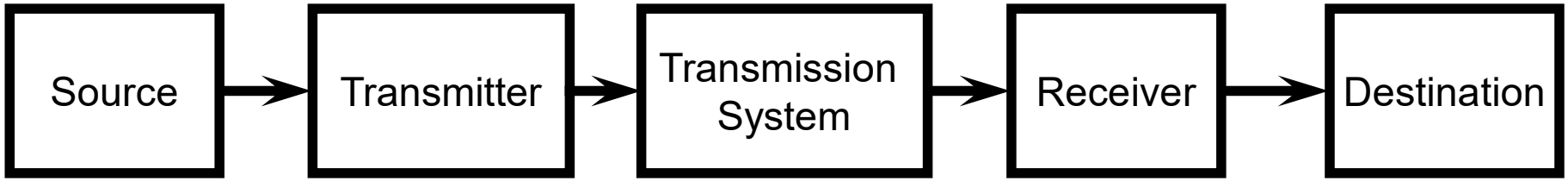
Protocol Hierarchies

- Computer networks are generally comprised of numerous pieces of hardware and software.
- Recall the previous attempt to ‘model’ how these components interconnect and interact with each other.



The Five-component Communications Model.

Protocol Hierarchies

- Whilst this horizontal model is useful when looking at hardware components, the reality is it does not explain everything.
- From previous discussions on: ***Data Link Control***, ***LANs technologies***, ***Internetworking*** etc. much of the functionality required to provide communications between host machines is not necessarily provided on individual hardware components:
  - For instance the transmission/reception of MAC frames requires sending and receiving binary ones and zeroes on a variety of transmission media using a variety of access techniques.
  - This mixture of functionality is provided on ***NICs*** (one per LAN technology) and in software elsewhere on the host.

Protocol Hierarchies

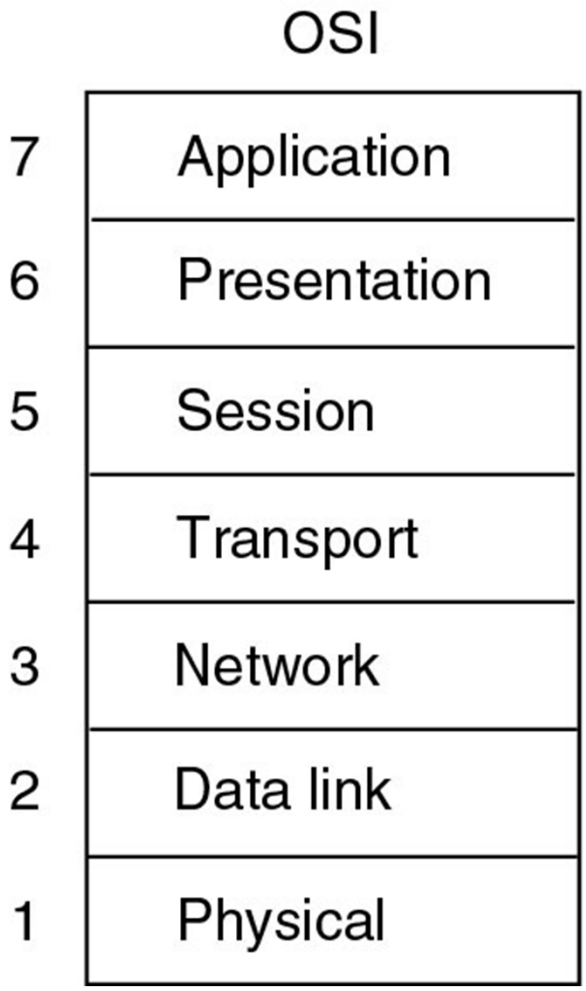
- A better model is required to explain these concepts and techniques.
- To simplify network design most networking technologies are organized as ***layers*** of protocol software:
  - The purpose of each layer is to provide one component of the overall solution to the “Communications Problem” such as “frame reception/transmission” etc.
  - These layers of software can exist on separate hardware components and/or hosts or is some instances multiple layers can exist on a single hardware platform and/or hosts.

