S38.3115 Signaling Protocols – Lecture Notes

Lecture 1: Introduction to Signaling

Contents

Prerequisites for the course

We assume knowledge of software specification, design and implementation and some hands-on experience on IP or circuit switching equipment. Hopefully, at least part of the software experience should be from a real time environment. The hands-on experience can be gained on S38 laboratory courses. In the past those with little or no software experience have had great difficulties in comprehending what we will be talking about. Probably, this is because we try to describe requirements and constraints and how these are transformed into design. This is far from trivial. I hope these lecture notes will make things clearer and at least a bit easier to learn.

Learning Outcomes

We summarize the goal of the course at the end of this Chapter of the Lecture Notes.

Purpose of signaling

The purpose of signaling is to set-up, supervise and tear down calls, connections or communication sessions. Traditional networks such as the PSTN (Public Service Telephone Network (yleinen puhelinverkko)) support the setting up of voice calls where an audio channel is established between two or more users or subscribers. Modern networks support the setting up of multimedia calls that in addition to audio or voice, transfer other types of media, like video or data.

Let us first define a few key terms: *circuit switching, packet switching, connection-oriented and connectionless.* These terms are fundamental to understanding the functions of signaling. First, *circuit switched networks* are always connection oriented. In such networks, the units of data that are switched in network nodes do not carry address information and a switching fabric state is established in each node between an incoming circuit and an outgoing circuit such that data units will continuously flow from the incoming to the outgoing circuit. In these networks only having first established a connection from end to end, users of the network can transfer data. Usually, we establish a bidirectional connection. In PSTN, these connections have a constant capacity of 64 kbit/s and the units of data that are switched are bytes. Note, that if there is no address information in the carried data units, it follows that connections have a constant rate. This applies equally to PSTN and light paths/wavelengths.

In circuit switching, the addressing of the switched data units is “implicit” i.e. the placement of the physical connection + the placement of the data within the multiplex that is carried on the physical connection = address.

Packet switched networks may be either connection-oriented or connection less. Packets always carry a header including some address information plus a payload. In a connection oriented packet network, the packets will carry address info that has only *local* significance. As a result, an end-to-end *virtual connection* needs to be established before two users can communicate. This is the case for example in ATM or MPLS.

Similarly, in IP networks private IP addresses have only local significance. Therefore, a NAT (network address translator) device is placed on the boundary of the private address realm and the global addressing realm of the Internet and that NAT device will establish a *binding state for a flow*. The binding state has a timeout, i.e. when no data has been transferred for the duration of the timeout, the binding will be removed.

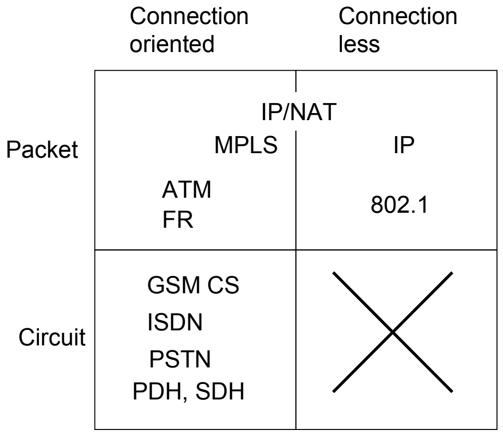

Figure 1.1 Types of networks

In a *connectionless packet switched network*, packets carry an address or addresses that have global significance, i.e. are *globally unique* (actually this term “globally unique” is confusing because any ID or address has a finite length and thus can label only things in a certain domain). An example is the traditional Internet based on the IP protocol. For end-to-end connectivity, a sender just needs to know a globally unique address of the receiver and data can start flowing from one user to another. In practice, the current Internet has a lot of limitations for sending packets between any two Internet users and besides globally unique addresses makes use of private IP addresses as well. The nodes of a classical IP network, called routers, do not keep flow state. There is no need, because given a globally unique destination address and a pro-actively created routing table; the node always finds the outgoing port for each individual packet.

Figure 1.1 shows a typology of networks in terms of switching and how users reach each other. Connection oriented networks need connection (session or flow) state in the nodes for mapping incoming traffic to outgoing traffic. This state can be created and maintained in three different ways.

(1) The state can be configured and monitored by a separate element and *network management system* while the nodes stay relatively dumb. This is the case for example with plesiochronous digital hierarchy (PDH) and synchronous digital hierarchy (SDH) cross-connect and add-drop nodes. This may also be the case in the so-called Carrier Grade Ethernet (CGE) transport networks.

(2) The state can be created dynamically by *signaling* like in PSTN, ISDN and circuit switched GSM. This is possible also in ATM although, most times ATM is used in the first mode.

(3) There can be an adaptive algorithm that conspires to create and maintain the state without users knowing about it. This approach is used in IP Network Address Translators (NAT). A NAT function resides on the boundary of a private and the public IP network. One can also state that the 3rd option is a special case of signaling. We call this kind of signaling, that is embedded in the normal message pattern, *implicit signaling*.

The first approach is feasible if the connections do not change too often and the number of communicating parties is not too high.
Signaling is a reasonable approach if the duration of connections is from several seconds to minutes or hours. The shorter the connections are the higher is the ratio of signaling bits to user's data bits. At some point this overhead grows too high. This is the case for example in Web browsing: if an object download would require first setting up a connection (which it does not), this would lead to very high overhead and slow operation. In such a case, the most reasonable approach is to use a connectionless network. Also, the adaptive approach e.g. using Network Address Translators is efficient enough to support short flows such as are typical of web browsing. Signaling scales to any number of destinations provided that a suitable addressing format is available.

The term "call" refers to a phenomenon that involves subscriber A - calling party or a caller (different terms are used for historical reasons), the caller's device, a chain of transmission systems and network nodes, subscriber B (called party or callee) and the callee's device. A term that is nowadays also used instead of "call" is *session*. For setting up the call, signaling is needed and within a call some communication between the parties takes place.

Let us now define what is signaling. *Signaling is the transfer of control information for the purpose of setting up, supervising and tearing down calls, connections or sessions between users in a network.*

From the definition one can see that we are dealing with a control system where the object of control are the resources in the network and the control intelligence called "call control" resides in several network nodes. The language that is used to carry control information between different call control elements is called signaling.

Network technologies used to carry calls differ. We may have a circuit switched, connection oriented network (e.g. PSTN or ISDN, GSM, 3G) or a packet switched connectionless network (Internet or an IP network). Physically, the resources that are used to build the networks also differ greatly. Signaling is different in different networks.

What is common to all known networks in terms of signaling is that we need *to locate the called party, figure out if he or she wishes to engage in a call and* *to agree on a minimum set of technical parameters for the session.* These functions are needed independent of technology. To create the agreement between a caller and the callee, the parties need to agree on things like: audio coding method and speed, video coding method and speed etc.

A characteristic of a call is that resources such as circuits (e.g. time-slots in the PCM transmission systems) are reserved in the network for the duration of the call. E.g. in PSTN, ISDN, GSM and circuit switched 3G networks a voice circuit is established end-to-end. The end-to-end connection has several legs that are clued together (switched) at exchanges on the call path. It may be that for a period during a call such as part of the establishment phase some special resources are needed. Examples are tone generators that can send for example dial tone or busy tone to the caller. Older signaling systems used to require signal receivers and signal senders. From a network point of view it is important that exchanges keep accurate track of the use of all these resources, so they are reserved for the call and released when they are no longer needed. From the need to connect e.g. tone generators to the voice circuit it follows that the establishment of a circuit connection takes many switching actions and that call control in circuit networks takes a lot of micro-management of network resources during a call.

