-컴퓨터 네트워크-

Protocol Stack

2021 Spring
Kyungseop Shin

Lecture Outline

- Protocol 및 layer architecture에 대한 모델에 대해 이해
  - service/entity/protocol 모델의 기본 개념 및 용어
  - layered architecture 특성
  - OSI layer 개념

Internet에서의 Protocol 역할

- Internet은
  - 여러 SW/HW 복잡체로 이루어진 복잡한 시스템
  - 기능 관점에서는 protocol들의 집합
- Internet을 이해하려면
  - Internet을 구성하는 각 protocol들의 메커니즘을 이해하면 됨
  - 추상적인 protocol에 대해 개념 수준부터 이해 필요

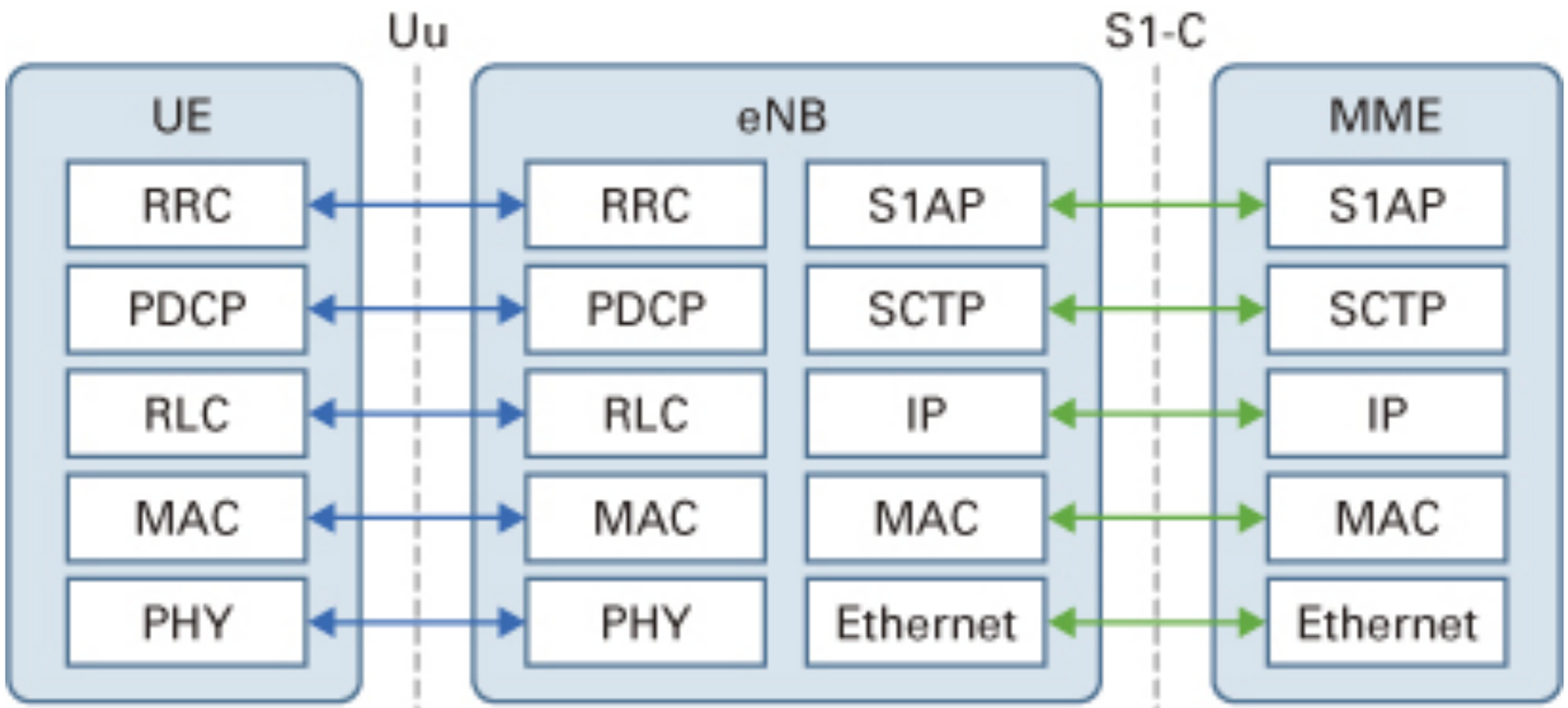
Protocol 이란

- Internet 안에서의 각 entity들이 동작하는 방식
  - 정보 송수신에 대한 기능적 실체
  - HW/SW 간 서로 메시지를 주고받으며 상호 작용을 하는 약속
- 주로 주고받는 메시지 + 관련 동작을 정의

