Section 2

Concepts and the ISO/OSI reference model

***

Pierre Rolin

Lecturer at Institut Mines-Telecom

***

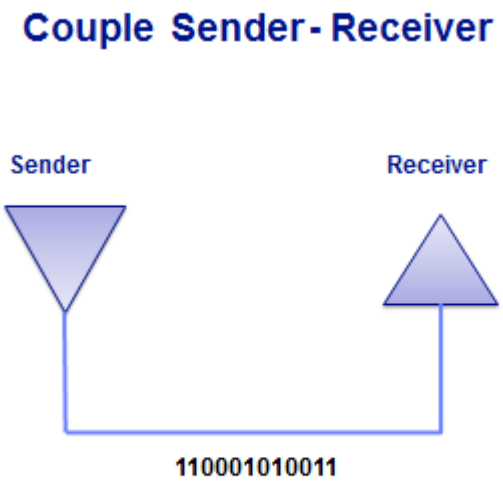
Hello, my name is Pierre Rolin. I will be taking you through the second Section.

Section 2, Lesson 1: Notions of communication channels

A communication channel is composed of 3 parts:

- an emitter that sends the data,

- a transmission channel and



The aim of every communication network is to allow communication channels to be

built between users.

We can illustrate this model by a physical connection comprising a telephone pair cable, for example.

The copper telephone cable constitutes the physical communication channel. The emitter modulates its data, for example in the form of electrical signals, and the receiver demodulates it to obtain the information sent.

The most elementary form of information is the bit, or binary digit, the digits 0 or 1. On a physical level, computers work only with these two digits.

The emitter device therefore has to transform the bytes it reads in the computer's memory into a series of 1 and 0 contained within these bytes. We say that the data has been serialized. At the other end, the receiver must re-assemble the bits it has received into bytes that to store them in its memory.

The electronic device that carries out these two functions is called a Modem, for Modulator De-modulator.

The **propagation delay** along the copper cable is inversely proportional to the propagation speed of electricity along copper (V_p) and the length of the cable (L), L/V_p. Propagation delay affects end-to-end transmission delay.

Physical links examples

Sender	Channel	receiver
Electrical modulator	Telephon cable	Electric demodulator
Modulator électronic -> optic	Fibre optique	Demodulator optic -> électronic
Modulato rélectronic -> antena	Freqency	Demodulator receiving antena -> électronic

propagation delay = Cable length / V_p

Other physical materials can also be used, however, such as optical fibers and radio wave frequencies (Wi-Fi, mobile telephony etc.). Ourself when we speak, we modulate the air ourselves.

But be careful, communication channels are not limited to physical channels.

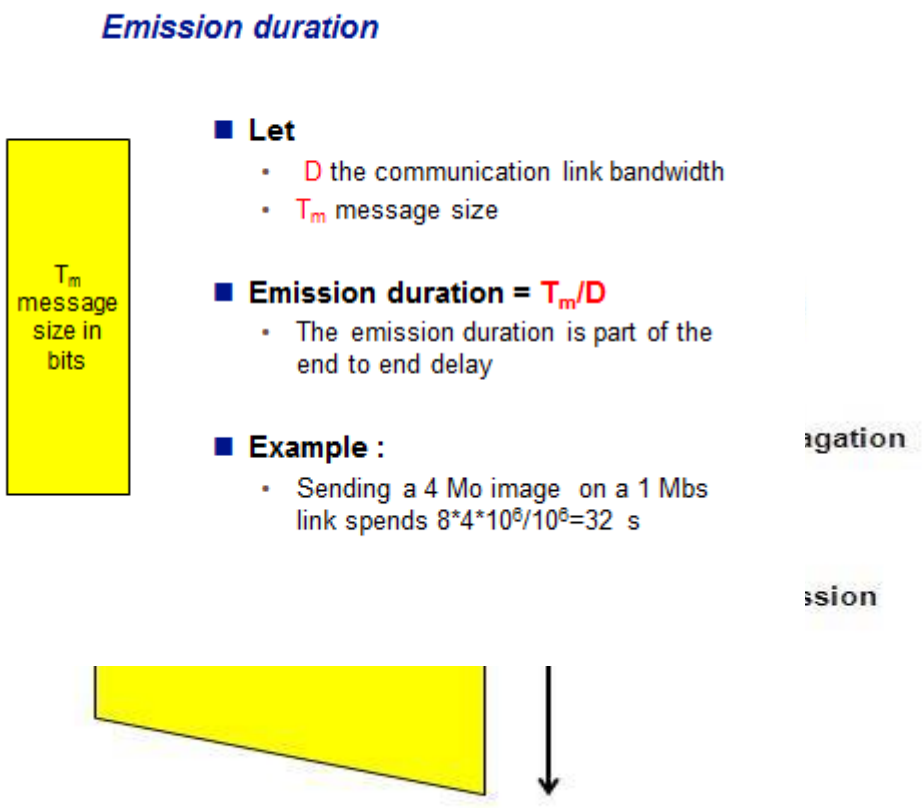
Bandwidth units

Units	Notation	Numerical
Bits / second	B/s	10^0
KiloBits / second	Kb/s	10^3
MegaBits / second	Mb/s	10^6
GigaBits / second	Gb/s	10^9
TeraBits / second	Tb/s	10^{12}
PetaBits / second	Pb/s	10^{15}

Transmission duration = T_m / D

The bit rate is the number of bits emitted per second along a communication channel. We shall often denote it as *D*. Bit rates are denoted in multiples of 10^3: bit/s, 10^3 => Kb/s (kilobits/s), 10^9 => Mb/s (megabits/s) etc. Over the past few years, bit rates have increased dramatically.

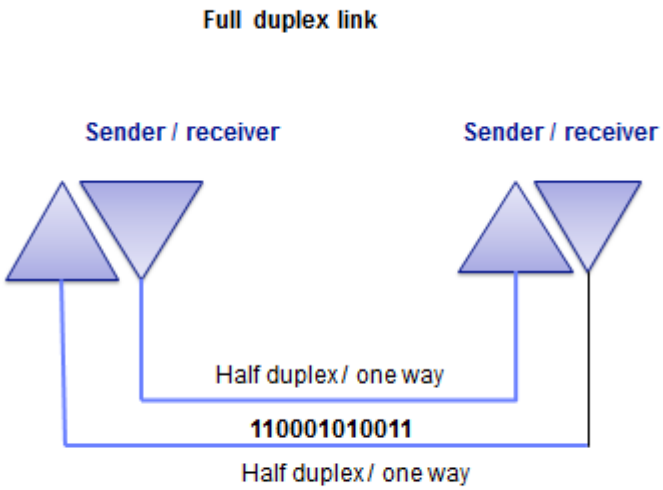
N.B.: In computing, due to memory addressing, data quantities (like file size, for example) are in powers of 2: Kilobytyes 1024 = KiB 2^{10}, Megabytes = MiB 2^{20} or Gigabytes = GiB 2^{10}.



The transmission delay along a physical channel is the sum of the propagation delay + emission time.

Owing to the propagation delay, the emitting modem and receiving modem do not work on the same piece of data at the same time. Since the devices are remote from each other, **they have no common clock** like in a computer. The two processes of sending and receiving are called asynchronous. The two work in parallel.

This brings us back to the parallelism we have already discussed but, in contrast to a computer, with no possibility of a shared memory or common clock.



