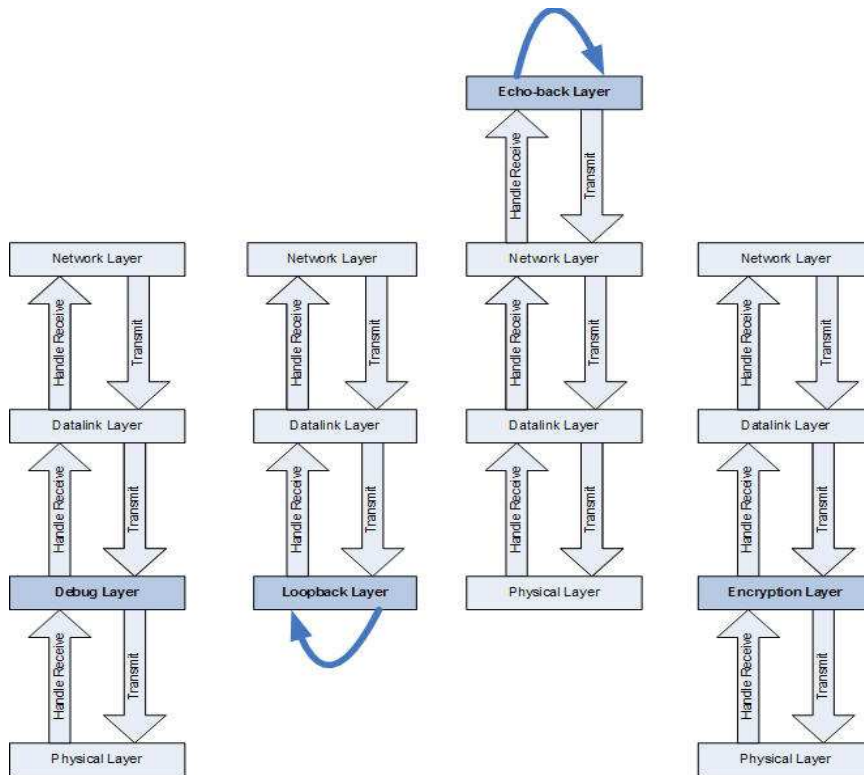


Figure 4.2: Examples of dynamical adding of protocol layers in the EventHelix protocol stack.

principles of their model. Since protocol redesign will be a key component in any CLD system, EventHelix was included as an example on how CLD-enabled protocols might look like in the future.

4.1.3 Example 3: MobileMan

The MobileMan reference architecture was presented in [6] February 2004. The primary design goal right from the start was to implement a system-wide CLD in a manet protocol stack using 802.11, and the authors claim no other existing reference architecture has yet to accomplish this goal. The reference stack design can be seen in Figure 4.3. Protocols belonging to different layers can co-operate by sharing network status, while still maintaining the layer separation in the protocol design. The authors list 3 main advantages of their reference design:

- Full compability with existing standards as it does not modify each layer's core function
- A robust upgrade environment: adding or removing protocols belonging to different layers in the stack is possible without modifying the operations at the other layers
- Maintaining the benefits of a modular architecture

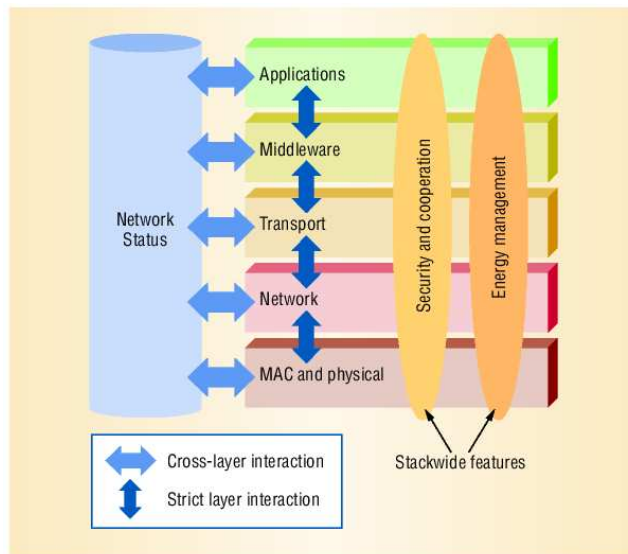


Figure 4.3: MobileMan’s cross-layered network stack

Energy management, security and cooperation are cross-layered by nature, as seen in Figure 4.3. The core component of the reference design, is the *Network status* repository. Whenever a protocol in the stack collects information, it will publish this to the repository and thus making it available for every other protocol. The authors state that *this avoids duplicating efforts to collect internal state information and leads to a more efficient design.*