To further characterize signaling, we note that from a network node point of view, relating to a single call it must exchange control information on the incoming side (towards the caller) and on the outgoing side (towards the callee), each neighbor works in its own pace while the node needs to make decisions based on its own data, or data it may need from other control nodes and keep track of the resources reserved for the call.

Call state

A call has **state** that is created at the beginning of the call and changes on events during the call. The state information on a call is needed in the caller's device, in each exchange or signaling node on the signaling path and in the callee's device. Moreover, it is convenient in a network signaling node to separate the following concerns: *incoming signaling*, *incoming call control*, *outgoing call control* and *outgoing signaling*.

By defining separate *state machines* for each of the concerns mentioned above we avoid *state explosion*. The term state explosion refers to the idea that a set of states is needed to reflect each concern related to the call. If all aspects were integrated into a single state machine, the overall number of states would be the product of the number of states of each of the aspects.

Each of the concerns has its own set of states. Incoming signaling state keeps track of the sequence of signals on the incoming side, incoming call control is responsible for making a routeing decision: i.e. based on called party number it will decide in which direction the call should proceed from the current node, outgoing call control is responsible for outgoing side resources and outgoing signaling state reflects the state of communication with the next hop neighbor on the call path.

In an IP network the resources that may need to be controlled by signaling differ from Circuit networks such as PSTN. Related to signaling we note that, if the Internet were as it was initially designed, an end-to-end transparent network for any user to communicate with any other user, the job of signaling would be simple. For such an ideal network, all that would be needed is to locate the callee, establish a wish to communicate and agree on codecs and other technical parameters for communication between the user devices. The weakness of this ideal Internet is that assumes that all users are nice people. We know that this is not the case and that denial of service attacks by flooding a link or node with useless traffic are an inherent malady of the Internet.

However, the current Internet is not an ideal end-to-end transparent network. Actually, one might say that the Internet does not have B-subscribers because users are behind firewalls, use private IP-addresses etc. So, a call over an IP network may require opening temporary holes in firewalls and finding means to traverse Network Address Translators that sit on address space boundaries. Traversing Network Address Translators is simple for Client-Server applications. However, the B-subscriber's device needs to act as a server. If it happens to reside in a private address space behind a NAT, it may be unreachable both from the global address space and from another private address space thus making incoming calls impossible (without the B helping out actively).

Signaling and mobility

It is convenient for users to be reachable where ever they are. For this purpose the network needs to keep track of user's locations. For example, in GSM the mobile makes location updates regularly automatically. The GSM network has a Home Location Register, a database that stores the location information with the accuracy of a Visitor Location Register that is inside a Mobile Switching Center taking care of the users roaming in a certain geographical area.

Also the transfer of location and other mobility related control information falls under the definition of signaling although the transfer may take place while there is no call. After all, the purpose of the location updates is to facilitate calls.

Digital transmission and the structure of PCM system

In order to understand why call state is needed and to understand the architectural constraints that needed to be taken into account when digital signaling systems were designed, we must understand the transmission plant over which calls and signaling are carried. The nature of the transmission plant has had a significant impact on the architecture and design principles of e.g. the CSS7 signaling system.

Pulse code modulation or PCM was the first digital transmission system. It was invented in the 1930's but implementation became feasible only late 1960's. For example Nokia used to be in the business of manufacturing telephone cables and during 1960's it decided to add value to their cables by designing a digital transmission system based on PCM. This is the origin of Nokia's path to electronics and telecom equipment manufacturing. Nokia's first PCM system went into live use in a network in 1968.

During 1980's a move to the SDH (synchronous digital hierarchy) started. Inside an SDH frame PCM signals are still carried. Also, a SDH signal is usually broken into PCM signals before it is interfaced to an exchange.

The basic assumption of the PCM system is that voice is sampled 8000 times per second in order to transfer the narrow band of less than 3.4 kHz. Recall the *Nyqvist theorem* of digital transmission: in order to transfer and recover a signal with a maximum frequency of N, the analogue signal must be sampled at double that rate. In the PCM system each voice sample is carried with 8 bits. This gives the basic speed of one channel that is used in the PCM system: 8000 × 8 = 64 kbit/s. A channel in PCM can be viewed as a continuous digital signal at the speed of 64 kbit/s.

PCM is a time division multiplexing system (TDM). This means that on a single cable, the system multiplexes many channels that can each carry for example one voice signal of 64 kbit/s. In ETSI networks, the first order multiplexed PCM signal has 32 channels of which one is used for frame synchronization and the rest in principle can be used for voice or data. In practice, from the era of analogue signaling usually one of the 31 channels is allocated for signaling, so 30 channels remain for voice or user's data. The basic multiplexed PCM signal of 32 channels is called E1. The transmission speed is 32 × 64kbit/s = 2048 kbit/s. Often we just talk about a 2Mbit/s signal. In a native PDH and SDH transmission system the flow of bits is continuous from the moment when the devices are powered up. When a voice signal is carried on one channel in such systems, the bits on that channel are useful from the end user's point of view. On channels that are not used for calls or data, the carried bits are in fact transmission system overhead. A channel must always carry something, if not voice or data then the alternative is for example an idle signal from a tone generator.

For understanding how PCM works we must look at the signal from the receiver's perspective. Sending bits is easy. The hard part is receiving and making sense of the received signal.

The E1 signal is a series of bits. Bits are represented in a line code called HDB3, we will talk about it later on the ISDN lecture. For the time being, let us assume that there is a way of receiving bits such that the method can be efficiently implemented on silicon. How can the receiver decide where are the channel boundaries in the series of bits? The answer is framing.

Frame synchronization in PCM system

The PCM signal can be broken down into timeslots. A timeslot is a placeholder for a voice sample or user's data of 8 bits or for some other 8 bits. In the PCM system a sequence of one synchronization timeslot of 8 bits plus the timeslots for the following 31 channels is called a *frame*. The length of the frame is 32 × 8bits = 256 bits. A PCM frame with a part of a subsequent frame is presented in Figure 1.2.

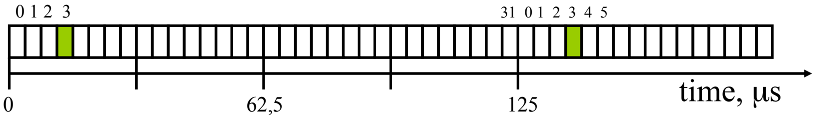

Figure 1.2: A PCM signal.

In Figure 1.2, timeslots are numbered from 0 to 31 and the figure shows how after the first frame the next follows immediately after. We can say that frames follow each other back-to-back. The Figure also shows timing of the signal. The length of the PCM frame in time is 1/8000 s = 125μs. The time duration of a single timeslot is 125/32 = 3,9 μs. The length of a bit in time is 1/2,048 = 3,9/8 = 0,488 μs.

So, how does the receiver figure out where the frame and channel boundaries are? It is agreed that the contents of timeslot zero are used for frame synchronization and that the contents in each other timeslot zero are c0011011 (we call this a frame mark). The frame in which this appears is called an even frame (frame number 0, 2, 4…). Having received this bit combination the receiver counts to 63 and looks if the next 8 bits are again the same c0011011. If yes, the receiver keeps counting and receiving and if it sees the same bit combination several times at this distance from the previous occurrence, it decides that it has found the frame boundaries and consequently just by counting it is also able to spot the timeslot boundaries.