The channel drawn the diagram at the beginning is one-way. We can also say **simplex**. Examples of simplex channels are radio channels and satellite channels.

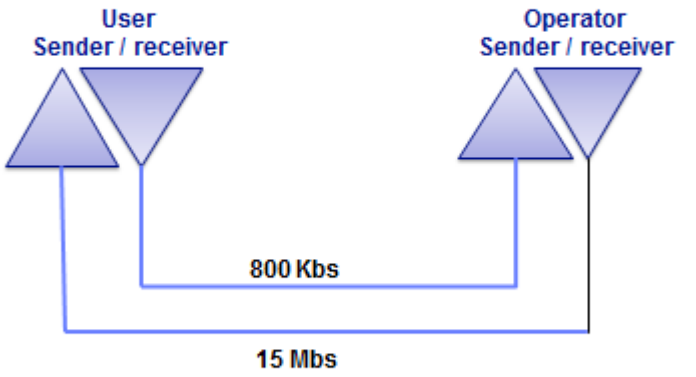
A **half-duplex** channel is a simplex channel used alternately in one direction and then another.

Most communication channels are bidirectional, or **full-duplex.** A modem allows emission and reception in order to establish a two-way channel. The emission and reception functions work in parallel.

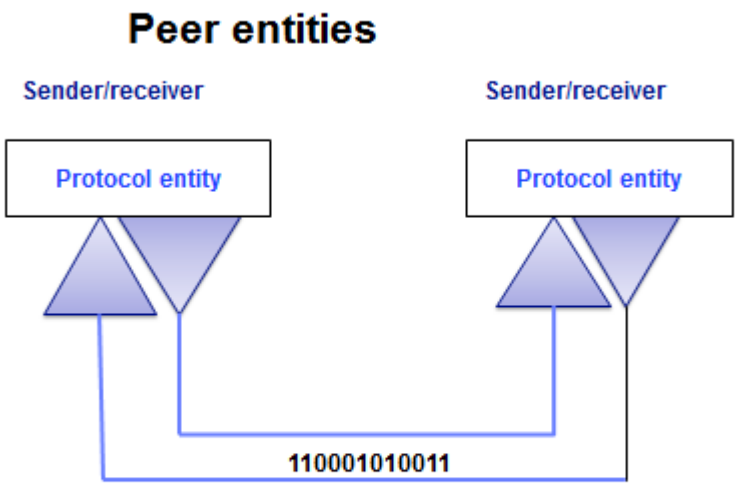
However, each direction in the channel can have its own bit rate, as in the case of ADSL where the download speed from the operator is greater than the upload speed to the operator.

aDSL full duplex link

The babdwidth in each direction can be different



Section 2, Lesson 2



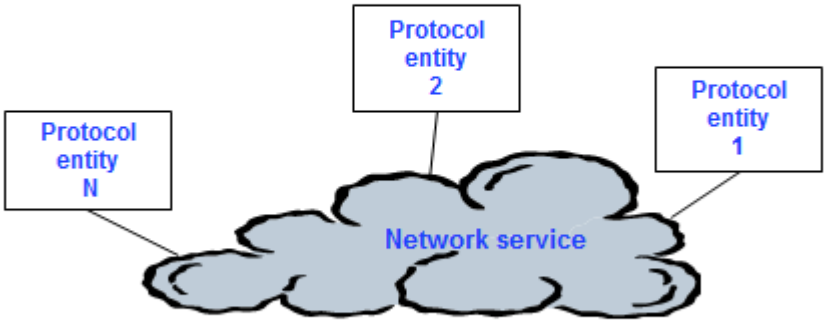
A communication channel is never perfect. When we discussed the postal service, we saw that errors may occur, that data is organized into packets, that the delivery time is variable, that multiple recipients can have the same letter box…

To improve the service offered by a communication service, and hence the communication channels it provides, a set of rules are determined and implemented. These rules are called a communication “protocol”.

We shall refer to a software or hardware component that executes these rules as a “protocol entity”. A “protocol entity” applies a particular protocol. There are hundreds of different protocols in networks.

Two protocol entities that execute the same set of rules at either end of a communication channel are called peer entities.

In the image we have two peer protocol entities situated at each end of the same channel.



The aim of a network is to allow communication channels to be built between any pair of subscribers. It is also possible to create multipoint channels between groups of subscribers, but we will not be looking at this in this lesson.

So, we can illustrate the communication channel by a cloud representing the network.

The peer entities are subscribers of this network service.

The rules that define a protocol are described in great detail in international standards and norms.

Putting a protocol and a network service together creates a new network service.

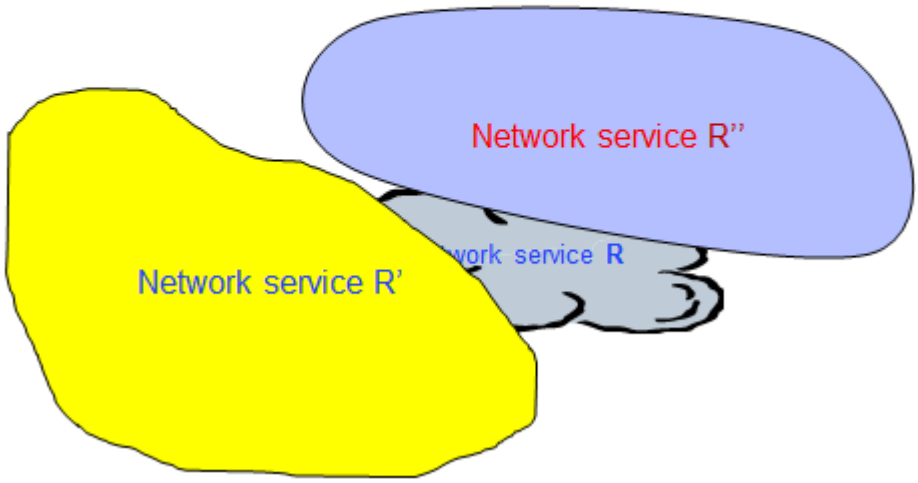
It is therefore possible to create as many services we want by combining services and protocols.

Combining a communication service S and protocol P creates a new communication service S' with different properties to those of service S.

We can also talk of protocol layers. All the entities that apply the same protocol over a single network service form a protocol layer. These protocol entities are therefore all peers. In each device, there must be an instance of the protocol entity of that layer.

You have probably heard the expression "IP layer". IP is the protocol that supplies the "Internet Service". The IP entity must be run in every device that aims to offer its local users the "internet service". All these entities make up the IP layer. In Section 4 we shall see what the IP protocol is.