Since co-operation between the different protocols takes place in the network status repository, this feature does not compromise the expected normal operation of the stack. You can even replace it with a legacy stack implementation without any other implication than losing the optimization and performance gain provided by the CLD.

The authors claim that the MobileMan reference architecture will offer the following performance advantages in an ad hoc network design:

- Cross-layer optimization for all network functions
- Improved local and global adaptation
- Full context awareness at all layers
- Reduced overhead

Since the MobileMan reference architecture is just a proposal so far, no performance analysis exists to back up their claims. At the writing of [6] they were in the process of a complete redesign of the protocols used together with MobileMan, to be able to take advantage of the CLD features in the model. Their approach *aims to optimize overall network performance, by increasing local interaction among protocols, decreasing remote communications, and consequently saving network bandwidth.*

The MobileMan implementation is what can be described as a system-wide CLD where stackwide layer interdependencies are designed and implemented to

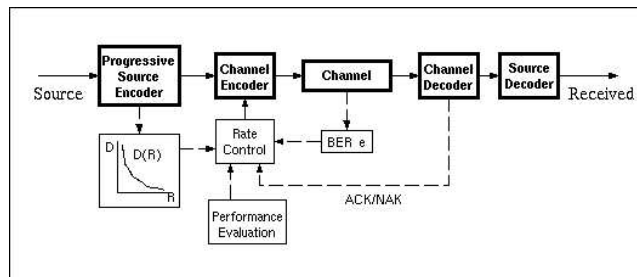


Figure 4.4: MobileMan's cross-layered network stack

optimize overall network performance. The authors claim that knowledge has to be shared between all layers to obtain the highest possible adaptivity. The different protocols will actively seek to use available state information throughout the stack to adapt their behaviour and thus maximize throughput and efficiency and minimize delay and power usage - all this while still being QoS efficient. While the schematics might look elegant and simple on paper, we have yet to see how the actual implementation will look like and how it will perform in real life.

4.1.4 Example 4: Joint Source Channel Coding

Traditionally, source and channel coding have been addressed as independent problems. Source coding aims to remove redundancy using an efficient representation of the source signal. Channel coding involves adding redundancy to achieve error free transmission in noisy environments. Shannon's separation theorem states that source coding and channel coding can be done separately and sequentially without loss of optimality. However for Shannon's separation theorem to be truly optimal, infinite block length codes have to be used, which induces infinite complexity and delay. Such requirement makes Separate Source Channel Coding (SSCC) approach problematic for most real-time applications. Further more, SSCC is designed for the worst case scenario. This means available resources is wasted if and when the channel is good. Meanwhile, when the channel state is worse than what the channel code is designed for, the system performance collapses and the BER can increase exponentially. And last but not the least, SSCC is no longer applicable for multi-user and non-ergodic channel environments.

When reviewing practical communications systems, it is often possible to find a better solution. Instead of treating the source and channel coder as independent blocks, the Joint Source Channel Coding (JSCC) approach shares information between the source coder, the channel coder and utilizes the soft information from the physical layer. As shown in the example in Figure 4.4, the Rate Controller (RC) takes feedback from both the source coder, channel and channel decoder and optimally allocates the overall rate between the source and channel coder under preset performance demands - for example from a QoS demand from the user or network provider.

Common JSCC techniques include:

- Source optimized channel coding
- Channel optimized source coding
- Iterative algorithms
- Channel codes for compression and error protection
- Shannon mappings

We will describe Shannon mappings later, when dealing with the revolutionary approach to CLD. JSCC has proven itself useful for multimedia content in manet and cellular networks [11], [12], [13], [14]. It allows the coder(s) to better exploit the changes in the channel conditions or variations of the source contents. As a result, we have more adaptive and robust systems which give better performance when operating under delay constraints or time varying channels.