What if one of the users would also send the c0011011 for a long time? While operators were carrying only digitally encoded analogue signals of voice or data encoded with analogue modems this would not have been possible. However, when the Integrated Services Digital Network was introduced, it meant that users are offered a transparent 64 kbit/s channel. Channel transparency means that a user can send any bit combination for any length of time. Due to ISDN, it would have been possible for a user, for the sake of prank if nothing else to keep sending the frame mark and if a receiver looses its frame sync and needs to recover, it could accidentally have synchronized to the user's mark instead of synchronizing to the timeslot zero. This was unacceptable and after close to 20 years of using the PCM system it had to be upgraded for ISDN networks. *The upgrade was that the first bit of the frame mark was redefined to carry a checksum of 8 bits calculated over the previous 8 frames.* The checksum bit is shown by "c" in our notation for the frame mark. The checksum calculated over 8 frames is sent in the next 8 frames in the c –bit.

The result is that besides the base frame, a PCM signal in ISDN also has a multi-frame of 16 frames.

The length of the multi-frame is 16 × 256 bits = 4096 bits.
In time the length is 16 × 125 μs = 2ms.

The result is that resynchronization of a PCM receiver to an ISDN compatible PCM signal may take several milliseconds.

In PSTN with analogue signaling a PCM has another multi-frame structure that is used for Channel Associated signaling (CAS). This used to be based on time-slot 16 that was sub-multiplexed to carry signaling bits for voice timeslots on the same PCM line. This CAS related multi-framing is completely independent of the multi-framing for ISDN. Even if both of these multi-frame structures are used on the same PCM connection, they are independent. If multi-framing or frame sync is lost, the alarm system in a network node may produce alarms related to both at the same time and these need to be correlated but otherwise, really the two multi-framings are independent. Since the timeslot 16 multi-framing was needed for analogue signaling and analogue signaling is rarely used nowadays, the timeslot 16 multi-framing is largely a historical phenomenon.

To conclude the discussion of PCM: it carries raw uni- or bi-directional bit-streams called channels in timeslots. A channel may be allocated to a user or a telephone conversation and it carries no overhead for management or other purposes. Also the fact that a channel has constant speed of transmission and the fact that there is no explicit address information in the channel are interlinked: variable speed channel always needs at least locally significant address information in the channel to identify the channel boundaries.

Impact of PCM system on signaling

First of all signaling systems need to able to allocate calls to channels or timeslots in a PCM connection. The call control needs to keep track of which timeslots are used by which call. This information is part of the call state. Signaling needs to be interfaced with the alarm system so that it will not place calls on non-functioning PCM –lines and it must also be possible to tear-down calls from a PCM, so that the PCM connection can be taken out of use.

Moreover, since on a call path, in each node, we need to establish call state, it follows from the nature of the transmission environment, that a call must progress through the network hop-by-hop, i.e. from node to adjacent node, until it reaches the terminating node.

We will later discuss how the capacity of PCM channels impacts the design of signaling systems.

Allocation of functions in different networks

In this Section we discuss the allocation of different functions related to signaling and telephony as a whole in different places in networks and user devices.

Voice coding

Packet networks such as the Internet differ from Circuit networks such as PSTN in where the responsibility for coding and decoding of voice resides. In Packet networks the connected devices are always computers of some sort while the network just moves packets. Therefore, it is natural that voice coding and decoding responsibility lies with the user devices.

PSTN phones can be simple electromechanical devices. In PSTN, such analogue phones send voice onto the local loop as an analogue electrical signal and it is typically encoded into a digital form on the first circuit board of the local exchange to which the local loop is connected. This board is called *a line card* or *a subscriber line card*. In the opposite direction decoding of voice takes place on the same card.

The fact that encoding and decoding of voice take place in the network lead to just two voice coding standards for digital PSTN network: G.711 A-law for ETSI and μ-law for ANSI networks. If each operator would use its own voice coding method, on network boundaries a lot of equipment would be needed to decode and encode between the different standards – this would obviously make no sense from an economic point of view because the cost of such equipment would have to be borne by the operators. Also, each translation would most likely degrade the quality. It has been agreed that the translation between A-law and μ-law is the responsibility of the ANSI network.

GSM and ISDN networks were designed with the requirement of smooth interworking with PSTN that at the beginning of GSM and ISDN had more than 500 million subscribers connected while the new networks had just a few.

The design decision in ISDN was that the G.711 codec was moved from the line card in the exchange to the ISDN phone.

In GSM initially a single codec was adopted and an element called the *Transcoder* was placed between the Base Station Controller and the Mobile Switching Center to map between G.711 and GSM voice coding. By adopting a new codec, air interface capacity was saved, because the GSM codec uses some 13 kbit/s while the PSTN, G.711 uses 64kbit/s for a single voice stream. So, this new coding method (GSM codec) was restricted just to GSM access while the network itself still continued to use the same old G.711 inherited from digital PSTN. Later on, additional coding methods were introduced to GSM networks, such as Enhanced Full Rate (EFR) and half rate codecs. All these are supported by the Transcoder element and mapped to G.711 for smooth interworking with the PSTN.

When calls from mobile to another mobile became commonplace, the so-called Transcoder Free Operation was introduced. This means that under the control of signaling, the GSM switching centers can tell the Transcoders at the Mobile Originated (MO) and Mobile Terminated (MT) sides to ignore the mapping to G.711. The result is that the end-to-end circuit connection uses only 2 bits out of each 8 in each timeslot of 64 kbit/s connection. Dropping the transcoding to G.711 is useful even if one does not save core network capacity because transcoding always degrades voice quality.

Sometimes in networks, elements called *Media Gateways* are needed. These have a similar function as the Transcoder in GSM. A Media Gateway supports several coding standards and is capable of mapping between them in real time. An example where a media gateway is needed is on the boundary of an IP network that supports Voice over IP and a Circuit Switched Network (PSTN, ISDN, GSM). In a Transcoder on a physical level all interfaces are based on the PCM standard for digital transmission (2Mbit/s lines) while in a Media Gateway the physical interfaces may include PCM, SDH, ATM, Ethernet in different variants etc.

Switching in circuit networks

Exchanges in circuit networks are responsible for switching the voice circuits through. For this purpose, all voice circuits are interfaced in the exchange into the Switching Fabric through *exchange terminals*. The exchange terminals take care of PCM frame synchronization and monitoring of the quality of the received PCM signals. An example is a switching fabric that has 8192 PCM interfaces of 2Mbit/s each. Total switching capacity of the Fabric would be 16Gbit/s. Each PCM has 32 timeslots each 64 kbit/s in both directions of which 30 can be used to carry voice signals. Each direction has to be switched separately in the switching fabric. Each timeslot carries a bit stream on

64kbit/s that, for example, supports the transfer of 8000 voice samples of 8 bits each per second.

Also things like tone generators and devices that need to process the contents of timeslots are interfaced to the same Switching Fabric. These also include all control computers of the switching system: since these need to process signaling that is carried over PCM lines, it is convenient to attach control computers to the Fabric with internal PCM lines. Establishing a signaling link from a control computer in an exchange to a control computer in the neighboring exchange means that a PCM timeslot between the exchanges needs to be reserved in both directions and through-connected in both exchanges to the internal PCM line that is connected to the appropriate control computer at each exchange.

It is a property of a so called non-blocking PCM switching fabric that it can map any incoming timeslot to any outgoing timeslot provided the outgoing timeslot is not in use already. Another property of the fabric is that no outgoing timeslot "hangs in the air" i.e. carries nothing. An outgoing timeslot always has some signal in it and in the absence of anything else to carry, an idle tone is typically sent from the tone generator. It is also typical of modern switching fabrics that they are able to *multicast* any incoming timeslot into any number of outgoing timeslots.