The network service R associated to protocole P1 creates a new network service R' and associated to protocol P2 a network service R"

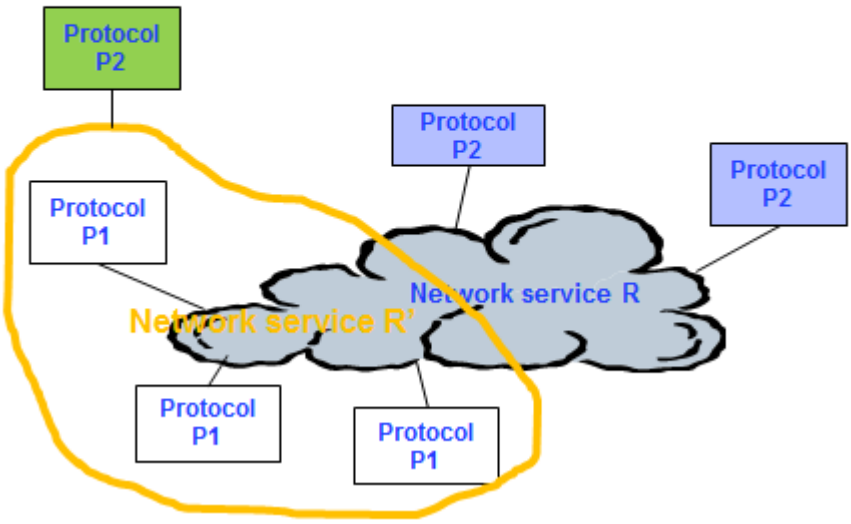


A network service is generally shared by several users who can all use it simultaneously.

Notably, several protocols can use the same network service.

In this diagram there are therefore two protocols, P1 and P2, which are able to use network service R. When combined with protocol P1, R creates a new network service, R'. When combined with protocol P2, R creates a different network service, R''.

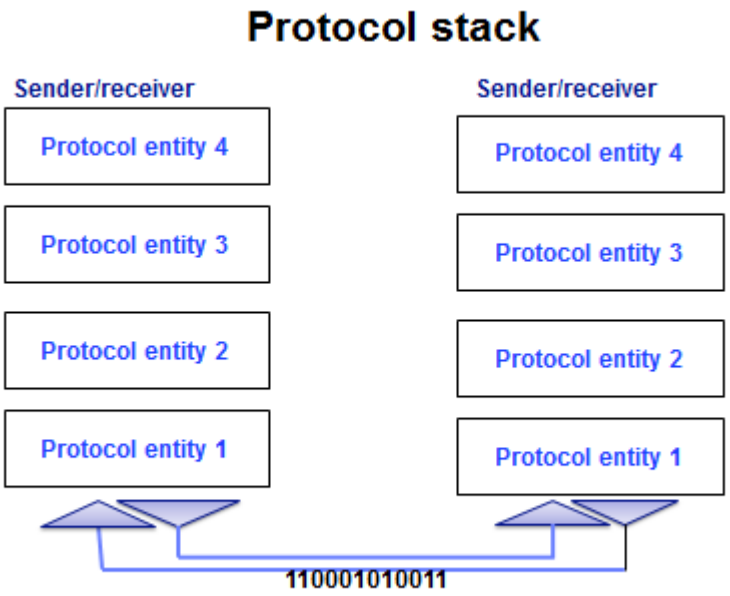
The green protocol entity is not peer with thes blue ptocol entities
These 3 protocol entities operate the same protocol P2
Butgreen P2 entity is over network service R' built over network service R with protocol P1 while the blues entities P2 are directly over network service R



Identical protocol entities, meaning entities that apply the same protocol, are only considered peers if they are connected to the same network service.

So in this diagram, the green protocol entity is not a peer of the blue protocol entities, although all three protocol entities apply the same protocol, P2.

But the green P2 entity is on the R' network service built above the R network service with the P1 protocol, whereas the blue P2 entities are directly on the R service.



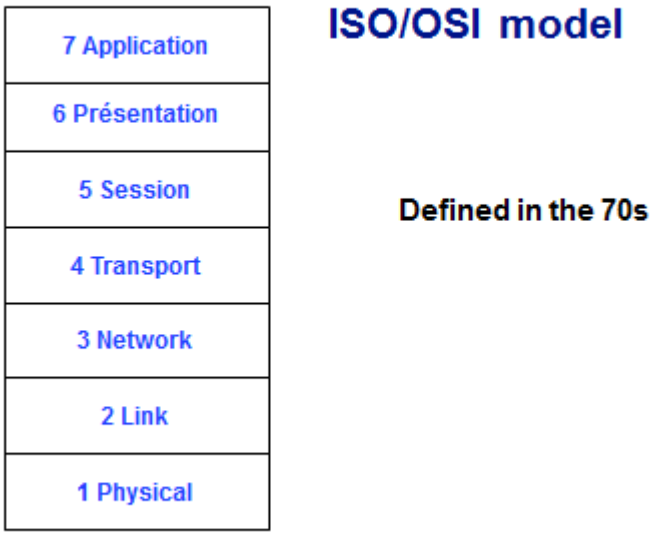
A user may want a service that guarantees the properties he needs.

For example for cash machines in a bank, the communication service must ensure that the data arrives in the order it was sent and without errors, that it remains confidential and that the transmission delay does not make the client wait too long.

Building all these properties into a single protocol entity is not a good idea. It is simpler and more effective to design a protocol for each service requirement.

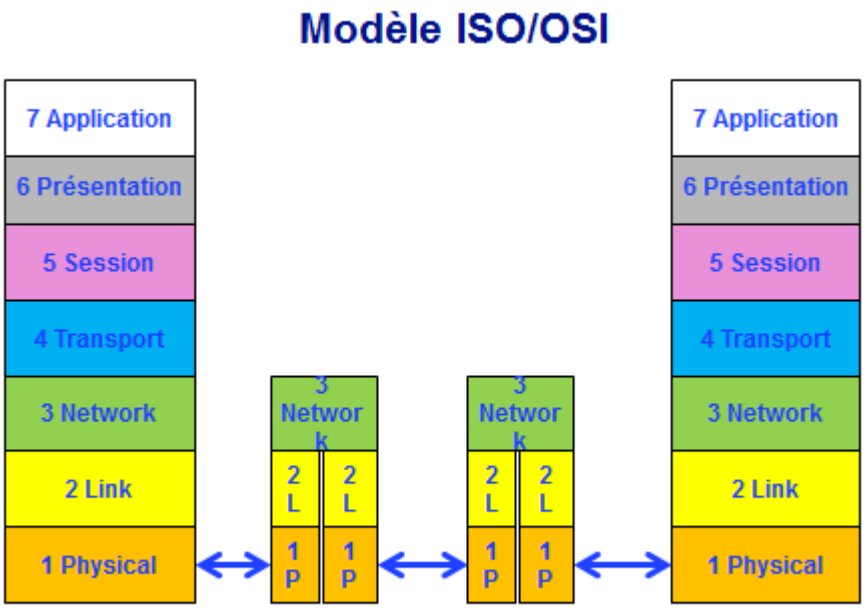
Once we have a protocol for each of these services, we simply need to **assemble the different protocols into a hierarchy** to obtain the final service that offers all of the

desired functions. We say that this assembly is in **layers**.



At the end of the 1970s, the International Organization for Standardization defined the Open System Interconnexion model, known as the ISO/OSI model. It is still serves as reference in data network design today.

The model has 7 layers. Within each layer, various norms and standards define the services provided.



- The **physical** layer defines the material specifications: metal, optical, radio waves or microwaves; the strength and range of the signal; cable lengths; the shape of the sockets; the encoding used to transmit bits along the cable.

- The **data link** layer controls the transfer of information between devices along the physical channel. It generally detects errors and can sometimes apply corrective mechanisms.

- The **network** layer handles the function of transmitting the messages to their destination. It carries out routing, relaying or switching of messages. It must be able to calculate and find paths, and in order to do so it must have several data links with neighboring network entities up to the final destination. Multiple different network services exist. In Section four you will see the IP protocol.

