Standard interfaces in networks

If heterogeneous networks want to communicate, they must use a common interface—in this case a common packet structure and a mutually agreed set of communication rules.

The International Organization for standards (ISO) came up with a standard for a common network interface and called it OSI. While nobody adopted that standard *verbatim*, all the currently used interfaces follow the approach introduced in OSI.

Layers and protocols

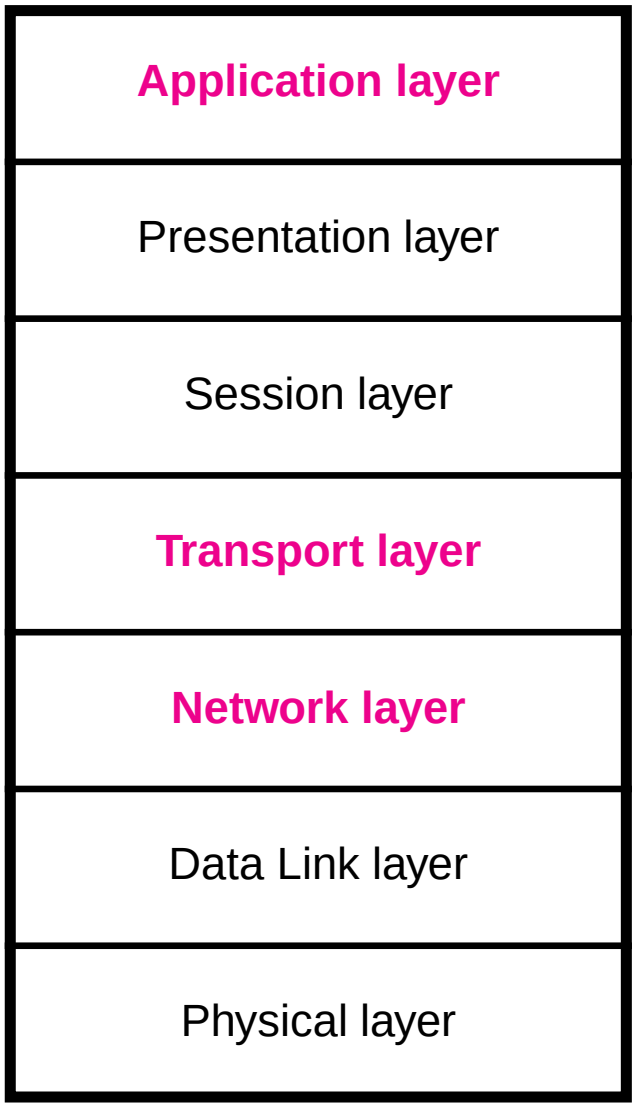
The foundation of OSI is the concept of dividing the network software into **layers**.

A user message generated by an application travels **down** the layers to the very bottom (the sender's telecommunication hardware).

The hardware transmits the message into a link (thinking it is sending it to the receiver).

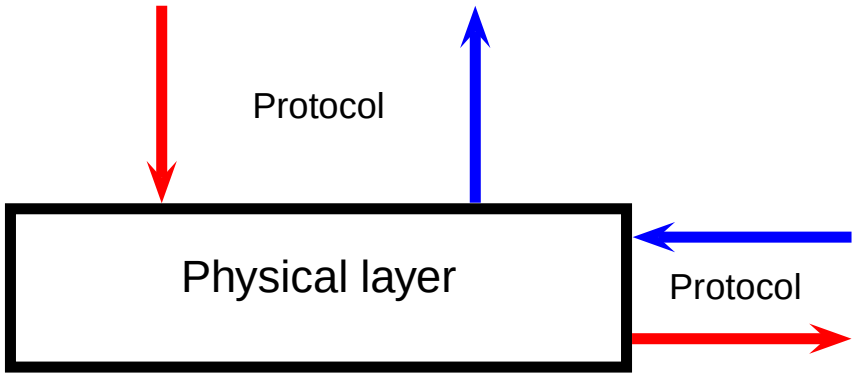
When the message reaches the receiver's hardware, it starts moving **up** until it is handed to the receiving application.