Switching a timeslot means that the exchange writes into the control memory of the switching fabric an instruction to map a particular incoming timeslot to a particular outgoing timeslot. The switching can be unidirectional or bidirectional. The latter is needed for voice and requires two write operations into the control memory. Unidirectional switching is needed for example during a call to set up the voice path to the caller from the callee so that the exchange closest to the callee can send for example a busy tone or an announcement about the state of the callee to the caller. At this time, when the callee has not answered, the exchange closest to the caller typically wishes to keep the caller from using the voice path for any useful purpose by the users because charging starts only when B-subscriber has answered and the answer signal has been received. Therefore, the forward call path, in the exchange closest to the caller, is through-connected only at the reception of an *answer signal* from the callee.

Signaling and routing

In circuit networks we talk about *routeing*. In packet networks such as the Internet we talk about routing. *Routeing and routing are the process of defining the path that the call, flow or packet takes through the network from the origin to the destination*. Packet routing in itself does not need signaling. In IP networks packets are forwarded in routers based on forwarding tables that are created using *dynamic routing protocols* for collecting information about the network topology. Once forwarding tables are in place, it is possible to send packets from origin to destination. Because forwarding decisions are made packet by packet, when a forwarding entry is changed, all packet flows making use that entry will change direction. Routing protocols create "network state" but not flow state. Many user flows use the same entry in a routing/forwarding table.

In case of circuit switched networks signaling brings input data for routeing decisions. The data includes: the type of service requested, dialed digits etc. In addition, for the routeing decision a lot of configuration information is needed. A routeing decision is implemented in a switching node by writing a command into the control memory of the switching fabric. User data forwarding in switched networks is the same as switching. Routeing in circuit networks is based on *static configuration information*. There are no dynamic routing protocols for these networks. When the operator makes changes in the route tables, existing call paths are not affected. Rather the change affects only fresh calls. We call this property of circuit networks, *route pin-down*. In networks like the IP-network where routing decisions are done independently for each packet, route-pin down does not exist.

Types of exchanges

We classify Exchanges based on their position on the call path. An exchange to which subscribers are physically connected to is called a *local switch* (Central Office or Class 5 switch in US). A switch to which no subscribers are connected to is a *transit switch*. Transit switches can further be classified to *long distance* and *international switching centers*. The circuits between exchanges are called trunks. Therefore, also the term trunk switch is sometimes used.

For a given call, the local switch or exchange closest to the caller is called the *originating exchange* and the local exchange to which the callee is connected to is called the *terminating exchange*.

When digital switches were first introduced, it was a serious design challenge to make them big enough for the purpose of allowing building convenient network topologies. One can easily calculate that if it is possible to build switching nodes with the capacity in the area of tens of thousands of users, networks of any size can be built. This applies to both connectionless and connection oriented networks. When even bigger switching nodes became possible, it became even easier to build networks with tens and hundreds of millions of customers.

During the 1990's Moore's law took care of the problem. The result was that in digital circuit networks operators moved to larger node size. This helped to reduce software and other maintenance costs. Large switching node size led also to blurring of the categories of switching nodes. A large local switch may be used as a long distance and also as an international switch provided that the vendor is able to provide a rich enough software build for the node. A large switching node (build later than early 1990s), in addition to supporting originating and terminating traffic to some users, may support *transit traffic* for users that are connected to other switching nodes.

A structure of an exchange for circuit networks

Let us return to the example of an exchange with a Switching Fabric of 8192 PCM lines. It might be connected to several tens of other exchanges. A single control computer would most likely not be enough to take care of all signaling and call control load. One way of splitting the load (used e.g. in DX 200 of Nokia) is to allocate a fraction of those 8k PCM lines to a control computer in such a way that a prevalent or a single signaling system type is in use for calls using all PCMs of the fraction. E.g. 64, 128 or 256 PCMs all using the Number 7 signaling (we will describe this signaling system later in detail) would be taken care of by a single control computer. Because between the control computer and the PCM lines there is the switching fabric, it is possible to re-map those PCM –lines to another control computer for example if the original computer were to fail or needs to be taken out of use for the purpose of upgrading its software.

Figure 1.3 shows a high level structure of a local exchange or a switching system that provides access for subscribers. The subscriber lines are connected to line cards in the *subscriber interface units*. For availability performance and capacity reasons each subscriber interface unit might serve e.g. up-to 1000 subscribers (this is purely hypothetical, each vendor has its own dimensioning rules). So, a local switch serving 100000 subscribers would have something like 100 subscriber interface units. Each of them would use some number of internal PCM-lines for connections to the switching fabric. E.g. if each subscriber interface unit uses 4 internal PCM –lines, the total takes 400 PCM –lines from the switching fabric interface. If the control computer capacities are such that 80 (PC-like) control computers are needed and each would be connected to the switching fabric also by 4 PCM-lines, this would consume 320 internal PCM –lines. As a result, still more than 7000 PCM lines would be available for connections to other equipment and to trunks towards neighboring exchanges.

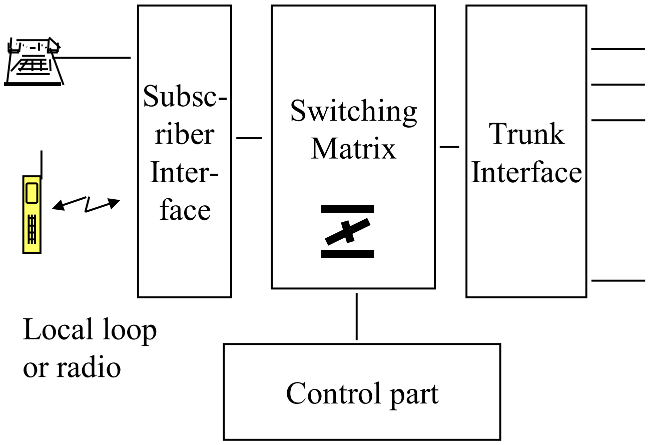

Figure 1.3: A high level structure of a digital switching system

Control part in Figure 1.3 consists of the control computers taking care of call signaling on the incoming and outgoing sides and also doing call control processing. Most often for a single telephone call one control computer would be dealing with the incoming signaling and incoming call control while another control computer would take care of outgoing call control and outgoing signaling. On both incoming and outgoing sides, the particular control computer would be decided based on the incoming circuit identity and outgoing circuit identity – recall that we assume that the mapping of trunks and internal PCM lines that carry user's calls to control computers is semi-permanent.

Like we discussed under call state, processing signaling and call control for a single call creates several state variables. These would reside in the respective control computers. During the process of setting up a call, the call progresses through many states reflecting the state of the communication on the incoming side, state of the switching fabric, state of the internal communications between different control computers, state of the outgoing call control and state of the outgoing signaling. The state of the switching fabric is replicated into memory so that it is easy to find free circuits. Moreover, reservation of a particular circuit from the fabric is recorded into a particular call's state information.

A typical *software bug* in an exchange is such that some resource is reserved for a call and never released. If the bug persists, at some point all the resources are exhausted and the system ends up in a deadlock. To avoid that, for example, we can set a time limit for any state. Such a principle would mean that the system will always strive towards an idle state even if explicit release of some resource does not happen.