- The **transport** layer provides transparent data transfer between users by relieving them of the execution details. Multiple different transport services exist. In Section five you will learn about the TCP and UDP services.

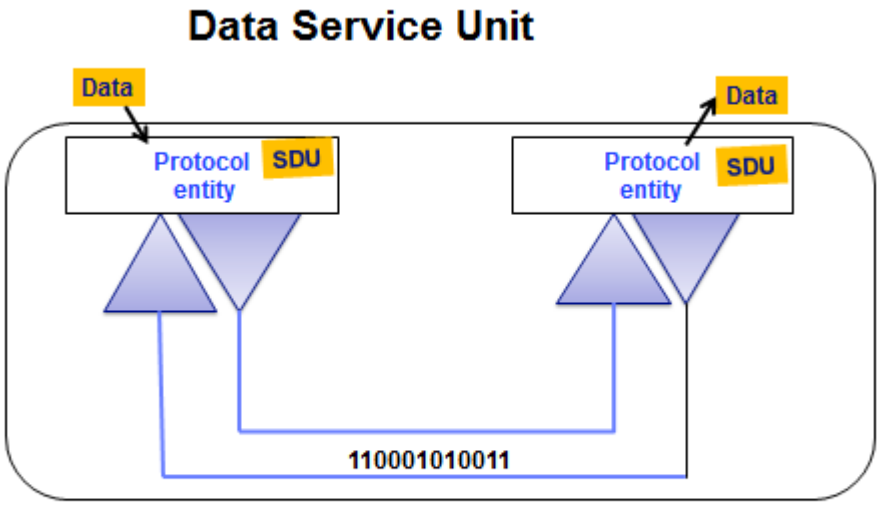
- The **session** layer aims to provide synchronization methods. It is rarely applied above transport.

- The **presentation** layer manages the representation of information sent amongst the application entities. It is needed due to the variety of forms of data representation within the different systems.

The reality of networks has played a major role in the way the functions are placed in the layers. Read the document in chapter 3 of the book "*Les Réseaux*" that you have for more information.

In this lesson we have considered the notions of peer entities and the ISO/OSI reference model. In the next lesson we shall look at the notion of protocols.

Section 2, Lesson 3

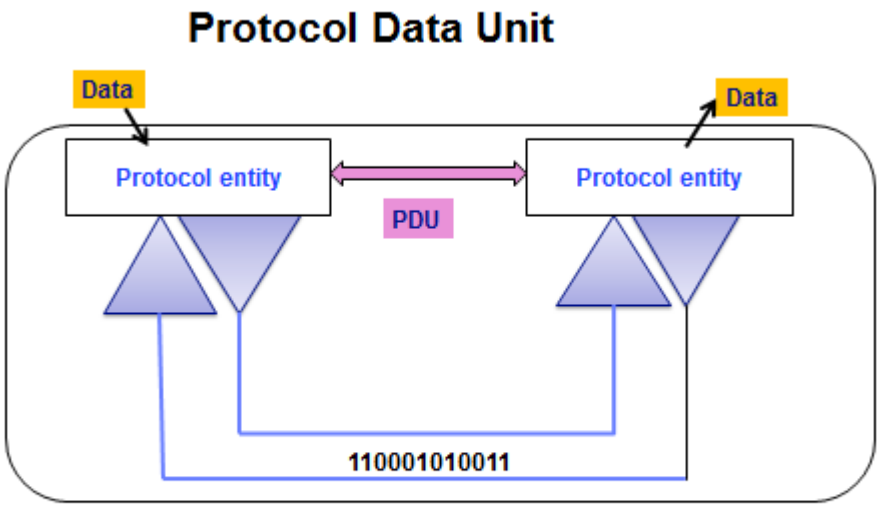


In the previous lesson we looked at the ISO/OSI model and the notion of peer entities. We are now going to study the notion of protocols.

To execute the rules defined by a protocol, peer entities need to exchange information.

In the protocol entity there is, as a minimum, the information that the user wants to transmit, which we shall call the **SDU** (for Service Data Unit).

In general, the aim of a communication service is to deliver the same piece of data, the same SDU, to its destination, so that what enters and leaves each end of the communication channel is identical.

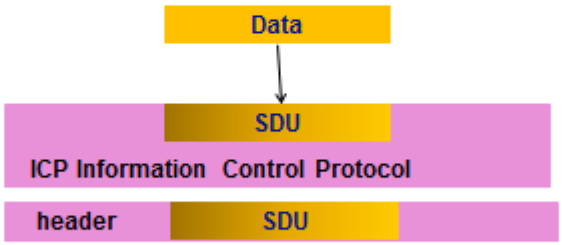


In order for the communication service to work, the two peer entities exchange messages that we shall call **PDUs,** for Protocol Data Units.

In addition to the contents of the SDU, these PDUs contain information that is needed by the protocol in order to build the properties it supplies. So, for example:

- A protocol that makes sure the SDUs are delivered in the right order will give them the numbers of a sequence.
- A protocol that ensures that SDUs with errors are not delivered will introduce a code for error detection.
- A protocol with a delivery report serving, like in the postal service, to inform the sender that the SDU has been delivered, will insert “delivery report” PDUs. These PDUs travel in the opposite direction to the data.

Service Data Unit
/
Protocol Data Unit



Two PDU représentations

We shall call all this information added by the peer protocol entities PCI (Protocol Control Information). We can also say header, because the information is often placed ahead of the SDU in the PDU. There is some often added at the end as well, such as a code for error detection. In this case we refer to it as a trailer.

This data is only used by these protocol entities. Users of the service have no knowledge of it.

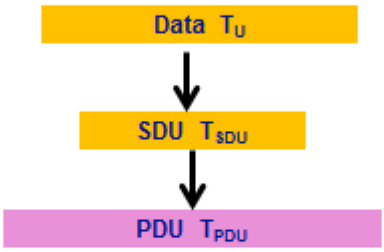
All the additional data has to be placed somewhere, so it is adjoined to the SDU. If the PDU carries all the SDU data, the protocol data is added, but if the PDU is only used to manage the protocol, we refer to it as a protocol PDU.

It is also possible for protocol entities to perform operations on the data. So, the data submitted can be transformed in order to form the SDU. This is the case, for example, of a protocol for data compression or for encrypting or presenting data. It is of course also the case in the physical layer where the data must be coded and modulated in a physical quantity. Next Section you will see an example of data modification in the case of the HDLC protocol.

This protocol control information that we called PCI is often called an “envelope”. We also say that the data is encapsulated in the PDU.

The description of the ICP (often called header) content is always described in a standard and is often called **frame format**.