Protocol Hierarchies

- This concept is not dissimilar to the approach used for software development:
  - Objects or libraries are used to perform specific operations.
  - Importantly the object/library function keeps details of its internal state and algorithms hidden from the main program.
- The same approach is used for network protocol software:
  - Each layer of software provides one part of the solution such as: frame transmission/reception or, transmitting one/zeros onto a wire/wireless transmission medium etc.

Protocol Hierarchies

- With networking software, the layers of protocol software are organised into a ***Stack***
- Each layer is said to offer ***services*** to higher layers and to use the services offered by the lower layers.



Protocol Hierarchies - terminology

- Some terminology:
  - *Protocol/Protocol Entity.* A software module that performs one sub-task such as: *frame transmission/reception* etc.
  - *Protocol suites/stack*. A number of such software modules that work together to perform all of the tasks required to enable two hosts communicate.
  - *Protocol Architectures/Model*. A model by which we can categorize or view the different protocol layers.

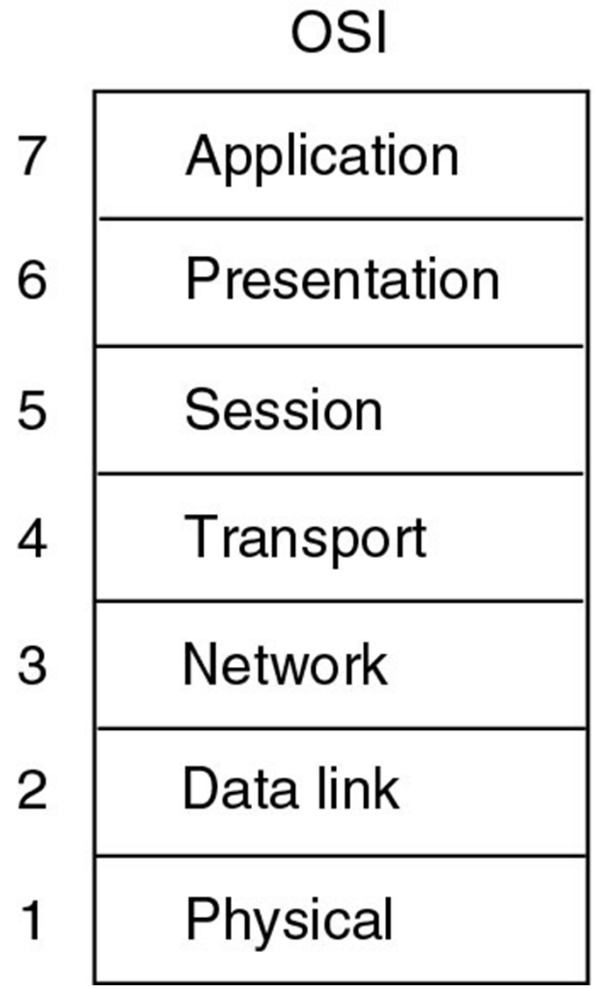
The OSI ISO Reference (7-Layer) Model

- The *OSI model* deals with connecting open systems:
  - i.e. systems that are open for communication with other systems regardless of who built them or how they were manufactured.
- The principles that were applied to arrive at the seven layers are as follows:
  - Each layer was created when a different ***level of abstraction*** was required.
  - Each layer performs one well-defined function with each function chosen carefully to facilitate the development of ***standardized protocols***.
  - The number of layers chosen was sufficient enough to ensure that distinct functions were not *lumped* together to become unwieldy.

The OSI ISO Reference (7-Layer) Model

- The layers of software work together in unison to provide overall solutions to the problems of communicating between host computers across complex networks.
- ***Information*** flows up-down the stack, across each of the interfaces (boundaries) between the layers:
  - These boundaries were carefully defined to *minimize* information flow across the interfaces.
  - The following slides discuss the functionality of each layer of protocol software. This functionality should be recognisable from previous discussions.