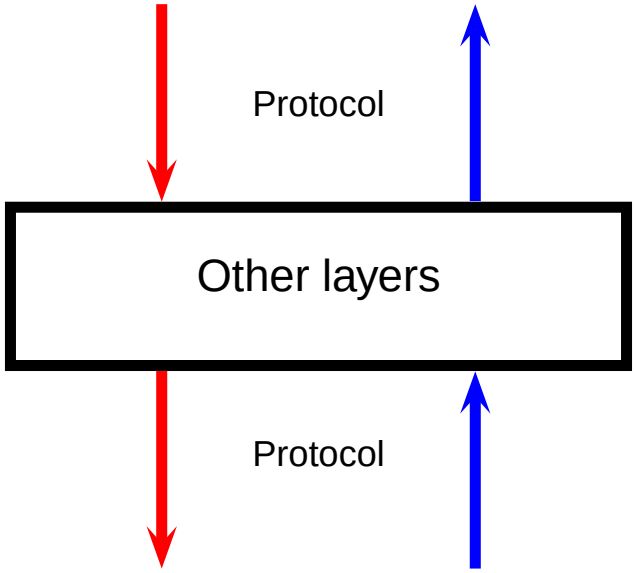
This view is labelled the **OSI stack** or **OSI model**.

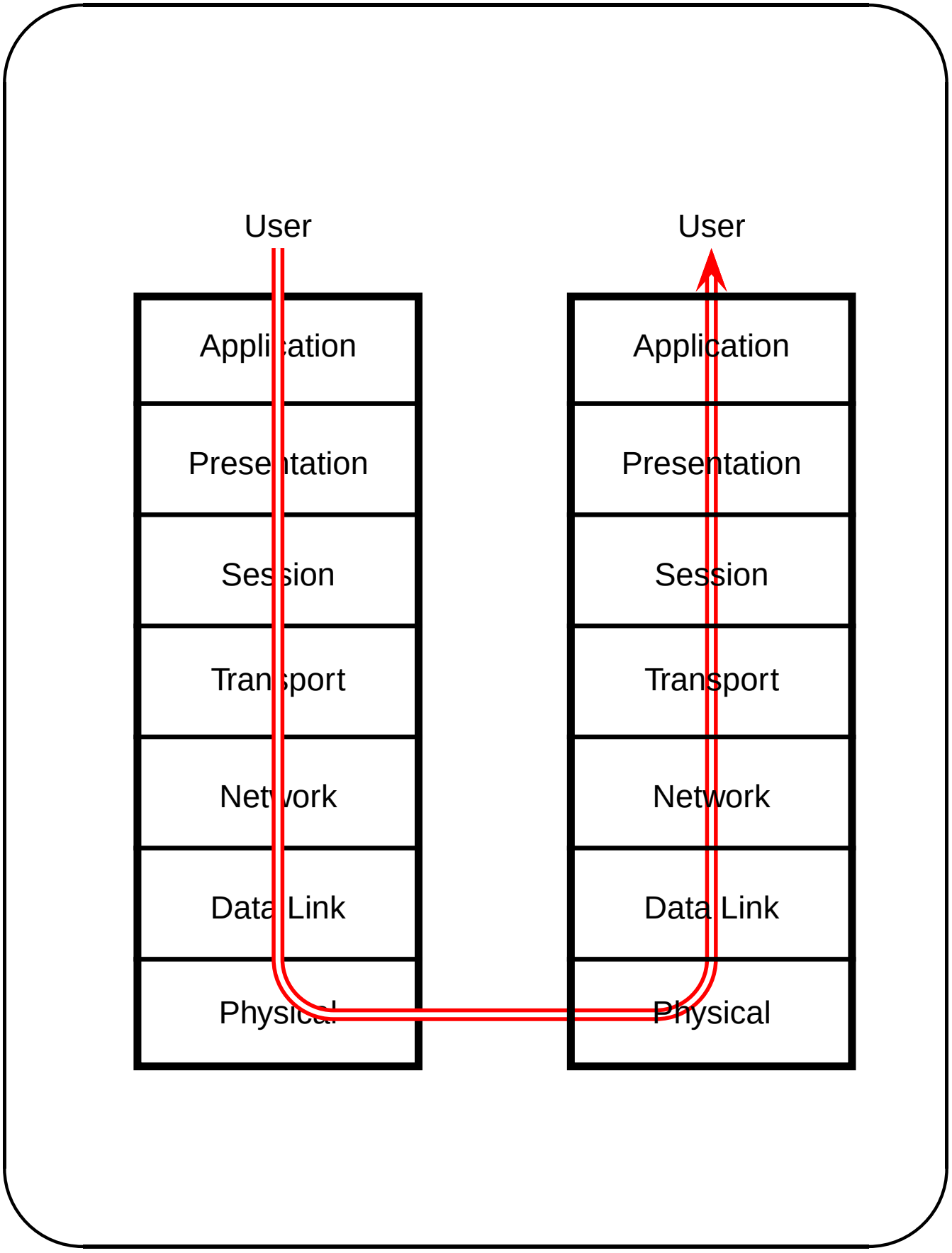


See howstuffworks if you still are curious.