Useful bandwidth / protocol efficiency



Protocol efficiency or

Performance ratio : $R = T_U / T_{PDU}$

Useful bandwidth: $D_u = R*D_{voie}$

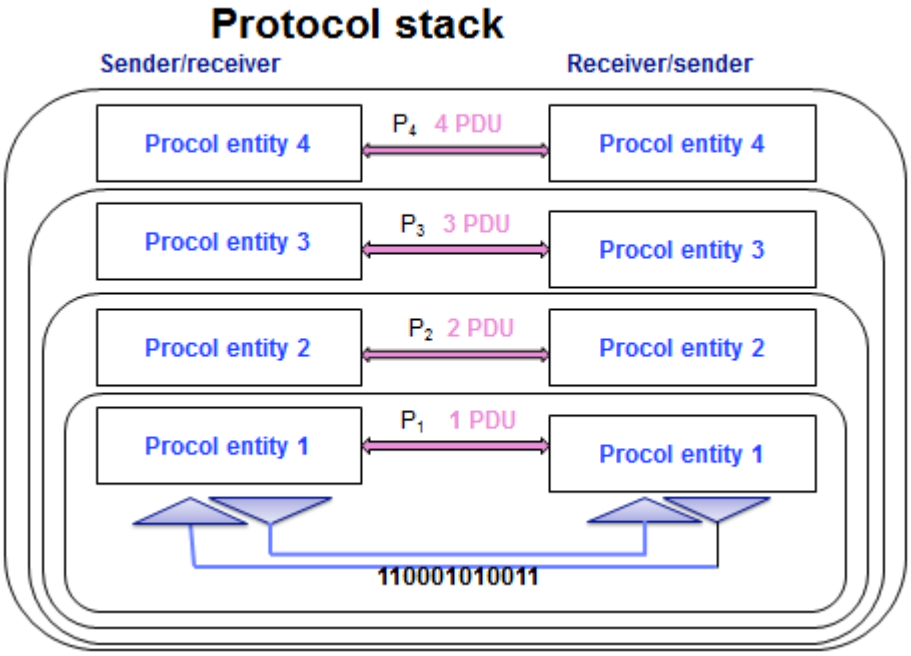
Let us take a look at the effects of this encapsulation mechanism on the use of the communication channel.

In general, the size of the PDU is greater than the size of the data. An overhead is inserted. Therefore, the useful bit rate for the user, D_u, is the quantity of useful data, T_u, transmitted per unit of time. If T_u represents the size of the data sent and T_{PDU} represents the size of the PDU, a simple preliminary definition would be that the performance ratio (efficiency) R of a protocol = T_u / T_{PDU}.

We can therefore see that the bit rate provided by a protocol will not be the same as the bit rate of the channel used by the protocol.

Usable bit rate = R * Channel bit rate

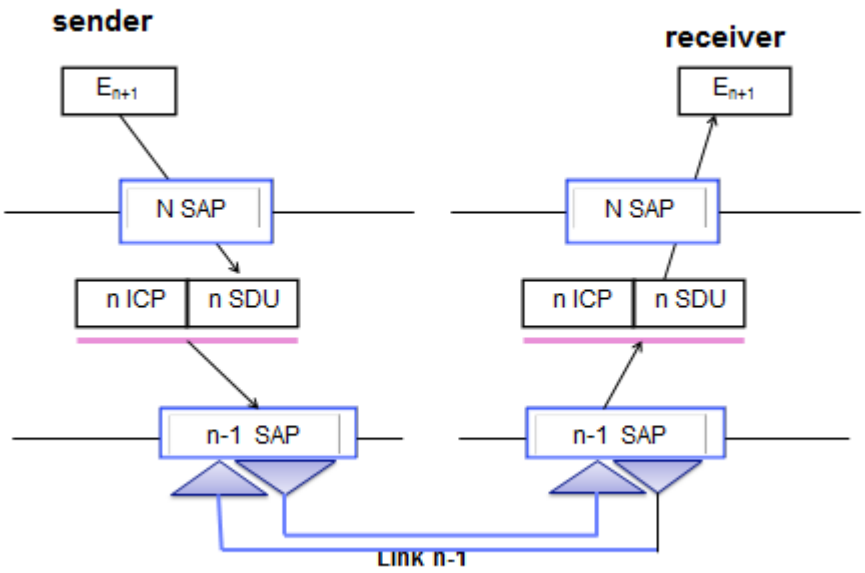
However, you will see in the rest of the course that this definition based solely on the size of the messages is inadequate.



We can now add these protocol messages to our protocol stack.

Each layer uses its own specific PDUs. For layer “i”, we shall refer to them as “i PDU”.

The i PDUs travel in both directions between the peer entities of layer i.



Consequently, the size of the PDUs changes until they reach the physical layer, which is the only level where information is actually transmitted to the remote device.

The data E_{n+1} is supplied by the user located in the layer immediately above. A PDU of protocol n is composed of the protocol control information “n PCI”, and the n SDU.

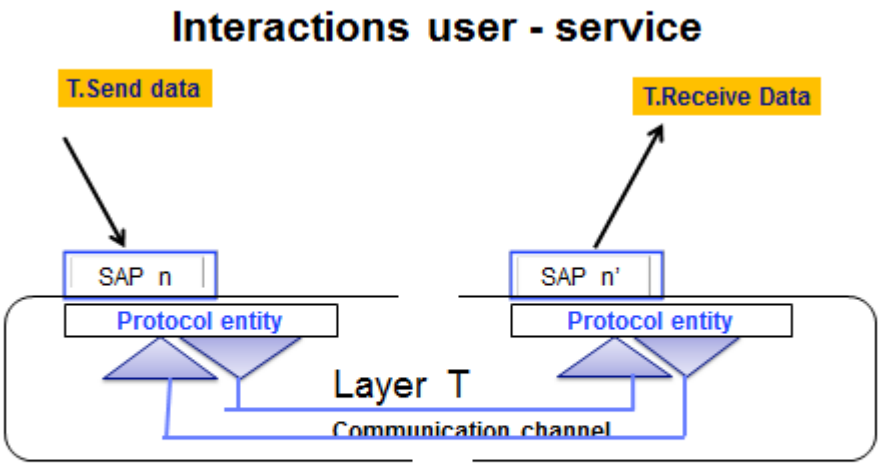
Between layer n+1 and layer n there is an interface that we shall call SAP for Service Access Point. We will describe the role of the Service Access Point in the next sequence.

At the sender’s end, you can see that in each protocol entity that is passed through, the initial data is supplemented, or encapsulated, with protocol control information. At the opposite end, the reverse happens and the protocol control information is extracted and used by the protocol. We say that the useful information is decapsulated so that it can be sent to the layer above.

These layers and envelopes are shown clearly by the Wireshark network analyzer that you will install and use in Section four.

In this lesson we have looked at the notion of the protocol and its consequences. In the next lesson we shall see how the layers interact.

Section 2, Lesson 4

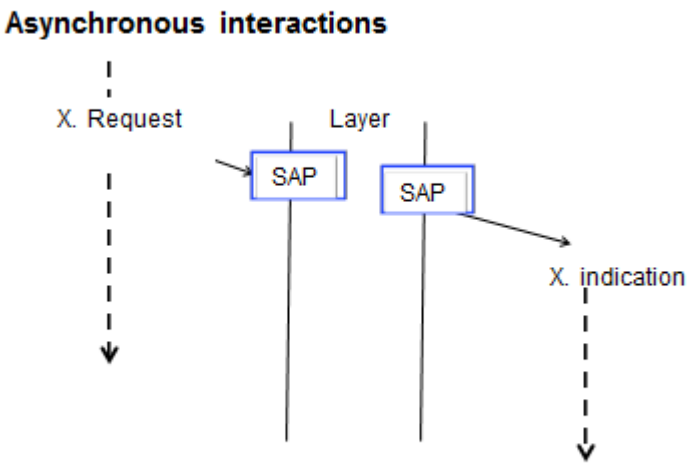


In the previous lesson we saw the notion of the protocol and some of it consequences. In this lesson we shall consider the interactions between layers.