For the purpose of controlling call traffic an exchange needs quite a lot of information. It is typical that *an exchange has a memory resident database*. In particular, *a local exchange has a memory resident subscriber database*. The exchange database stores information about the users and the topology of the network around the node. An exchange may also need to collect charging data about calls. Moreover, an exchange usually needs to be connected to a network management system, so that the node can be managed in terms of failures, configuration, accounting, performance and security (FCAPS).

Fault-tolerance requirements on switching systems

It is required in ITU-T standards and by operators that a switching system has a downtime of less than about 2 min/year (we talk about 5 nines availability performance = the probability of failure is < 0.00001).
Even a single *full restart* of a distributed system can easily take longer than 2 min. It follows that spontaneous restarts because of hardware or software faults need to be avoided for several years of operations on average. It also follows that many equipment blocks need to be duplicated or at least some level of hardware redundancy need to be provided. Typically, the switching fabric is duplicated. For example in DX 200, control computers that deal with signaling support either duplication or the so-called N+1 –redundancy schema.

To make use of the hardware replication, *signaling and call control computations* need to be replicated as well. If a control computer is duplicated, the *active machine* takes care of the signaling and call control load while the *hot standby computer* mirrors the state of the active one but is not visible to the outside world. In case, the active machine fails, the recovery system of the exchange orchestrates the *switch over* or *fail-over* of the duplicated machine. The target is that as few of the ongoing calls as possible see the difference because of the fail-over. This means that the fail-over must be fast. During the fail-over, the exchange should not miss any external event related to any call. A fast fail-over is achievable because of the mirroring of call and signaling state in the standby machine that takes over the load after the fail-over. However, it would be unfeasible to store the call states of calls handled by a control computer to a disk and restore the state after failover to the new machine. While the system would be retrieving data from disk, some event might happen in a neighboring exchange that our node under recovery would miss. Therefore, call state information is normally not stored on disk. Instead, call and signaling state is maintained only in random access memory.

The particular replication schema for equipment in an exchange is decided based on the impact of failure of the particular equipment. If for example less than 100 subscribers would be affected, probably no replication is needed. If the failure of a part would lead to full system restart or downtime of the whole system, no less than full duplication will suffice. For some functions it is sufficient to have one extra hardware block for many active blocks (we call this schema N+1). In such a case continuous active mirroring of the call state in the active blocks may not be possible. The spare block (or computer) is however useful in software upgrades and other maintenance operations. Also, the time for repair of a failed block becomes short. The calls controlled by a failed computer are lost but the spare computer taking its place will be able to take fresh calls almost immediately.

What happens to switch functions if calls are carried over an IP network

Like we discussed, signaling is still needed. Also a limited set of call control (or session control) actions are needed. These functions can be easily run on, for example, a Linux based control computer with Ethernet connectivity. Such computers are called for example call managers, call session controllers etc. Such a computer does not need to control a switching fabric. Instead voice packets are carried over the IP network and packet forwarding takes place in routers under the direction of IP-routing packet by packet. It may still be necessary for a session controller to control Media Gateways and some middle boxes such as NATs and firewalls.

In VOIP, voice is encoded into packets in the end devices and decoded back to voice also in the end devices. If the connection is over a packet network end to end, the network does not need to understand anything about voice coding. It is up to the end devices to agree on the use of any codec, standard or proprietary. Because codecs are just software on general purpose digital hardware and the cost of that software and hardware is borne by the user, it has become feasible to use many codecs.

Generalization: telephony paradigm

Let us take a high level view of the digital transmission based on PCM and digital switching from the users and operators point of view. This leads us to formulate the networking paradigm that is the foundation of telephony networks.

We can say that in telephony networks on user request the operator establishes a (64kbit/s) channel connecting the user to another counting the time for using the channel. On user request the operator releases the channel and stops counting. What the two communicating users do with the channel it is up to them. Based on the counted time, in each billing period, the operator sends a bill to the user.

For example, it does not matter that half of the time during a voice call each user is silent listening what the other is saying. The channel is still reserved in both directions and the cost of that reservation is billed usually to the caller.

From this paradigm it follows that keeping track of the channels is important. It is the task of call control. The networking paradigm in IP networks is quite different and the economics are quite different as well. When we move to voice over IP, it will be interesting to see how the paradigm shift will impact telephony services.

Differences between public and private switched networks

In public networks such as the PSTN, we call the switching nodes exchanges (puhelinkeskus). In private (or corporate) networks we call similar nodes Private (Automatic) Branch Exchanges or PABXs (PBX) (vaihde in Finnish). Although the functions of exchanges and PABXs are similar, there are significant differences. One should also note that the products used in public and private network are of different brands. The differences can be explained by the differences in requirements for private and public networking.

For example

- Maximum capacities of public switching systems are large, smaller nodes are enough for private networks,
- Large capacity leads most often to a distributed design while in a PABX even a single computer may be enough to handle all signaling and call control load.
- Availability performance requirements for the node and for many parts are stricter for public networks than for private networks. (We discussed earlier the famous down-time requirement of 2 min/year for public network switches). In a public exchange many subsystems such as the switching fabric, many types of control computers are duplicated or at least some replication schema is used to improve availability performance. In PABXs this may not be necessary to the same extent.
- Wired subscriber line requirements are easier for PABXs than for public subscriber exchanges (cmp indoor installations to outdoor).
- A very wide set of services (supplementary services) are provided by PABXs, usually a poorer set is enough for public networks.
- International or at least national standards for signaling are typical of public networks while proprietary variants of common standards are typical in PABXs.
- Charging is very important in Public networks, in PABXs call statistics may be collected but usually it is used for company internal purposes if at all in the corporation whose network is supported by the PABX. Duplicating statistics for outgoing calls in PABX (and the local switch) may also be provided and used but usually call charges are just a minor item in company costs while for the operator accurate call charge data collection is a matter of life and death.
- Used signaling systems are different. PABXs use private network signaling systems while public exchanges mostly support public

network signaling systems. A common denominator is also needed – otherwise the two could not talk to each other. Public exchanges may support international signaling and inter-operator signaling systems, these do not appear on PABXs.

- Both are moving towards voice over IP, PABXs are leading the way.

Internet Architecture for Voice over IP

The original Internet architecture followed the end-to-end principle as formulated by Dave Clark in 1984. *This principle says that we should not place into the network any function that cannot completely be implemented in the network. Instead, such functions should be left to be implemented in end devices or hosts.* It followed from this principle, for example, that communication error correction is taken care of by TCP in the hosts while the network concentrates on routing packets from one host to another. In the original Internet all hosts could directly see each other because each one of them had a static IP address. For the hosts, the network is transparent but if they wish, they can find out its structure by for example using the ping or trace-route utility.

All this is the idealized past. Today's reality of IP networks is different. Corporate and many residential users are behind NATs, so they use private IP addresses. We talk about either different address spaces or address realms. Uniqueness of addresses is ensured in one address space or address realm but not across any two address realms or spaces. Most users have dynamic IP addresses that are allocated at host startup or even later.

Most corporate users are behind *Firewalls*. These may normally let packets pass from the protected network to the public Internet and take care of letting the acknowledgements also come through. If however, someone wishes to take the initiative and first send a packet from the public network to a host that sits behind a Firewall, there is a problem in such packet pushing. The firewall is likely to block the pushed packet. This means that telephony that requires sending signals from the originating A-subscriber to the callee or B-subscriber is far from straightforward in modern IP networks. We will discuss NAT traversal in one of the later lectures in detail.

To recognize the existence of NATs, Firewalls and other middle-boxes, Dave Clark formulated the Trust-to-Trust principle in 2007. This says that *"The function in question can completely and correctly be implemented only with the knowledge and help of the application standing at points where it can be trusted to perform its job properly."* — David Clark, MIT Communications Futures Program, Bi-annual meeting, May 30-31, 2007, Philadelphia, PA.