The *ISO OSI Reference Model*



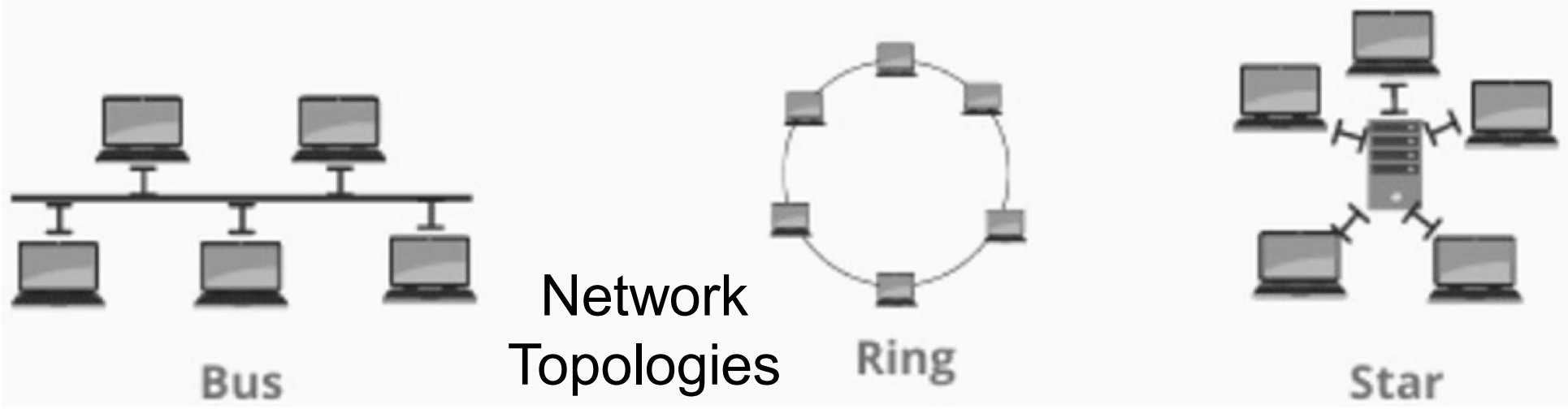
The **7-Layer *Model: Layer-by-layer***

- The *Physical* Layer (Layer 1): Concerned with transmitting raw bits over a *communication channel*:
  - Its purpose is to ensure that the transmission of binary 1s and 0s adheres to what is appropriate for the transmission medium and what is expected by the *receiver.*
  - Primarily it deals with matters such as: What voltage levels are used? What frequencies are used? How long does it take to transmit a bit (Bit Duration)? etc.
  - There are certain design issues to address such as: the *mechanical and electrical* design of the plugs (RJ-45, BNC etc.) and sockets, *timing interfaces*, the physical *transmission medium* etc.

The **7-Layer *Model: Layer-by-layer***

- The *Data Link* Layer (Layer 2): Concerned with the successful transmission of data across an individual link:
  - Its purpose is to transforms a raw transmission facility into a *data communications channel* that *appears* free of transmission errors.
  - Primarily it deals with matters such as the transmission/reception of *data frames.*
  - There are certain design issues to address such as: the creation of localised unique addressing, the creation of a unique framing structure, flow control, error control, controlling access to a shared channel etc.

Example DL functionality



Network Topologies

Ethernet (802.3) Frame Format

7 bytes	1 byte	6 bytes	6 bytes	2 bytes	42 to 1500 bytes	4 bytes	12 bytes
Preamble	Start of Frame Delimiter	**Destination MAC Address**	**Source MAC Address**	Type	Data (payload)	CRC	Inter-frame gap

For TCP/IP communications, the payload for a frame is a packet

WiFi (802.11) Frame Format

2 bytes	2 bytes	6 bytes	6 bytes	6 bytes	2 bytes	6 bytes	0 to 2312 bytes	4 bytes
Frame Control	Duration	MAC Address 1 (Destination)	MAC Address 2 (Source)	MAC Address 3 (Router)	Seq Control	MAC Address 4 (AP)	Data (payload)	CRC