For the user to be able to interact with the service and carry out what it is they want to do in the network, they need primitives and access points. For a postal service, only two primitives are needed: “post a letter” and “collect a letter”.

A layer supplies a set of services to the layer above, which are invoked by primitives. These primitives are usually referred to by a name preceded by the first letter of the layer’s name. So, T.CONNECT is a primitive of the transport layer, N.DATA is a primitive of the network layer.

These primitives are implemented by programming languages such as C, Python, C++ and java. Here we shall only describe a model to avoid reference to the different programming languages.



Four types of primitive can be useful, according to the nature and direction of the interaction:

Request, Indication, Response, Confirmation.

A Request enables the user to take the initiative of sending information.

A Request primitive enables layer i+1, which is using the services of layer i, to give a command. The data part, the parameter message for this primitive, will become the i SDU. The content is totally transparent to the service provider.

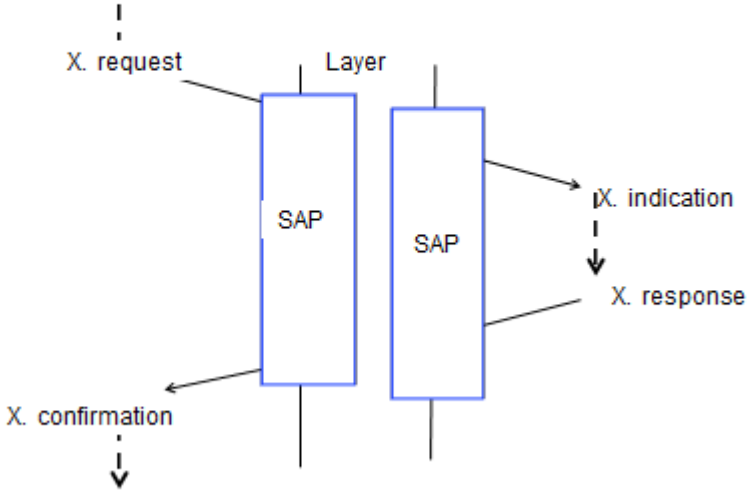
To execute its protocol, the service provider exchanges protocol messages, called PDUs, with its peers, that are necessary for the service to be executed correctly.

The receiver does not know when a piece of data will arrive. It must therefore be informed by an Indication primitive. The idea is that an indication is an “event” that wakens the receiver, like a phone ringing for an incoming call.

In the languages, it usually takes the form of an instruction placing the program on standby.

Synchronous intercations

Synchronous interactions



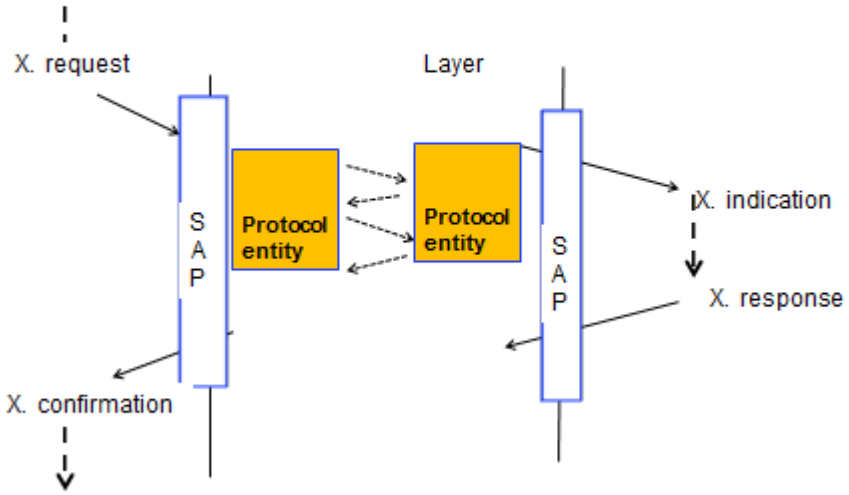
Two other types of primitive are necessary:

- The Response primitive enables the receiver to send information back to the sender. The Confirmation primitive means that the sender knows that its request has been executed successfully.

In asynchronous interaction, the Request primitive is not a block, meaning that once it has been executed the program that invoked it continues what it was doing. It cannot therefore know the result of its operation. In the case of an indication, there will be an interruption.

In synchronous interaction, primitives, like a call for a subprogram, block the calling program. A synchronous Indication primitive is carried out by a reception request that will make the calling program wait until data arrives.

So, communication is carried out between two SAPs. An SAP is therefore addressable.

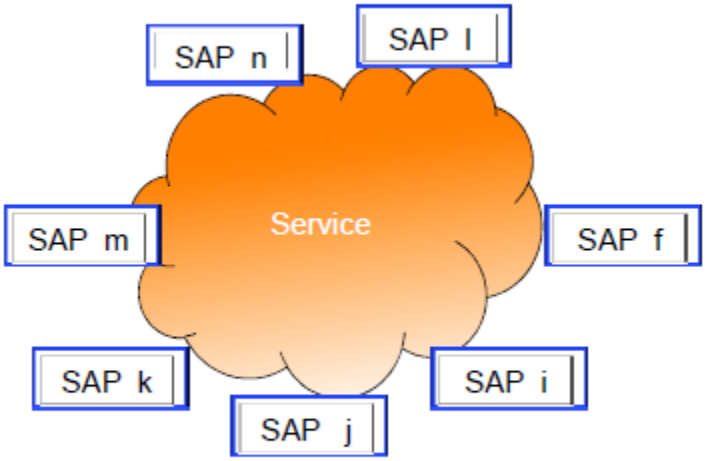


To carry out such operations, protocol entities exchange a certain number of PDUs that cannot be seen by the users.

For example, the data may be divided into small blocks, or delivery reports may be used to check that the data has arrived.

During the first Section we saw that there are two main types of services: datagram services and connection services.

Connection services require specific primitives to establish the connection. Data can only be exchanged once the connection has been made. Requests to send data have a parameter that is an identifier of the connection. This identifier is similar to creating an SAP at either end to which requests for sending data and the delivery indications are applied.



We have seen that a service can be used by a large number of users, and that interaction with the service is carried out at SAP level. An SAP therefore possesses a number, an address that is local to the protocol entity.

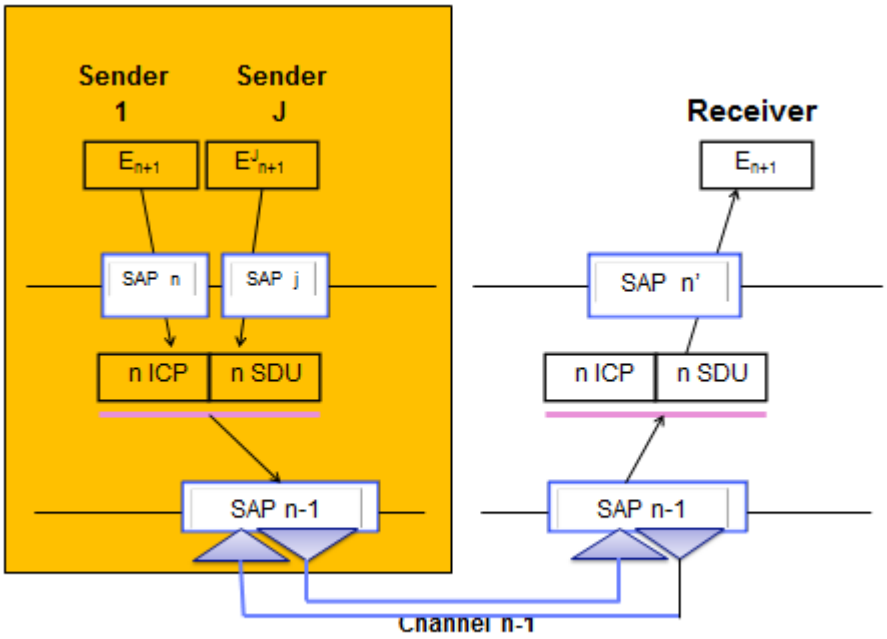
You can therefore consider the service provided by layer N as a “network” service with attached access points.