The reality of IP networks in a corporate environment is such that every stakeholder wishes to hide its network from every other stakeholder. For this purpose they place *session border controllers or application gateways* on administrative boundaries. An application gateway processes all packets that go through it on application level. For telephone calls it will process call signaling and also the voice packets. The result is that a call may need to traverse e.g. 3…9 networks that are invisible to each other (on layer 3), traverse application gateways on the borders of those networks and also that end-to-end delay grows to unacceptable levels. It remains to be seen how these problems will be resolved.

Software Defined Networks

The classical Internet was based on the ideas that (a) the network has no flow state and (b) each node is autonomous leading to a distributed design of control functions and capability to recover on network level from link and nodal failures. These ideas can be also summarized that “routing is good – while switching is to be avoided”. It is a part of the picture that user data and network control messages are carried over the same links and the base network is not giving any preference to control traffic over the user traffic. Similarly, there is no systematic separation between “user plane” and “control plane”.

When the classical Internet design almost by accident escaped from the Lab (the University world) to the realm of commercial use, its weaknesses have started to emerge. We can list at the least the following:

a) IPv4 addresses are just 32 bits long so there are not enough addresses for all potential users.
b) IPv6 has been proposed and is promoted as a replacement but it has turned out to be difficult to move to the new version: it forces existing users and operators (with no growth in number of subs in sight) to invest time and money for the sake of some future potential customers of other operators in other countries.
c) Core routing tables (RT) in IPv4 Internet have grown to become large. The RTs consume fast power hungry memory in routers making them more and more expensive. The growth of the RTs depends on what the users want like more reliable connectivity (multi-homing) to the network.
d) The core protocol (IP) does not support mobility directly.
e) NAT traversal is cumbersome and existing methods are not well suited to wireless devices.
f) Source address spoofing is still possible although many technically valid solutions have been proposed and some of them are even widely supported by equipment vendors: once again investments/costs and benefits do not fall into the same hands.
g) Distributed denial of service attacks (DDoS) are possible and make it impossible to give strong guarantees of services being available in a predictable manner. Both spoofing and DDoS are inherent features of the traditional Internet.

h) It is difficult or impossible to guarantee quality of service (although solutions have been proposed and even implemented).
i) Etc.

Because the many issues in Internet design the topic of "Future Internet" became an important topic for research during the first decade of this millennium. One of the proposals that emerged from the research community is called "Software Defined Networking".

Contrary to the classical Internet design, SDN assumes that (a') "there shall be some flow state" and (b') centralized design of some network functions is fine and easier than distributed design. Also, the starting point for SDN is that control and data planes are separated. *Data plane carries the packets from a host to another host while the control plane carries the packets that are needed to control the state of the network.*

A motivation for SDN comes from the fact that more and more of the services we consume on the Internet are provided from the cloud. This means that of the two hosts in the basic network mediated interaction, one is the user's machine and the other resides "in the cloud". This implies that the identity of the last physical machine is unimportant – only the content or the service that is provided is important. To make services provisioning efficient and to improve the quality of experience of the user, it makes sense that the network somehow conspires to provide the service from a host as close to the user's host as possible.

The concept of SDN is already used in limited contexts. For wide area networks it is a topic for research. Based on our own work, I expect that the overall end to end network control functionality can and will be split into a set of network applications (SDN Apps) such as

- Base station (a lot of small base stations will need to added to provide more capacity)
- Mobile backhaul provisioning (a proactive App)
- Mobility management (reactive App – reacts to mobility events)
- Access and load balancing (firewalling etc)
- Edge to edge capacity provisioning (a proactive App)
- Edge to edge service delivery (a reactive App)

Different Apps on the above list have quite different requirements in terms of delay performance and scalability. Obviously, the above list is just an example of how the overall functionality could be split into manageable and understandable parts.

SDN relates to where the network functionality is placed and how the service is implemented. The functionality itself does not need to change at least initially. In the long run, it may be possible to optimize some of the network functions leading to higher level of automation of services provisioning and less operating expenses.

An example of longevity and continuity is what happened to the example switching system (DX200) we use on this course. The vendor has seized to design and produce the hardware but the software of that system is still used by running it on cloud/datacenter platforms. In 2014, it is still serving 1.6 billion users when counted based on the visitor location capacity.

So, initially, we do not expect that SDN would change for example many of the known signaling protocols. SDN will however have an impact of which particular nodes will talk many of the protocols.

Let us, however, make it clear that it is too early to say what will be the mid or longer term impact of SDN on the technology – these are early days of this new technology.

Network intelligence vs. terminal intelligence

We speak loosely about intelligence referring to the software and its role for service implementation. Traditional PSTN networks assume no software in phones, rather for a long time, PSTN phones used to be electro-mechanical devices. Sometimes, we label PSTN phones dumb.

The Internet, on the other hand, was from the very beginning built with the idea that the network is connecting computers and that the computers implement the services while the role of the network is just to transfer packets i.e. provide a bit-pipe. Two other major example networks that we will talk about are ISDN and GSM/3G networks. In the allocation of “intelligence” these networks reside somewhere between PSTN and the Internet. We will see on the ISDN lecture that ISDN actually makes poor use of the idea that terminals are computers although in practice they must be so. User devices in mobile networks such as GSM and 3G are battery powered, so power consumption of mobiles has always played a major role in the design of cellular systems. Nevertheless, one can claim that cellular systems make a fairly good use of the idea that mobiles are “intelligent” without going as far as the Internet.

However, software that was designed for mains powered Internet host computers does not necessarily work very well on mobile battery powered devices (even if the device would have the same operating system as a regular host computer). Applications that tend to be active all the time do not let the mobile device go to sleep mode and thus exhaust the battery quite quickly.

The historical trend in communication networks over the past several tens of years has been that “intelligence” is moving from the network to the user devices. One should expect additional moves in this direction in the future.

What comes to the remaining network intelligence, it tends to move to the edge nodes in the network.

Fundamentally, why is the intelligence moving to user devices? We point out a few reasons:

- User devices are created under *consumer market economies*. Volumes drive the prices down. As a result, a lot of more value to the consumer can be created in the terminal. On the other hand, network equipment volumes are several orders of magnitude lower.
- Users love to own gadgets. Branding of the gadgets leads to users being willing to pay several 100€ for their device. At the current and future level of electronics, a lot of functionality can be realized for that money provided volumes are high.
- Intelligent user devices facilitate service innovation by the users themselves. Of course users know best what they need. No single vendor or operator can compete in this understanding.
- Networks that allow and rely on intelligent user devices assign communication services value to the users – the users benefit. Networks that try to control accurately what the users can do aim to draw as much of the value as possible to the operator of the network. If the users have a choice they will choose to use dumb networks and intelligent terminals.

Scope of our discussion on this course

Switching and signaling form the infrastructure on top which communication services are implemented. Communication networking is one of the largest industries in the world. Networks are supposed to create social value, a part of which is realized in operator revenues and equipment vendor revenues. New networks require huge investments. A part of the value falls to the users who can work more efficiently or stay in contact with their friends and fellow humans.

In the era of digital communication networks, many networking systems have been created that have under-performed on the market compared to high expectations that were originally placed on them. This cannot be explained only by relative *technical* merits of the systems. Also, the roles of consumers, corporations and operators are important in shaping the technical solutions and the outcomes on the market.