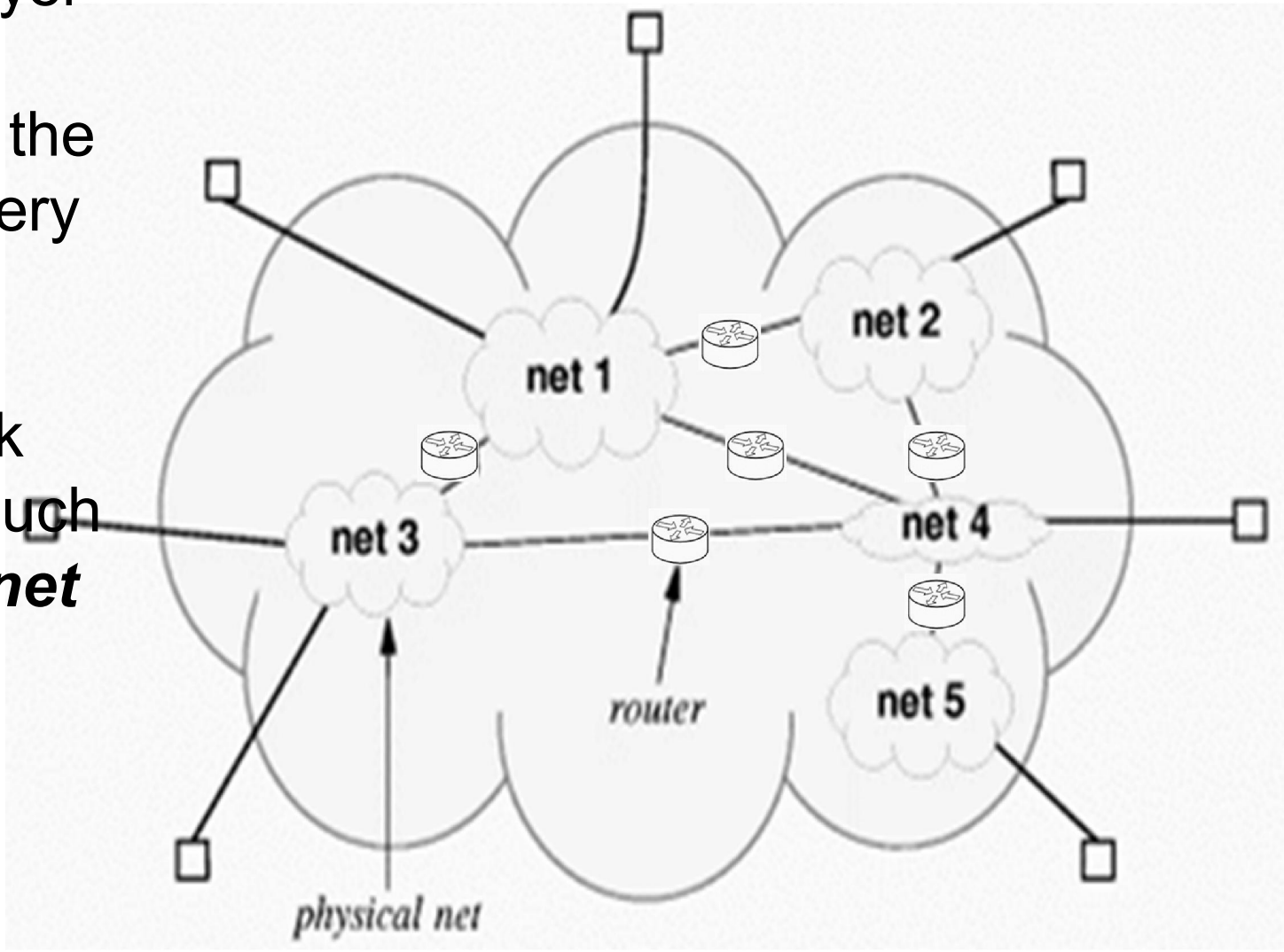
Frame Structures

The **7-Layer *Model: Layer-by-layer***

- The *Network* Layer (Layer 3): Concerned with the successful delivery of data between hosts across complex network infrastructures:
  - Its purpose is to control the operation of the *sub-networks* to achieve host-to-host delivery.
  - Primarily it deals with matters such as the routing of *packets* from the *source* station towards the *destination* station across sub-nets etc.
  - There are certain design issues to address such as: the creation a globally unique address space, the creation of a globally unique framing structure.
  - Essentially this layer is responsible for managing data communications across interconnected *heterogeneous* networks.

The **7-Layer *Model: Layer-by-layer***

- The *Network* Layer (Layer 3): Concerned with the successful delivery of data between hosts across complex network infrastructures such as a large ***internet***



The **7-Layer *Model: Layer-by-layer***

- The *Network* Layer (Layer 3): Its purpose is to control the operation of the *sub-networks* to achieve host-to-host delivery:
  - Primarily it deals with matters such as the routing of *packets* from the *source* station towards the *destination* station across sub-nets etc.
  - There are certain design issues to address such as: the creation a globally unique address space, the creation of a globally unique framing structure.
  - Essentially this layer is responsible for managing data communications across interconnected *heterogeneous* networks.