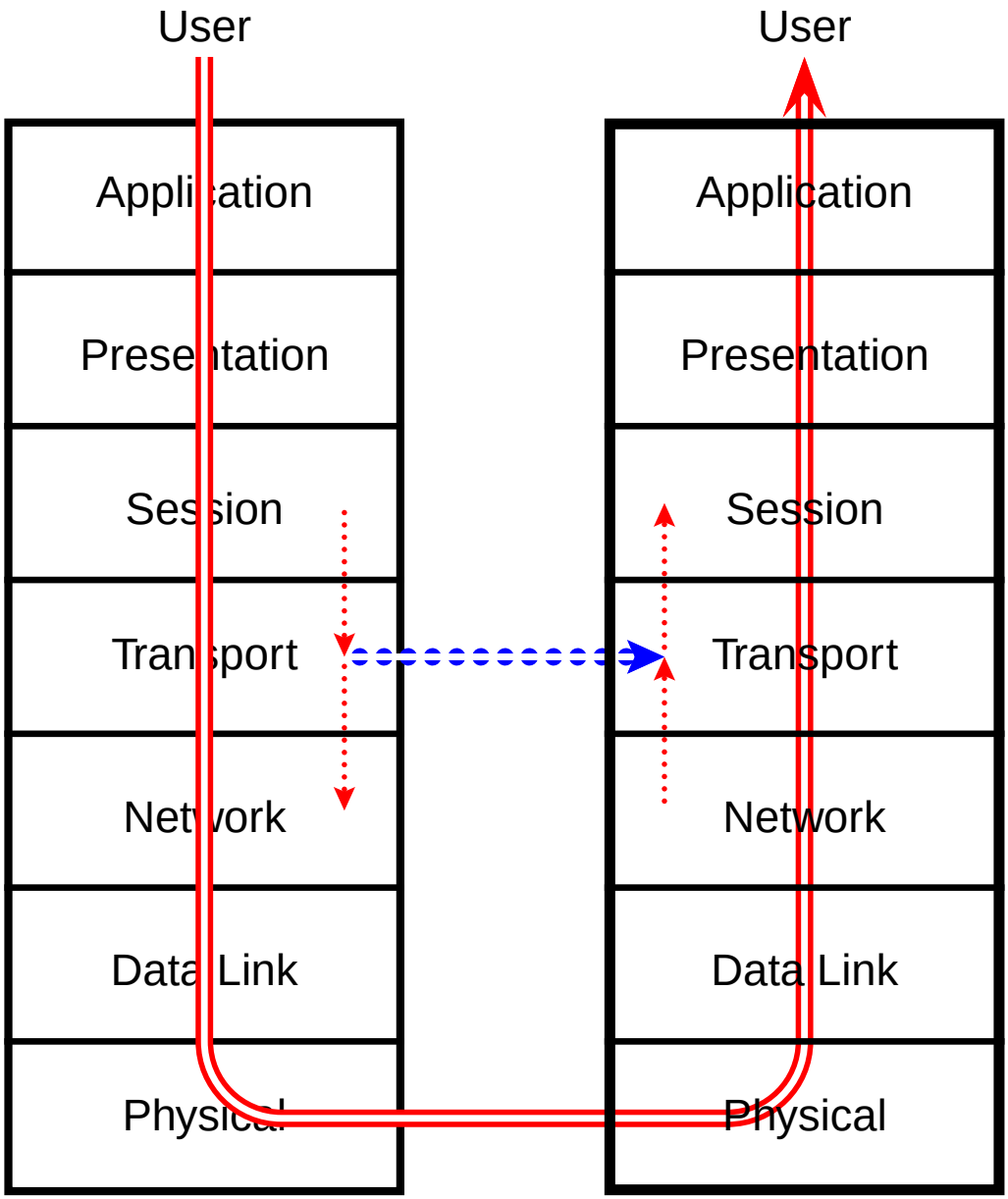
At each layer, the movement of data units is controlled by standard protocols describing the form of legal packets as well as some required behaviour of the software implementing the layer. Several protocols may coexist within one layer.






User
User
Application
Presentation
Session
Transport
Network
Data Link
Physical
Application
Presentation
Session
Transport
Network
Data Link
Physical

Each layer at the sender has the virtual perception of communicating directly with its counterpart at the receiver.



Internet layers of protocols

The most commonly used protocol set is the DoD suite, also known (incorrectly) as **TCP/IP**, which ignores the presentation and session layers. It is the basis of the layers of the Internet **protocol stack**.