4.1.5 Example 5: Designing a Mobile Broadband Wireless Access Network

In [15] a new 4th generation (4G) wireless mobile broadband access network is proposed, discussed and compared against a 3G high-level architecture. There are several reasons why CLD optimization in any 3G network is difficult :

- The different layer protocols follow a strict layered design.
- The different network elements do not have sufficient real-time information about instantaneous channel condition or traffic queue, which is essential for any scheduler making decisions. This results in sub-optimal resource and QoS utilization.
- Due to excessive communication delay between the different network components, sharing information between them can not be done in an efficient manner.
- Isolation between the layers limits the ability of the transceiver and the controller to be IP QoS aware.

The proposed 4G system use the IP but with a new air interface technology based on Orthogonal Frequency Division Multiplexing (OFDM), which allows it to achieve higher spectral efficiency than Code Division Multiple Access (CDMA) based 3G networks. CLD has been one of the main design goals, and the physical (PHY), Medium Access Control (MAC) and Logical Link (LL) layer are jointly optimized, while maintaining the compability with the standard IP network architecture.

Today it is not enough simply sharing the resources among the different users, you also have to do it efficiently and fairly. The scheduler in this system becomes the focal point for achieving any CLD optimization. The general function of any scheduler, is to intelligently allocate available resources between different users having different QoS demands. Thus, all packets should not be treated equally in the scheduler. Through CLD optimization the scheduler is provided with a rich set of cross-layered information such as traffic packet queue

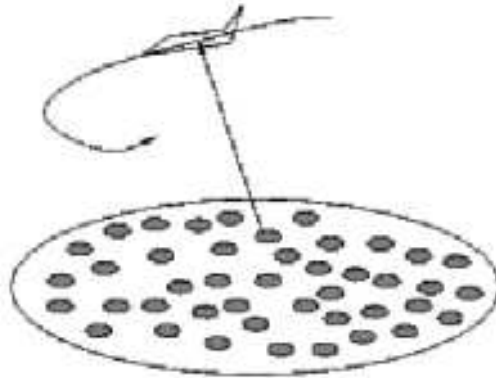


Figure 4.5: Deployment and monitoring of WSN by aircraft

state, QoS demands and channel condition for all users, enabling it to make the best possible decision. The scheduler depends on a MultiUser Diversity (MUD) environment, where the more users, the better the channel and the higher the sum-rates which can be achieved. Using opportunistic transmissions and MUD enables the scheduler to always prioritize the user which at any given time has the best possible channel conditions [16]. For any next generation manet or cellular network - which should facilitate the seamless operation of the various applications and protocols - high bandwidth, low latency, QoS efficient and power savings will become key factors in its success.

4.2 Revolutionary Cross-Layer Design Examples

4.2.1 Example 6: Wireless Sensor Networks

The fundamental goal of any sensor network, is to collect and aggregate meaningful information from raw local data obtained by the individual sensor nodes. The sheer number of sensors nodes in a sensor network, combined with the unique characteristics of their operating environment, makes designing WSNs and their applications a unique challenges to every researcher or engineer. The designers have to consider, among other things:

- Anonymity of individual sensors
- Limited power budget
- Possibly hostile environment

Since each sensor node is powered by small batteries that are hard to replace, they can only transmit a finite number of bits before they run out of energy. This makes reducing energy consumption per bit for end-to-end transmission an important design consideration for any WSN. Every sensor node reports back to a local hub node within a certain deadline, and since all layers of the protocol

stack contributes to energy consumption and delay, an efficient WSN requires a CLD across all these layers as well as the underlying hardware [7].

Article [17] shows how to achieve a significant energy saving in WSNs using variable-length Time Division Multiple Access (TDMA) compared to the traditional uniform TDMA scheme, when jointly designing the MAC layer and the link layer with regards to hardware constraints. Articles [18, 19, 20] also shows that significant energy saving is possible using CLD approaches compared to traditional MAC and routing schemes based on traditional layered structures. Article [21] even shows that the CLD approach can be extended horizontally to allow cooperation between multiple nodes, and thus allow virtual Multiple Input Multiple Output (MIMO) systems to be created. Several numerical examples show that both energy savings and delay reduction can be achieved at the same time within a given transmission in cooperative MIMO systems, even when taking into account the extra delay and energy usage caused by the horizontal information exchange.