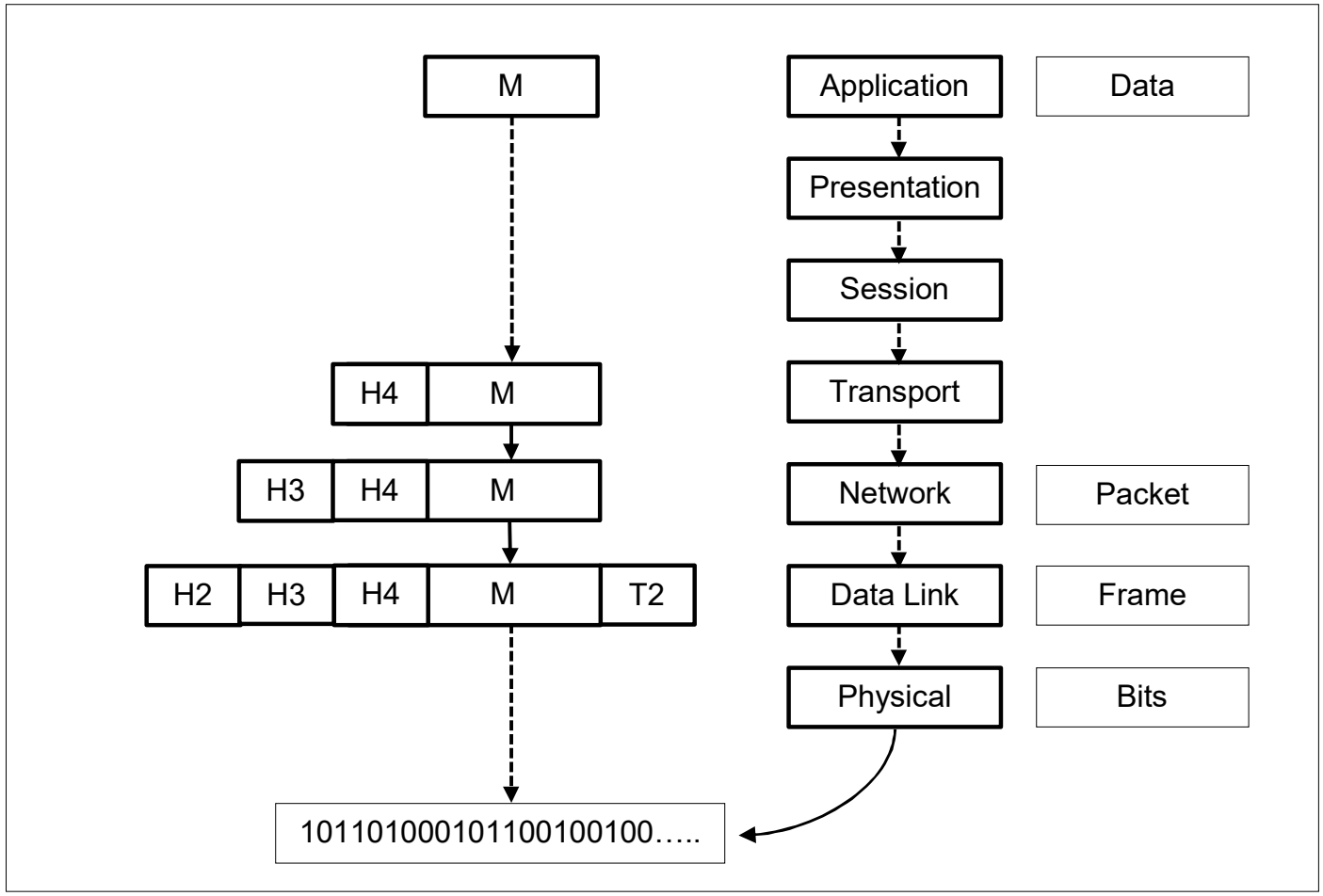
The **7-Layer *Model: Layer-by-layer***

- The *Transport* Layer (Layer 4): This is a key layer.
- It is a true *end-to-end* layer responsible for the reliable delivery of data between **processes** on end-host machines i.e. process-to-process delivery:
  - Its purpose is to provide a reliable data transport service to applications.
  - Primarily it deals with interfacing with applications for the purpose of exchanging data between applications across a network.
  - There are certain design issues to address such as: multiplexing data streams from/to remote applications, data loss, network latency etc.