Application

Transport

Network

Data link

Physical

The "Session" layer exists inside the TL.

Application layer

This is the layer we all use when we access networks. It is responsible for supporting access to the network by application programs. Many standard protocols provide this support (HTTP, SMTP, FTP).

Anyone can create an application by writing a program acting as a server or as a client. Typically, such programs use specialised libraries or the general–purpose **socket** interface provided by the session/transport layers.

Session layer

This layer sets up the **end–to–end circuit** that will last for the duration of the session (from sign–on to sign–off). Its main task is to allow a **handshake** between the two hosts to be performed.

This layer also handles checkpointing (if implemented) and recovery from lost connections (recreating a circuit).

Note that not all data traffic requires circuits (or sessions).

Transport layer

This layer is responsible for moving data from one end of the circuit to the other. It is the lowest end–to–end layer and is not concerned with the details of the network core.

Connection–oriented transport which uses a circuit to transfer data from host to host. Many standard protocols for connection–oriented service have been proposed, but TCP is king supreme. You can create your own (should be better than TCP).

Connectionless transport does not establish a circuit, nor a session ("connection"). Each packet is moved independently of all other packets; UDP is the connectionless protocol used in real applications.

Note that connectionless protocols cannot be reliable, since there is no feedback.

Network layer

This layer controls the flow of data inside the network core: **routing** (from *to route*, not *to rout*). The basic protocol is the connectionless **IP**, but there are other protocols in use (inside private networks, not on the Internet).

LANs do not need a network layer, since there is no need to route packets; hence, the presence of a network layer makes a network a WAN.

A network–layer can be:

- connection–oriented, in which case the routing decision is made when establishing the virtual circuit for the connection (by the session layer).
- connectionless, in which case a separate routing decision is made for every outgoing packet. The Internet is connectionless; hence, IP is.

Performance issues

The Transport and Network layers are the subject of most research, because they have the most obvious impact on performance of networks.

Some aspects:

Max effective throughput is the maximum number of user bits that can reach the receiver divided by the bandwidth ("nominal throughput") of the network. Some assumptions about data traffic are needed to calculate an exact value.

Max delay is often calculated in order to determine whether a network satisfies some real–time quality requirements (for instance in live audio/video transmissions).

Loss rate is a relevant measure for lossy network layers (such as IP).

Throughput vs. Delay shows the network's ability to survive congestion periods.

Data loss may occur in connectionless networks and in over–subscribed connection–oriented networks (that use statistical multiplexing).

Scalability implies that the networks behaviour will not change if the distances between nodes or the transmission rate are increased.

(Data) Link Layer

This layer transmits packets one hop forward, along one physical link. Examples: *Ethernet* (**CSMA/CD**), *PPP*. Note that a single link may include bridges (e.g. multiple–segment Ethernet).

The main responsibility of the DLL protocol is to ensure an **error–free** transmission, so that higher–layer protocols can assume they are handling packets of meaningful data. This responsibility is unrelated to **reliable transport** layer, which takes care of lost packets, not corrupted bits.

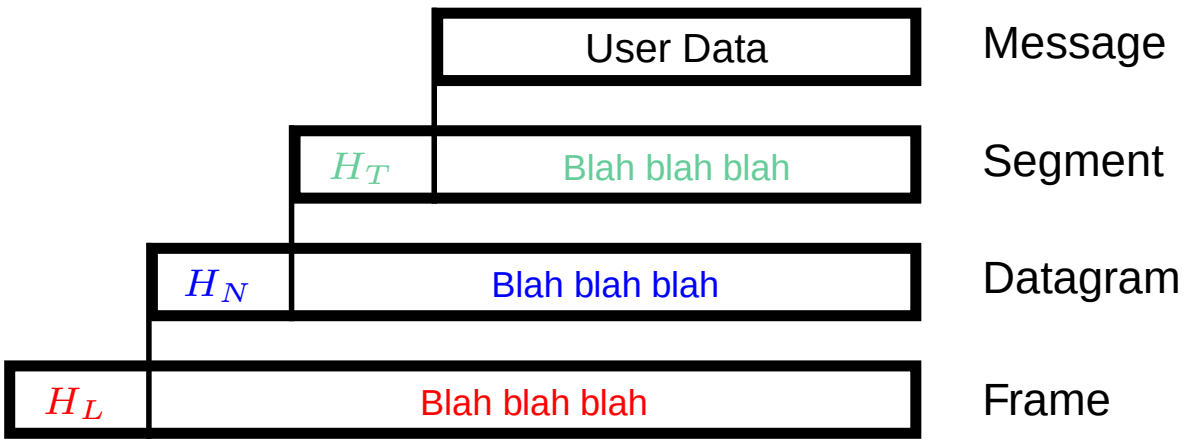
Physical Layer

This layer transmits individual bits on a single physical link, which may include repeaters. What constitutes a single link is product dependent. There is some error control at this level achieved by using redundant codes.