Each protocol entity can serve multiple applications through local SAPs.

N.B.: in the case of our protocol stack, a physical machine with a single CPU can only carry out one task at a time. This will be the assumption in the exercises.

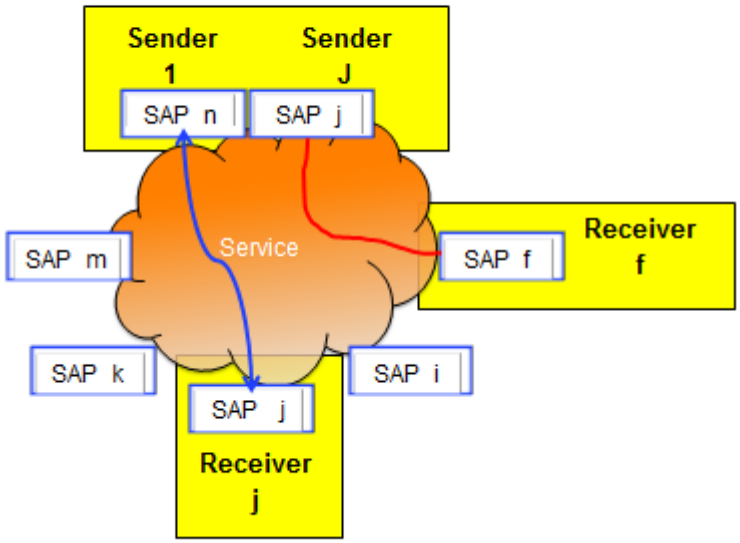
The physical layer, the MODEM, is usually provided by a dedicated coprocessor, and can therefore work in parallel with the CPU.

A physical machine equipped with more than one processor is known as multi-core, and is able to execute the code of one protocol entity per processor.



A single protocol entity is implemented in a physical machine, represented by the yellow square in the figure. It contains J applications. Each of these applications has at least one SAP with protocol entity n.

These applications can have the same or different correspondents, like in the diagram.



“Multiplexing” is having the possibility of sharing use of the same channel by several “sender / receiver” pairs.

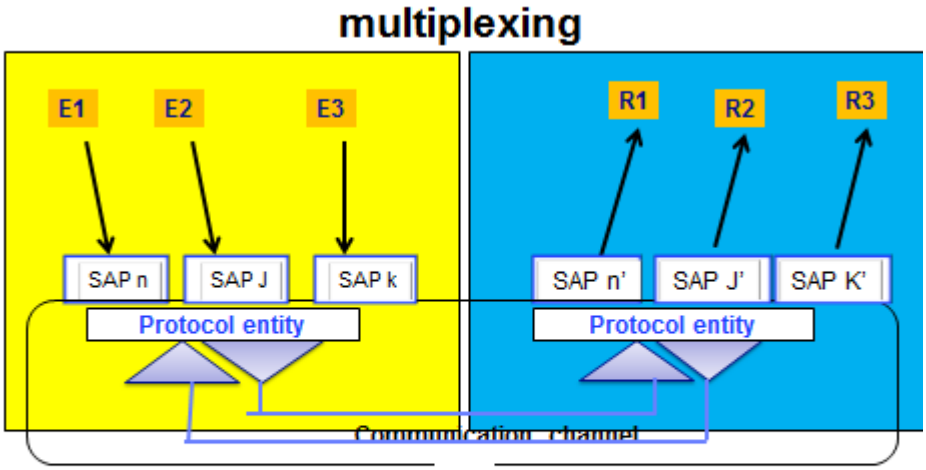
In the diagram, the communication channel between the yellow and blue machines is multiplexed: it is shared between three pairs of sender / receiver users. Multiplexing is very common in the physical layer to make the maximum use of capacity, but it is a general concept and can be applied to all the levels.

You will see several notions of multiplexing in networking:

- Time-division multiplexing: each user is able to use the channel for a certain share of time. It is most frequently used in telephony.
- Space-division multiplexing: each user has a share of the bandwidth. In GSM mobile telephony, users have a share of the frequency spectrum.
- Code-division multiplexing: each user is attributed a code for the transmission of data.

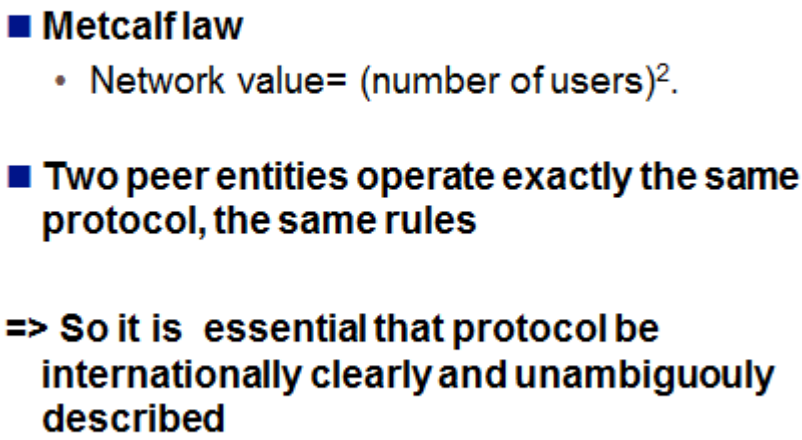
Dividing data into limited-sized packets is also a form of multiplexing since users are able to send PDUs in turn.

Sharing a communication channel implies that there are mechanisms to allocate the shared resource as well as time, frequencies and transmission capacity etc.



In this lesson we have dealt with service primitives used on service access points (SAPs).
In the next lesson we shall discuss standardization authorities.

Section 2, Lesson 5



During the previous lesson we discussed the interactions between layers. In this lesson we shall look at standards bodies.

I have mentioned the importance of standards on several occasions since the start of this course.

There are a large number of players involved, such as makers of computers and telecoms and computing equipment, as well as software companies and others.

It is essential that two peer entities apply exactly the same protocol in order for them to function correctly, regardless of the make and format (programming language, equipment etc.).

Networked systems must be open to all types of stakeholders, hence the name Open Systems Interconnection (OSI) standard.

Metcalf's law, named after the inventor of the Ethernet and founder of the company 3Com, says that a network's value is proportional to the number of users squared. In other words, the more potential correspondents you have, the greater the service's usage value. It is therefore rare for system owners to use non-public protocols, this being reserved for a few very specific domains.

- *ISO/OSI* is the author of the reference model ISO 7498 . It publishes many norms
- *Internet Engineering Task Force (IETF)* is the main actor for Internet protocols.
- IEEE inside commitee 802.x has normalized Local Area networks sur as Ethernet, Wifi..
- ITU International Telecommunication Union is very active in mobile commication

Unfortunately for you, there are many authorities who play a major role in standard setting, whether legal or more technical.