Article [22] and [23] use JSCC applied to a joint estimation problem in an energy-constrained sensor network. The articles conclude that for an optimal power scheduling scheme, sensors in a WSN with bad channels or poor observation qualities should decrease their quantization resolution or even perhaps become inactive in order to conserve power. The remaining active sensors however, should have their optimal quantization and power levels determined jointly by using individual channel gain and noise variations using JSCC. This results in a significant energy saving compared to a uniform quantization applied to all sensors.

Because of the extremely limited power supplies in each sensor node, this will require lightweight communication protocols as well. Such protocols have to support security, data collection, training, calibration, self-organization, maintenance, protection from defective sensors, routing and many other tasks used today in WSNs. It might even be useful if sensor nodes can resupply their energy from the specific environment they are placed in, by employing light, temperature or kinetics for example.

Perhaps it is more clear now why a WSN is a revolutionary approach to CLD. Since WSNs are highly task specific, and most often a stand-alone implementation which does not need to communicate with other IP based networks, it does not need to follow the strict layered design most other examples do in this tutorial. This also enables researchers and engineers to open up new doors, and approach problems in a whole new way than what they can do in other systems based on a strict layered structure. WSNs involve several technological issues bringing different research communities together, but also social and economical aspects of WSNs must be discussed and addressed before introducing such potentially all-changing technologies to the society. These issues have made WSNs a very hot topic the recent years, and a lot of resources have been devoted to bringing this technology to life.

4.2.2 Example 7: JSCC Shannon Mappings

A fundamentally different and revolutionary approach to the JSCC described in example 4, which involves Shannon mappings [24, 25, 26], tries to tackle the source-channel coding problem in the revolutionary sense. The traditional source and channel coding blocks that work in the digital domain are replaced

by the direct analogue source symbol to channel symbol conversion through mappings. The result is a highly unique and robust JSCC system, and fits the description of a revolutionary CLD well. Work is being done on building its backward compatibility into the network's digital domain. This will require optimal quantization of the Shannon mappings depending on the channel state conditions. Further work on JSCC with Shannon mappings can be found in [27, 28, 29],

Chapter 5

Cross Layer Design Skepticism

While CLD in wireless manet and cellular networks has been given **a lot** of attention in the last few year, there are also critics who argue to exercise caution while engaging in CLD approaches [30]:

1. A modular architectural design has proven itself time and time over. Modularity provides the very essence of abstraction necessary for researchers and engineers to fully understand the overall system. The layered OSI reference model is used as an example, in which the internet and its tremendous success is loosely based on. A modular design can also accelerate the development, since different designers can focus their efforts on different subsystems, with the assurance that the entire system will interoperate once it is brought together. This is not the case for a CLD.
2. CLD by nature creates interactions between processes and layers. Some interactions are intentional, others might be unintentional. CLD can thus create loops, and it is well known from Control Theory that stability becomes an issue under such conditions. The "Law of Unintended Consequences" can potentially take over if one is not careful.
3. Standardization allows subsystems to be used across many applications, thus resulting in lower development costs and time-to-market which in turn increase usage. In contrast, a CLD system will need to be adopted to every application and this will increase cost and time-to-market greatly.
4. Once the layering is broken, the possibility to review and redesign parts of the system is lost since everything is interconnected one way or the other. Protocols have to be reimplemented in a cross-layered fashion, taking into account several layers as opposed to earlier where a protocol could be developed in isolation.
5. A system-wide CLD can lead to "spaghetti" implementation, which in turn will hamper further innovation and be difficult to maintain. Future design improvements may become impossible, because it will be difficult to foresee exactly how a new modification will affect the overall system operation.