For analyzing the economics of different systems we will not create a framework on this course – these subjects are discussed in depth on other courses given by Comnet.

Nevertheless, with this historical background when describing different signaling and networking solutions, we will discuss not only technical merits of the systems but also point out some economic reasoning behind the fundamental design decisions.

On this course we will start from systems that historically resided on the boundary of the move from analogue telephony to digital telephony and move through different digital telephony systems to packet based telephony that is currently under development and early deployment. In explaining the technology we will try to more or less consistently pay attention to a set of issues related to each new signaling system.

We can summarize the goal of the course in the following way:
Signaling will be analyzed on a functional level. Focus is on understanding advantages, drawbacks and fundamental design aspects of widespread solutions in different networks. *The idea is to learn something about the art of systems design for networking.*

Through understanding several signaling systems, we consider how such systems interwork. By comparing circuit and packet networks we build understanding of the technology trends and how functionality from legacy networks is inherited into new systems.

In our discussion we cover *constraints* and *requirements* and high-level design of the signaling systems. The challenge is to understand how a particular design appears based on the constraints and the requirements. This transformation is far from trivial. We hope that by discussing several examples we can gradually build a somewhat generic understanding how requirements and constraints are transformed into real systems.

Let is now formulate the idea of the translation in a pictorial form.

From requirements to design

Figure 1.4 depicts the key categories of requirements and the translation into the design space.

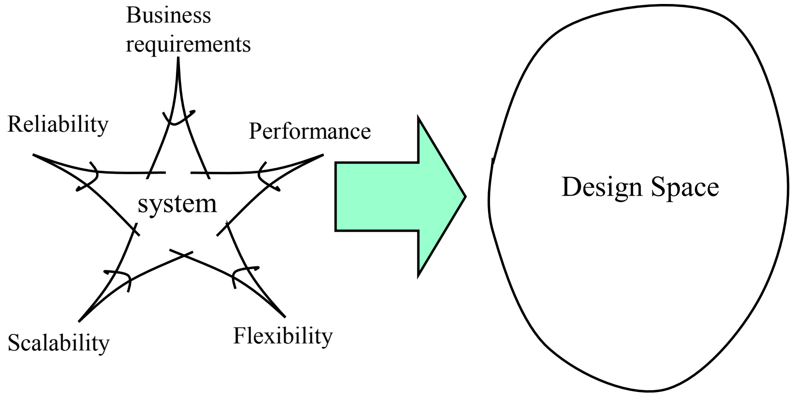

Figure 1.4: Translating requirements into design.

Classification of Requirements

First of all, we discuss different requirements that were formulated or implied for a particular signaling system. Figure 1.4 shows the classification of the requirements that we will pay most attention to.

Business requirements
These include e.g. the motivation of different players behind the design or level of control over the resource use by the operator or charging for all communication between users or how the design allows using different vendors. For modeling the dynamics of competition, a *stakeholder model* can be used. Stakeholders include network operators, service providers, subscribers, users and vendors. When looking at the relationships of the stakeholders, one aspect that is important is *trust*. Trust is an expectation of the trustor to the trustee that the interests of the trustor will not be betrayed by the trustee. Often it is impossible to assure trust completely. Trust then assumes that the trustor is taking and willing to accept a risk.

Behind the business requirements lay the different methods and ideas of how each of the players competes on the market of communication services. I.e. they are about the ways to earn money. Most business requirements are non-numeric and thus not quite straightforward to model.

Performance requirements
Performance can be characterized e.g. in terms of number of signaling bits generated by a call or post dialing delay. Most often, performance can be measured in *units of time* that an action takes. One should be aware that high performance is usually in contradiction with all other nice requirements.

This requirement relates to the costs incurred by using the signaling system. Performance can be enhanced by keeping the system structure as simple as possible and making sure that only absolutely necessary requirements are set on each component. As a result, each component is simple.

Performance is most often a numeric requirement. Thus mathematical modeling of performance is straightforward and over the years has attracted a lot of attention. In practical engineering one should first pay attention to orders of magnitude in performance. Gains in performance in the order of a few percent (e.g. less than 30%) are usually not that important that they would be sufficient grounds for significant changes in design. Such small gains that can be achieved with existing equipment may however be welcome.

Reliability and Availability Performance.
Reliability is the probability of correct operation (no repair is allowed). *Availability performance* is also probability of correct operation but allows

repair. Reliability relates to perceived quality of service supported by the system and also to efficiency of making use of different resources that are controlled by the signaling system.

Reliability of a system is determined by the reliability of the underlying hardware such as Fibers, radio transmission, memory, CPU and other electronics but also by the software design. In principle given the types of components for building a system, reliability can be enhanced by *redundancy*.

Redundancy can be in terms of space or time. Instead of just one computer, we can use two of the same kind for the same job, one active and one spare. We can add redundant bits into each memory word. We can send redundant bits in a message for error detection and correction. We can repeat a message if the original was lost. Instead of sending blindly irrespective of the state of the receiver, we can intelligently set the limit how much we send before an acknowledgement is required etc.

Reliability is also a numeric requirement lending itself to mathematical modeling. This however has attracted less attention than performance modeling and analysis in the history of network engineering. However, reliability engineering is a fairly well developed area of engineering and has been applied for example to build robust flight control systems for avionics and spacecraft as well as control systems for nuclear power stations.

Classical results show how the reliability of a complex system can be determined given the reliabilities of the parts forming the system. Conversely, given the target reliability of a whole complex system, acceptable failure probabilities can be allocated to the parts. Based on such analysis the designer can decide whether a particular part of the complex system needs to be replicated or some form of redundancy needs to be applied.

The requirement of fast recovery from failures leads to duplication and the need for fast fault detection. The latter may be quite difficult to achieve in a complex system that may fail because of hardware or software faults, configuration (data) errors or user action. A software implementation of recovery from failures assumes a certain *fault model* that captures the properties of failures that can be detected by the means available in the system.

Systems with redundancy exhibit the property of *fault tolerance*. Fault tolerance is the capability of the system to continue its specified (or intended) behavior in the presence of failures. Usually, this means that faults must be detected and failed components must be automatically isolated from the rest of the system. There may also be a separate system for *fault location*. It can exercise the different functions of the suspect or failed component (while it is isolated from the rest of the system) and suggest possible corrective actions for the human user. The faster the system is repaired the higher will be Availability Performance.

We can look at reliability on a node level or on the network level. Redundancy on the network level implies for example alternative paths or routes (e.g. multi-homing of customer IP networks with the public Internet), routing convergence, route restoration etc.

Commonly used mechanisms are also load sharing and load balancing. Load balancing shares load on a task-by-task or flow-by-flow level while load sharing usually relates to a more permanent arrangement of splitting the load (for example for call processing) among the available subsystems/computers or nodes.

Scalability

Scalability answers questions like how many subscribers need to be served, or how to scale the function into a global network with many operators. This requirement relates to meeting the growth on the market when the signaling system is in use. One can claim that scalability is actually a derivative requirement from performance, reliability and the possibility of parallelization.

Because those factors can be modeled mathematically, also Scalability lends itself to mathematical reasoning. However, the important thing is to ask the right questions because the interesting questions imply some growth scenario that is unknown. Once the right question has been formulated, finding answers using some simple mathematics is usually much easier.

What comes to scalability as a requirement, we can claim that it is invariant i.e. it has always been important and the importance remains very high. The phenomenal growth of many services like the Internet and cellular mobile networking confirm this invariance.

Flexibility

Flexibility of a design defines e.g. how easy it is to add new services or meet new requirements.