We shall not list all the authorities here, but only those we shall be referring to in this course:

- ISO/OSI is the author of the reference model and publishes a large number of standards.
- IETF, Internet Engineering Task Force, is the principal actor for Internet protocols. It publishes RFC, Request for comment.
- The IEEE's committees defined 802.x, the basis for standards for local Ethernet networks, Wi-Fi etc.
- The ITU International Teecommunication Union, based in Geneva, is a UN authority that is very active in telephony networks, especially mobile networks.

ETSI, the European Telecommunication Standard Institute, based in Sophia Antipolis, is officially in charge of standard-setting for Information and Communication Technologies (ICT) for Europe, in collaboration with the CEN and the CENELEC.

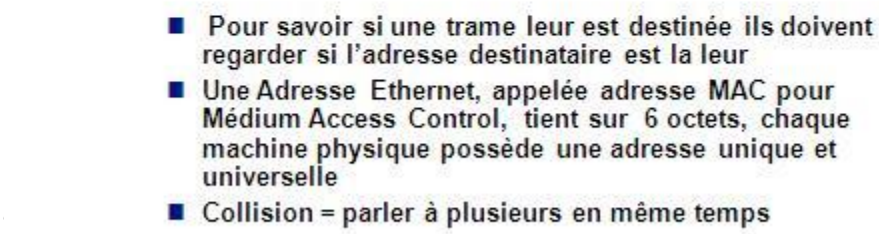
- ANSI, the American National Standard Institute, also plays an important role.

- Forums of various industrial stakeholders, builders and operators are frequently created and contributed to in order to develop and standardize technology, such as 3GPP for 4G mobile telephony.
- Makers of cables and electrical connections etc. have also grouped together to form associations.

- **ETSI European Telecommunications Standards Institute**
  http://fr.wikipedia.org/wiki/European_Telecommunications_Standards_Institute
  Play a key role in Europe as the institut mission is the normalisation of Information & Communication Technologies (ICT) for Europe

We recommend consulting different norm and standards documents during this course. The Wikipedia articles on these topics generally provide a suitable introduction for the requirements of this course.

In this lesson we have discussed contributors to standard-setting. In the final lesson, we shall study Ethernet protocol and its PDU, the Ethernet frame.



During the previous lesson we looked at standards. I now invite you to take a look at the Ethernet to close the Section.

The Ethernet standard is IEEE 802.3: Media layer CSMA/CD Ethernet.

Ethernet is a protocol on the data link level, known as Logical Link Control because it enables the sharing of communication media.

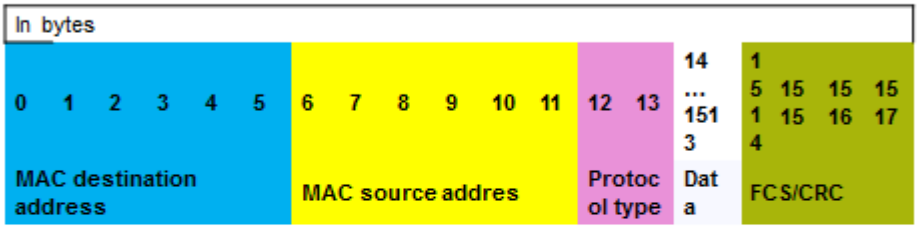
It uses a physical "bus" architecture, which means that all devices connected to the bus receive messages. Likewise, the devices can all emit information simultaneously.

Ethernet protocol is designed to ensure that only one device is emitting at any one time. The mechanism implemented by this protocol is based on rules of polite conversation. The devices all listen to the cable and none of them emit while another is already emitting. If no-one is talking or emitting, the device can emit. If it is the only one to do so, everything is fine, but if two or more devices speak at the same time each one stops and waits for a random amount of time before speaking again.

In particular I would like to illustrate SAP mechanisms. On the Ethernet, a SAP is called a MAC (Medium Access Control) Address. It is formed of 6 bytes. Every physical machine has a unique and universal MAC address.

To see whether a frame is intended for it, each machine must look to see whether its recipient address is in the frame.

PDU format / frame Ethernet



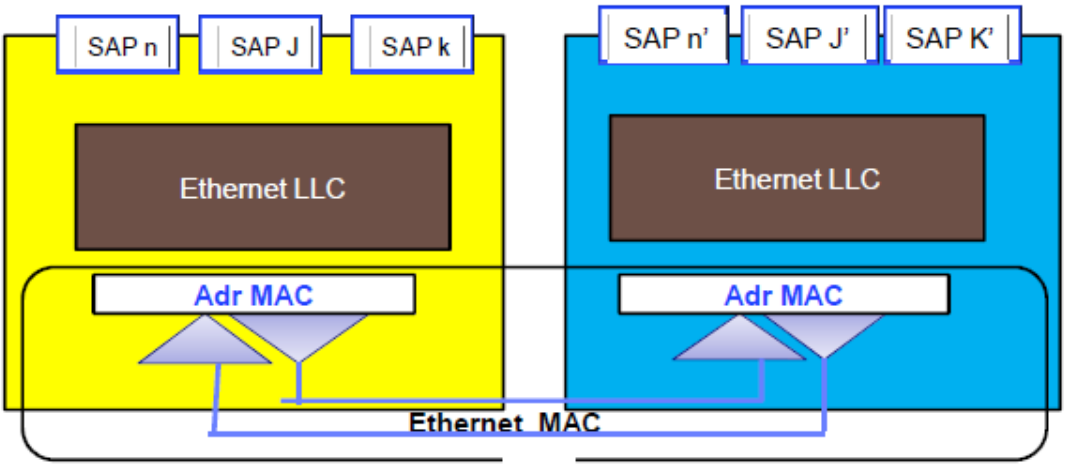
MAC = Medium Access Control
Is the layer name

Ethernet frames have a maximum length of 1518 bytes and a minimum of 64 bytes.

The Ethernet frame format is composed of 6 bytes for the recipient address, 6 bytes for the source address, which is the address of the machine that emitted the frame, the PDU. This address is useful for the recipient in order to know which machine to respond to if necessary.

Next, there are 2 bytes named “EtherType”. This field simply contains the number of the SAP to which the Ethernet must deliver the SDU it is carrying.

The SDU section can vary in length, from 46 bytes to 1500 bytes. While there is no logical reason for establishing a maximum length of 1,500, the minimum was required by the first Ethernet network’s physical properties for detecting the presence of multiple simultaneous senders. If the SDU does not contain 46 bytes, the LLC layer will add in nonfunctional bytes, a process known as stuffing. Finally, the last 4 bytes are a CRC, or Cyclic Redundancy Check, which is destined to detect transmission errors. If an error is detected upon reception, the frame is discarded.



To summarize, the destination address is the lower-level SAP. The "EtherType" field refers to the upper level of the SAP.

In the practical work, you will look at frames circulating in an Ethernet network and identify the destination protocols of the SDU. To do this you will first have to consult several IEEE and RFC standards.

Next Section's lessons will be led by Olivier Paul, who will describe various protocols.

_____

To find out more

http://en.wikipedia.org/wiki/Ethernet

and

http://en.wikipedia.org/wiki/IEEE_802.3#IEEE_802.3_et_standards

RFC 1166 Internet Numbers