The main point raised by [30], is that while CLD gives a short term performance gain, good architectures are usually based on longer term consideration. While this claim is impossible to contradict or prove wrong right now, since the CLD concept is relatively new, it might be wise to use CLD with caution in the evolutionary approach and perhaps avoid the more system-wide CLD variants if compability is a main concern for the implementation. In revolutionary approaches however, you already have a task-specific implementation and compability is (normally) not an issue. As such, most points raised above does not apply to the revolutionary part.

While the authors of [30] certainly raise valid concerns about CLD, there is no denying the gains CLD can provide as shown in a number of examples in this tutorial. It has been used with great success in JSCC and WSNs and in various trigger implementations. The MobileMan project certainly looks interesting on paper, however it remains to be seen how this system will perform and how "spaghetti"-like the implementation actually will become. At least MobileMan certainly has the potential of "proving" or "disproving" point 5 listed above, and because of this it is very important that such ideas and approaches get tested at least.

What it all comes down to eventually is, are the system designers willing to abandon a modular design - in part or as whole - in favour of a CLD, in order to benefit from its advantages?

Chapter 6

Conclusion and Future Work

In this tutorial we presented the Cross-Layer Design principle from the evolutionary and revolutionary point of view. In the evolutionary approach we looked at CLD as a design principle involving state information sharing between two or more layers. This approach is bound by the strict layering found in traditional implementations, and needs to rely on extending the existing layered structure to maintain compability. We gave examples ranging from CLD enabled protocols such as EventHelix, to system-wide CLD implementations such as the MobileMan, to information theory focused Joint Source Channel Coding and last, but not the least, a 4G CLD enabled Mobile Broadband Wireless Access Network.

Although different literature show many advantages with CLD, previous work has mostly focused on joint design of two or three layers only, such as the PHY, MAC and routing layers. It is however necessary to think CLD that involves cooperation and state information sharing between all layers. This brings us to the revolutionary approach. Here the CLD principles are applied to systems where strict layering or backwards compability is no longer neccessary. This also enable us to keep an open mind about future wireless network design structures. Examples in this approach include Joint Source Channel Coding using Shannon Mappings and Wireless Sensor Networks.

However, as much as CLD being the new buzz-word in communications research community, preliminary precautions should always be exercised. The system-wide CLD should be closer examined for necessacity. The longivity of the CLD is yet to be proven in time.

Future work on CLD implementation can be aiming at replacing the traditional layered structures completely. However this might not even be possible because of the demand for compability with every other network using the IP. We therefore propose the following scenario for future CLD implementations: *For very task specific purposes a revolutionary CLD is often beneficial. By removing every redundant part of the layered structure and the protocols used, you get a highly optimized and power efficient system designed exclusively to be applied to a specific problem..* This proposal applies very well to the example of WSNs, and our prediction is that such task specific implementations

based heavily on CLD will be increasingly popular and important in the future. Greater development cost and a delayed time-to-market will be outweighed by the many advantages a system-wide CLD provides.

Chapter 7

Appendices

7.1 Appendix A: OSI quick reference model overview

- **Application layer:** Provide services that directly support user applications.
- **Presentation layer:** Translate dataformats and add encryption to the session.
- **Session layer:** Set up, administer and tear down sessions (connections).
- **Transport layer:** Add identifiers to processes and deal with error-handling information.
- **Network layer:** Handle internetwork sequencing, adressing and routing.
- **Data Link layer:** Add error-checking information and organize bits into frames.
- **Physical layer:** Transmits and receives bits over the physical media.