One way of ensuring flexibility is to use a modular design. For example this calls for protocols that each solves a compact problem.

Flexibility is needed because over time the use scenarios of a system tend to become more varied. In ICT it is typical that all systems and technologies are gradually stretched until the system becomes so messy and complex that it is too costly to use and needs to be replaced by a new system or technology.

Most obviously, flexibility is in conflict with the requirement of high performance. Also, it is an engineering challenge to come up with a robust design that is also flexible at the same time. Over the years of Moore's law, engineers have been sacrificing performance for the sake of flexibility.

Flexibility is a non-numeric requirement. It does not easily lend itself to modeling. Engineers tend to argue about the flexibility purely based on experience of using different approaches and based on some form of qualitative reasoning. But let's face it, how can we know what the future requirements might be? Since we cannot reliably predict the requirements, we can never be sure that the design can be adapted to them. On the other hand given two competing designs on a market, the one that cannot adapt, or is slow to adapt to new user requirements will fail on the market.

When we move from older systems to newer, we will see how the weights of these requirement categories have changed over time. Due to Moore's law, performance problems have become easier to solve lowering the importance of performance as a requirement. At the same time, the importance of time-to-market has become higher and higher in all ICT services and systems. This is because of the experience that "winner takes all" that has been seen many times in the ICT business. Fast time to market has increased the relative importance of flexibility in networking system design over the years.

The Internet is less reliable as a network than the ISDN network. This has been acceptable to the users. Generalizing, if a new service is really attractive from the user's perspective, it can be acceptable to the users that the service is less reliable than comparable legacy services. However, on established markets, reliability requirements on products and services tend to be invariants in the long run. Therefore, a less reliable design most likely must lend itself to gradual improvement in terms of robustness.

Design space: Mind map for learning protocols

Based on our experience so far, let us try to generalize the subject of protocol study into a generic mind map. *The mind map depicts different aspects of how a protocol works. It describes key aspects of the design space.* The idea of the mind map is to give placeholders for different issues that are relevant when discussing protocols. We present our first attempt of such a mind map in Figure 1.5

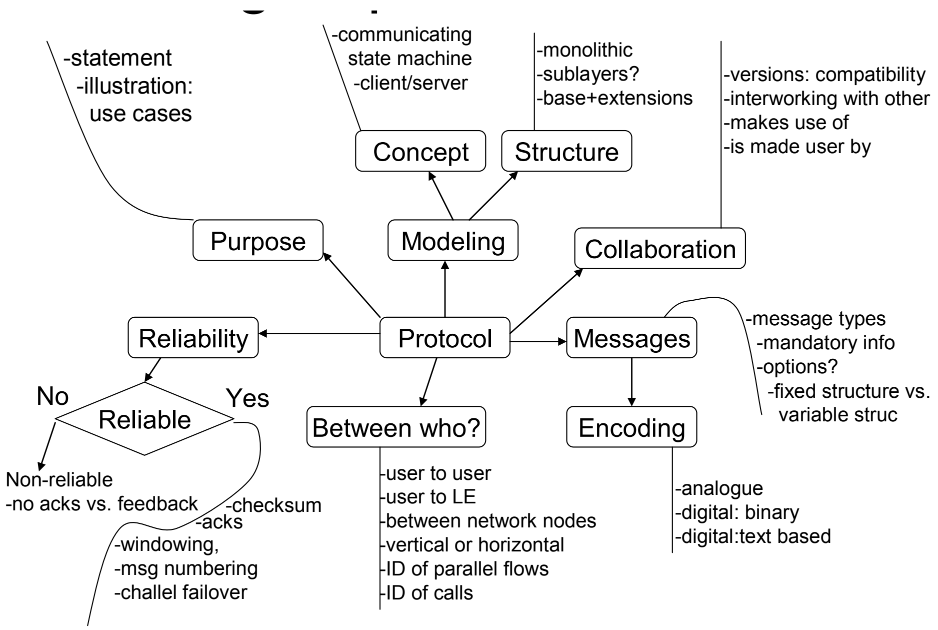

Figure 1.5: A mind map for protocols.

Let us comment the design space against the generic requirement categories.

The stated *purpose* and the aspect of the communicating parties (between who) relate to the business requirements. It follows from the purpose what kind of information is carried by the protocol and between which parties, entities or devices.

Identification and addressing are the first technical things to look at any protocol. They tend to set the strongest constraints on what the protocol can be used for. I.e. they relate very strongly to the purpose of the protocol. Methods of identification are also very difficult to change in a given design because of loss of compatibility with older protocol versions. One needs to identify things like the communicating parties and parallel protocol message flows etc. in a protocol. Many protocols carry some sort of addresses. An address is an identifier of a location in a network or an interface in a node or device.

An important case of identification in a network is how the user devices or subscribers are identified in communication protocols. Examples are host identities in IP networks or subscriber identities in cellular networks. Identification of subscribers and user devices is important because the user devices are physical endpoints of communication and subscribers are the logical end points.

Many protocols assume a *session model*. A session has a beginning, some operations and the ending. Sessions need to be identified since it is typical that a network node must be able to run many parallel sessions. The messages in a session are said to belong to a flow. For example, all signaling messages related to a single telephone call form a flow. We can talk about the size of the flow and about its duration. The size of a flow can be measured in octets. A message flow may assume a maximum time distance between two consecutive messages in the flow. More generally, a flow is a set of packets such that a certain time constraint applies to the packets that all meet a certain match criteria (have a certain bit pattern).

Modeling and structure most of all relate to the requirement of flexibility. Compatibility determines how easy it is to upgrade the protocol that is live and working in a network carrying some work for users at the time of the upgrade. (We should realize that it is not possible to stop a whole public services network for the purpose of software or configuration upgrade – rather the software must be possible to upgrade one network node at a time). Other aspects of collaboration relate to the place of the protocol in a broader architecture. These aspects tend to follow from the purpose of the protocol.

Messages encoding and structure tend to have an impact on the flexibility of the protocol, i.e. how easy it is to add new features to the protocol in order to meet new emerging requirements.

Reliability: There are two broad categories of protocols: (1) unreliable and (2) reliable. For example, IP or UDP/IP are unreliable protocols. They do not give *guarantees* of any kind that the information sent is actually received.

Reliable protocols try to guarantee delivery of the information that was sent. They make use of numerous mechanisms to implement the guarantees: most common is that a delivery is acknowledged by the receiver. This is a time redundancy mechanism: reliability is achieved due to additional time it takes to send and receive the Ack and resend the original message. For easy message bit error detection both protocol types use either header checksums or checksums over the whole message. Naturally, checksums do not help if a whole message is lost which is actually more common than the occurrence of message bit errors. In addition, protocols may number all sent messages or octets. Most reliable protocols may also have some failover mechanisms. *Windowing* is a quite common mechanism for increasing reliability and achieving high performance at the same time. The idea is that the sender numbers all sent messages by sending an increasing message counter in each message. In Windowing, the communicating parties may have agreed on or the protocol may imply that only some number *W* messages can be sent before the sender must see an acknowledgement from the receiver. The Acks are also numbered so that in all situations the protocol can quickly recover from message loss.

Scalability is determined for things like the length of different fields of data in the protocol, such as the addresses, user identifiers, channel identifiers or flow identifiers. In addition, reliability and performance have an impact on scalability.

Higher flexibility usually leads to lower performance and efficiency.

We will use this framework of requirements and design aspect to determine the coverage of our discussion of different signaling systems. Depending on the case different aspects of the framework will get more or less attention.