Data Flows

- The layers work together to provide the complete functionality required to facilitate communications between two remotely connected host machines regardless of how they connect to the internet.
- The following slide shows how data flows between the layers (up-and-down):
  - Note that sometimes it is necessary to break up the data to meet some size restrictions associated with some lower layer protocol software. Recall use of *Fragmentation* at the Network layer.

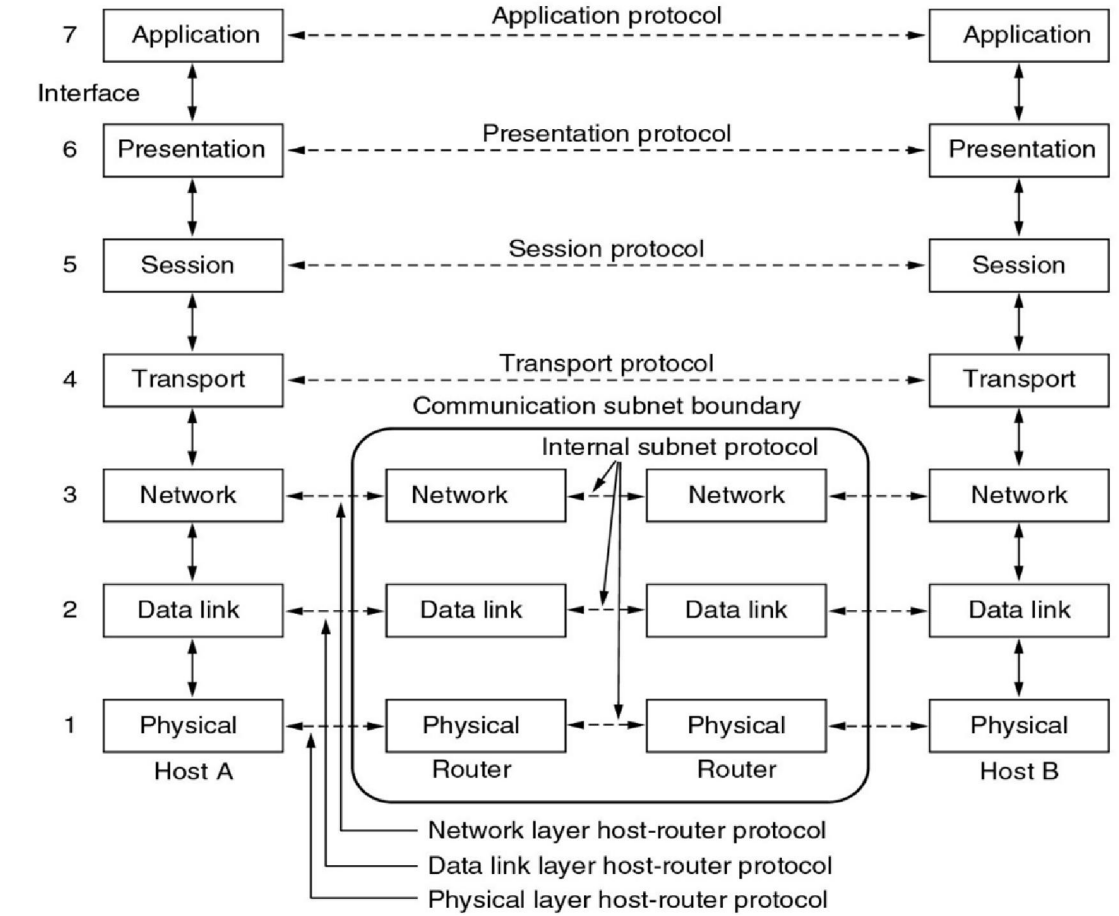
Encapsulation and Information flow between the layers on an end-host.