Protocol defines the format and the order of messages exchanged between two or more communicating entities, as well as actions taken on the transmission and/or receipt of a message or other event

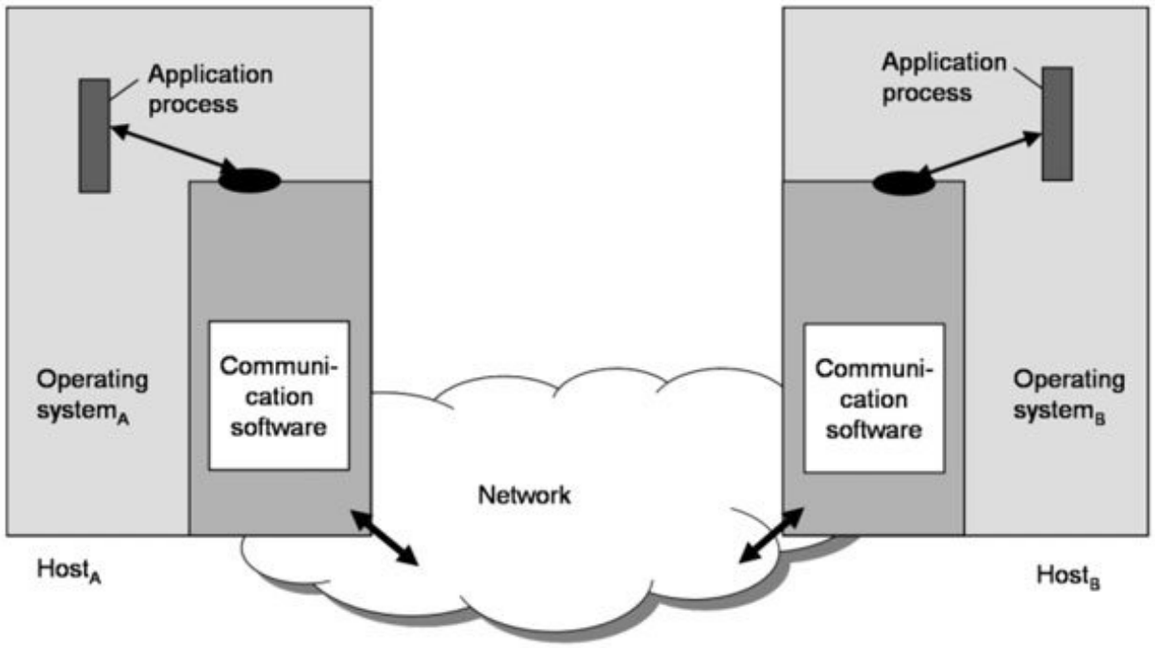
Protocol 구조

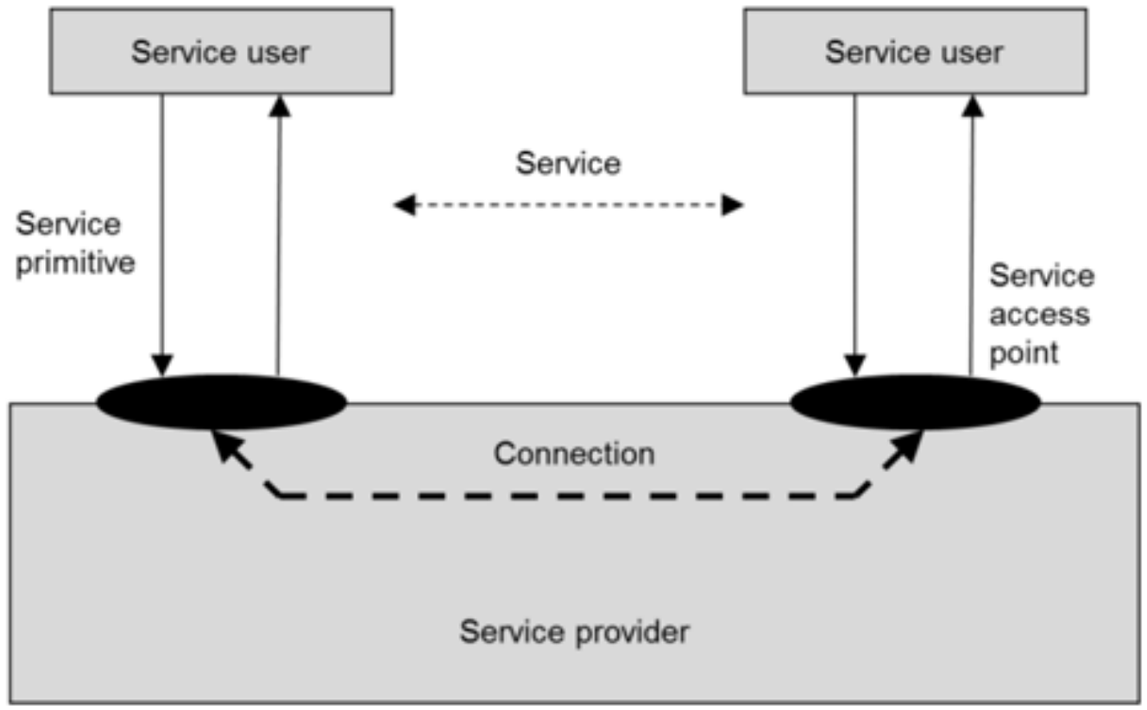
- 통신 네트워크는 여러개의 protocol이 서로 얽혀서 동작