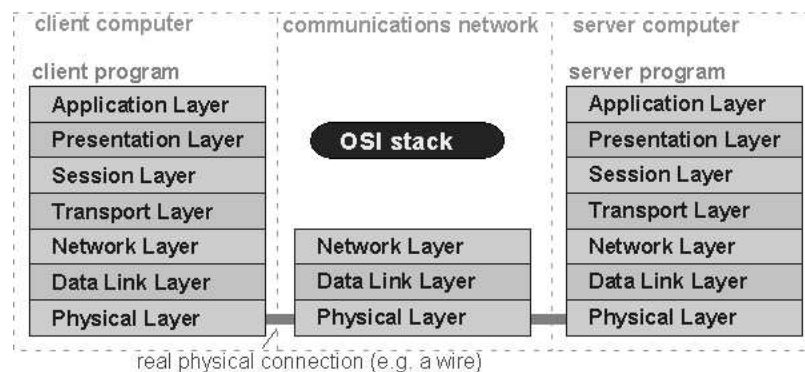


Figure 7.1: Appendix A: OSI reference model applied to a communications network

7.2 Appendix B: TOS-bits

The Type Of Service (TOS) bits are a set of four-bit flags in the IP header. When any one of these bit flags is set, routers may handle the datagram differently than packets with no TOS bits set. Each of the four bits has a different purpose and only one of the TOS bits may be set at any time, so combinations are not allowed. The bit flags are called Type of Service bits because they enable the application transmitting the data to tell the network the type of network service it requires.

The classes of network service available are:

- **Minimum delay:** Used when the time it takes for a datagram to travel from the source host to destination host (latency) is most important. A network provider might, for example, use both optical fiber and satellite network connections. Data carried across satellite connections has farther to travel and their latency is generally therefore higher than for terrestrial-based network connections between the same endpoints. A network provider might choose to ensure that datagrams with this type of service set are not carried by satellite.
- **Maximum throughput:** Used when the volume of data transmitted in any period of time is important. There are many types of network applications for which latency is not particularly important but the network throughput is; for example, bulk-file transfers. A network provider might choose to route datagrams with this type of service set via high-latency, high-bandwidth routes, such as satellite connections.
- **Maximum reliability:** Used when it is important that you have some certainty that the data will arrive at the destination without retransmission being required. The IP protocol may be carried over any number of underlying transmission mediums. While SLIP and PPP are adequate datalink protocols, they are not as reliable as carrying IP over some other network, such as an X.25 network. A network provider might make an alternate network available, offering high reliability, to carry IP that would be used if this type of service is selected.
- **Minimum cost:** Used when it is important to minimize the cost of data transmission. Leasing bandwidth on a satellite for a transpacific crossing is generally less costly than leasing space on a fiber-optical cable over the same distance, so network providers may choose to provide both and charge differently depending on which you use. In this scenario, your minimum cost type of service bit may cause your datagrams to be routed via the lower-cost satellite route.

Bibliography

- [1] T. S. R. et al, “Wireless communications: Past events and a future perspective,” *IEEE Commun. Mag.*, vol. 40, pp. 148–161, May 2002.
- [2] S. Shakkottai, T. S. Rappaport, and P. C. Karlsson, “Cross-layer design for wireless networks,” *IEEE Communications Mag.*, vol. 41, pp. 74–80, Oct. 2003.
- [3] W. Stallings, *High-speed networks: TCP/IP and ATM design principles*. Upper Saddle River, New Jersey 07458: Prentice-Hall Inc, 1998.
- [4] J. Mahdavi and S. Floyd, “Tcp-friendly,” June 1999. This is a webpage overview of different research material related to adaptiv congestion control.
- [5] S. Haykin, *Communication Systems*. New York: John Wiley & Sons, Inc., 3rd ed., 1994.
- [6] M. Conti, G. Maselli, and G. Turi, “Cross-layering in a mobile ad hoc network design,” *IEEE Comp. Soc.*, pp. 48–51, Feb. 2004. Commercial implementation.
- [7] A. J. Goldsmith and S. B. Wicker, “Design challenges for energy-constrained ad hoc wireless networks,” *IEEE Wireless Communications*, pp. 8–27, Aug. 2002.
- [8] S. Kunniyur and R. Srikant, “End-to-end congestion control: Utility functions, random losses and ecn marks,” *Proc. IEEE INFOCOM*, vol. 3, pp. 1323–1332, Mar. 2000.
- [9] H. B. et al., “A comparison of mechanisms for improving tcp performance over wireless links,” *IEEE/ACM Trans. Net.*, Dec. 1997.
- [10] S. Floyd, “Tcp and explicit congestion notification,” *ACM Comp. Commun. Rev.*, vol. 24, pp. 10–23, Oct. 1994.
- [11] A. J. Goldsmith and M. Effros, “Joint design of fixed-rate source codes and multiresolution channel codes,” *IEEE Trans. Commun.*, vol. 46, pp. 1301–1312, Oct. 1998.
- [12] J. W. Modestino and D. G. Daut, “Combined source-channel coding of images,” *IEEE Trans. Commun.*, pp. 1644–1659, Nov. 1979.
- [13] H. Jafarkhani, P. Ligdas, and N. Farvardin, “Adaptive rate allocation in a joint source-channel coding framework for wireless channels,” *IEEE VTC’96*, pp. 492–496, Apr. 1996. In proceeding.