Protocol Data Unit

Abbreviated as **PDU** this is the unit of data from the point of view of a given protocol and is used instead of “packet” in pro circles. There is a synonym for the PDU used by each layer:

7	Application	A–PDU	Message (“Text”)
4	Transport	T–PDU	Segment
3	Network	N–PDU	Datagram
2	Link	DL–PDU	Frame
1	Physical	1–PDU	Bit/Symbol



A simplified view

A typical network may be seen as a stack of 3 layers:

- Application layer—this is where “we” write application programs that access the network.
- Transport layer—reliable movement of data is provided by this layer. “We” have no direct access to this layer and can only request its service through library **system calls** from within our program.
- Physical layer—bits are moved there. “We” have no way of talking to the software driving the physical layer (there is some software there, in the form of device drivers).

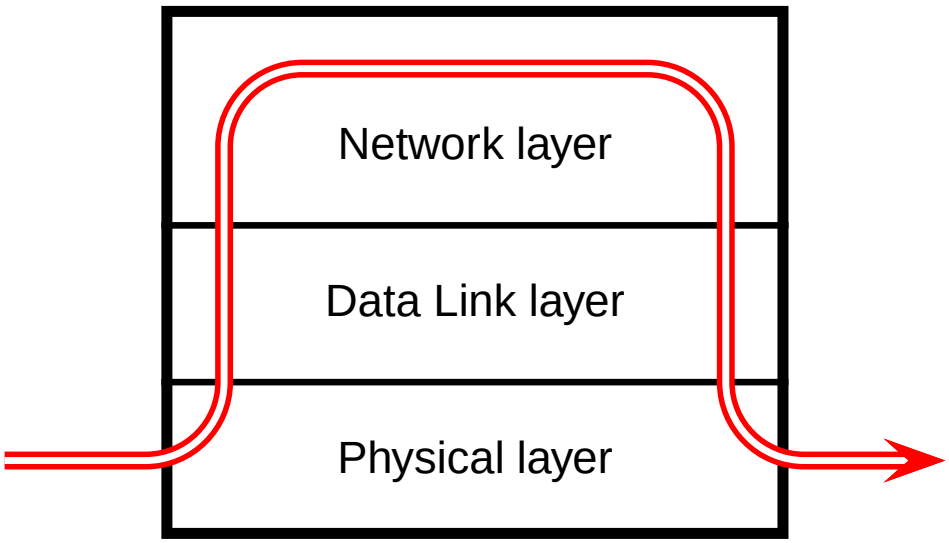
Traditional *Ethernet* (CSMA/CD) with TCP looks exactly this way.

More on protocol stacks

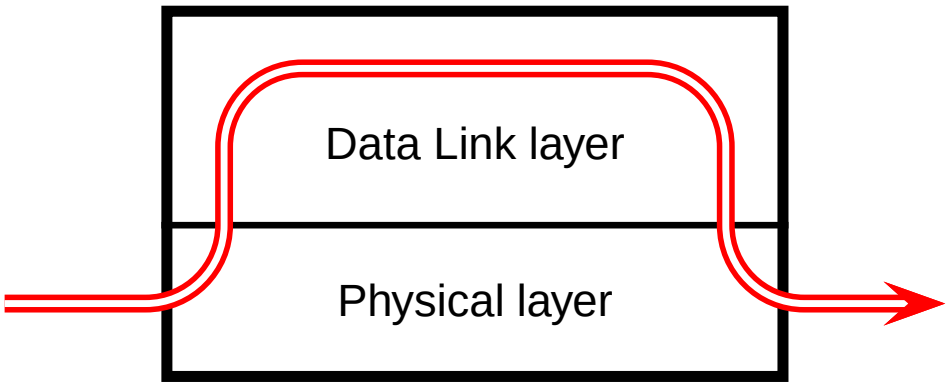
The OSI stack and the Internet stack represent the layers of software in the host stations (source and destination). But a message travels through the network core before reaching the destination host. The switches (etc.) inside the core do not need a whole stack to perform their functions (routing and relaying).

Their stacks are truncated; moreover, they play simultaneously the role of sender and receiver.

Network router stack



Link–layer switch stack



There is no need to impose standards inside a switch or router.