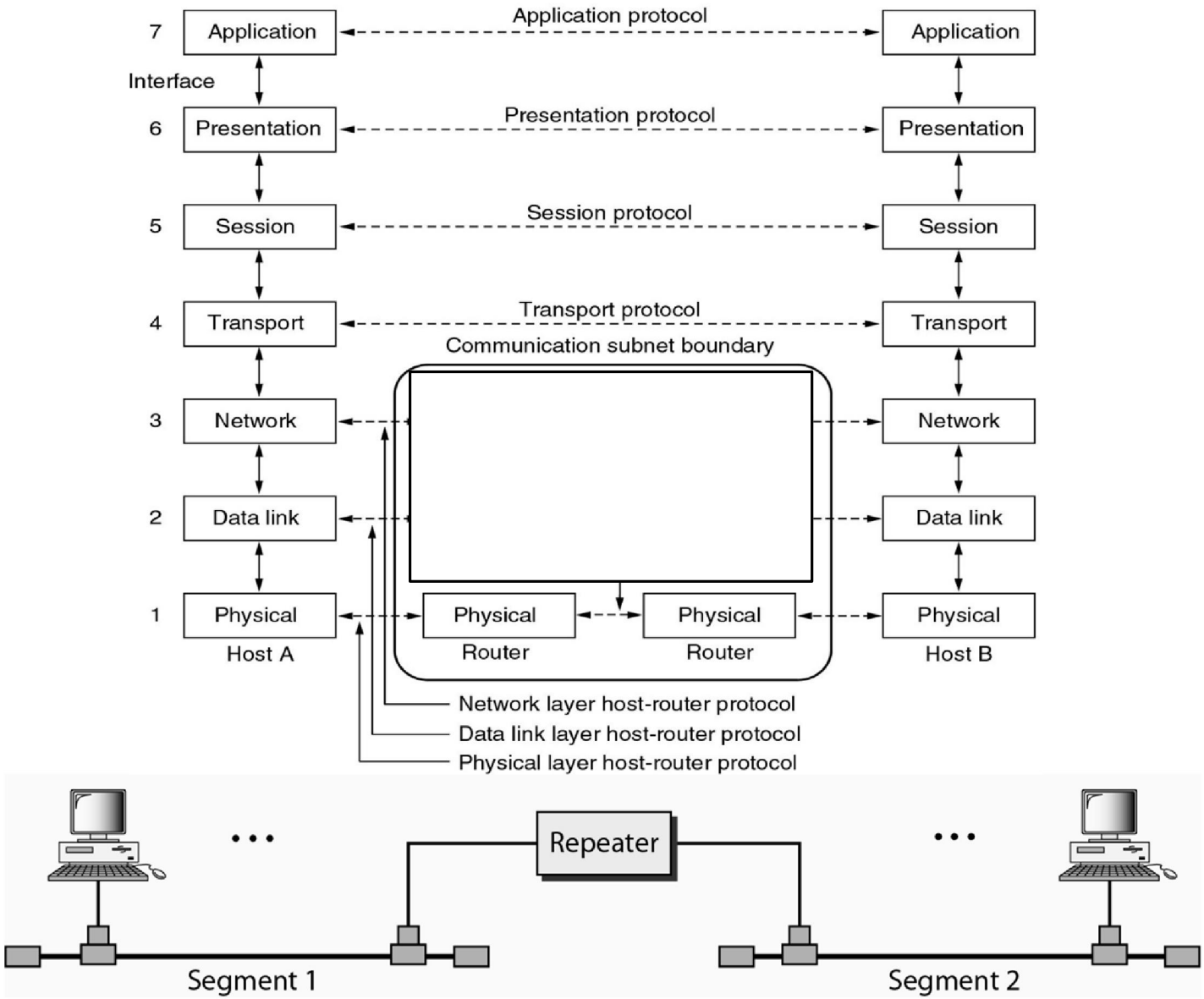
Data Flows

- Some of the layers are also implemented on intermediary networking devices such as Routers:
  - This is because the functionality associated with these layers is needed for a particular purpose such as: frame reception/transmission, packet routing etc.
- The following slide shows how the complete set of layers are implemented on the end-hosts as well as on the intermediary networking devices.

The *ISO Reference Model* – Layers 1-3 V Layers 4-7



Repeater Implementation



Repeater Implementation