- [14] N. Farvardin and V. Vaishampayan, "Optimal quantizer design for noisy channels: An approach to combined source-channel coding," *IEEE Trans. Inform. Theory*, vol. 33, pp. 827–838, Nov. 1987.
- [15] R. Laroia, S. Uppala, and J. Li, "Designing a mobile broadband wireless access network," *IEEE Signal Processing Mag.*, pp. 20–28, Dec. 2004.
- [16] R. Knopp and P. A. Humblet, "Information capacity and power control in single cell multiuser communications," in *IEEE Int. Conf. on Communications (ICC'95)*, (Seattle, WA), pp. 331–335, June 1995.
- [17] S. Cui and A. J. Goldsmith, "Cross-layer optimization in sensor networks with energy constraints," 2004. To appear at Globecom 2004. Also available at <http://wsl.stanford.edu/Publications.html>.
- [18] S. Cui, A. J. Goldsmith, and S. Lall, "Joint routing, mac and link layer optimization under energy constraints," 2004. Submitted to ICC'05, South Korea May 2005. Also available at <http://wsl.stanford.edu/Publications.html>.
- [19] S. Cui and A. J. Goldsmith, "Cross-layer optimization in tdma-based sensor networks with energy constraints," *IEEE Wireless Communications*, 2004. Also available at <http://wsl.stanford.edu/Publications.html>.
- [20] S. Cui and A. J. Goldsmith, "Energy-delay tradeoffs in tdma-based sensor networks," 2004. Submitted to ICC'05, South Korea May 2005. Also available at <http://wsl.stanford.edu/Publications.html>.
- [21] S. Cui, A. J. Goldsmith, and A. Bahai, "Energy efficiency of mimo and cooperative mimo in sensor networks," 2004. In preparations.
- [22] J. Xiao, S. Cui, A. J. Goldsmith, and Z. Q. Luo, "Joint estimation in sensor networks under energy constraints," *IEEE first conference on sensor and Ad Hoc communications and networks*, Oct. 2004. Santa Clara, CA, October 2004.
- [23] S. Cui and A. J. Goldsmith, "Joint estimation in sensor networks under energy constraints," *IEEE Trans. Signal Processing*, 2004. Submitted to IEEE Transactions on Signal Processing, 2004.
- [24] T. Ramstad, *Insights Into Mobile Multimedia Communications*. Academic Press, 1st ed., 1999. Chapter 26: Combined Source Coding and Modulation for Mobile Multimedia Communication, pages 415-430.
- [25] T. Ramstad, "Signal processing for multimedia: Robust image and video communication for mobile multimedia," *Nato Science Series: Computers and Systems Sciences*, vol. 174, pp. 71–90, 2000. ISO Press.
- [26] T. Ramstad, "Shannon mappings for robust communication," *Teletronikk*, vol. 174, pp. 114–128, 2002. Information Theory and its Applications.
- [27] J. M. Lervik, A. Fuldseth, and T. A. Ramstad, "Combined image subband coding and multilevel modulation for communication over power- and bandwidth limited channels," *Piksel'n*, vol. 11, pp. 7–12, Dec. 1994. (Reprint from Proc. Workshop on Visual Signal Processing and Communications, New Brunswick, NJ, USA).

- [28] T. A. Ramstad, J. M. Lervik, and A. Fuldseth, “Shannon mappings in robust visual communication.” To be submitted IEEE Signal Processing Magazine, 1996.
- [29] H. Coward and T. Ramstad, “Robust image communication using bandwidth reducing and expanding mappings,” *Conference Record of the Thirty-Fourth Asilomar Conference*, vol. 2, pp. 1384–1388, Oct. 2000.
- [30] V. Kawadia and P. R. Kumar, “A cautionary perspective on cross layer design,” 2004. Draft.