  - stack 구조로 위 아래 protocol 끼리 API로 상호 작용
  - 상대방 protocol과 상호작용



Protocol Model : Service 개념

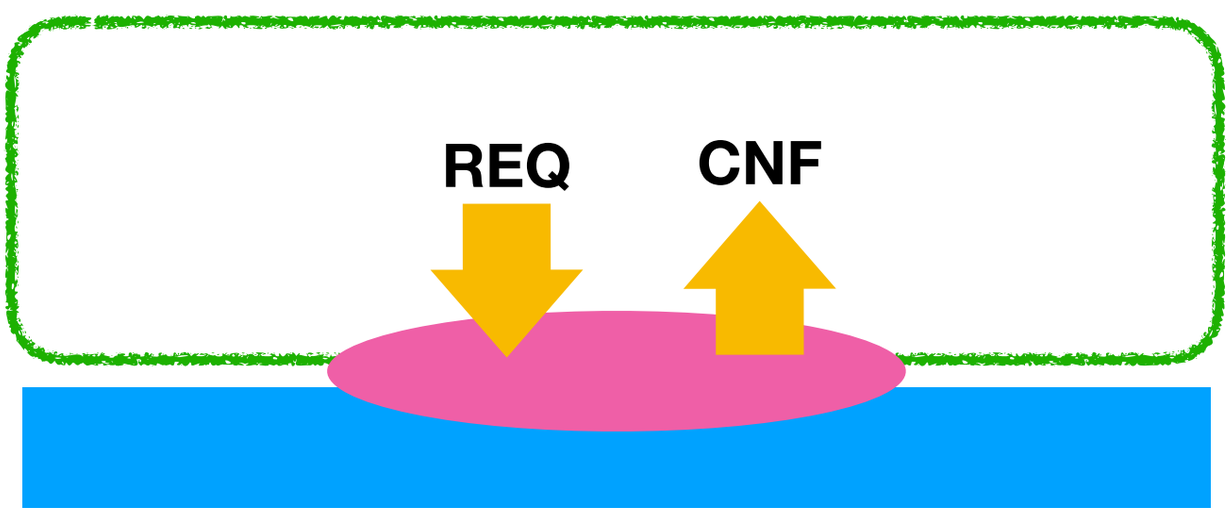
- Client (혹은 service user) 에게 무언가를 해주는 것
- Communication service model
  - SAP (service access point)를 통해 packet 전달 서비스 제공
  - service user는 SP (Service Primitive)를 통해 서비스 받음

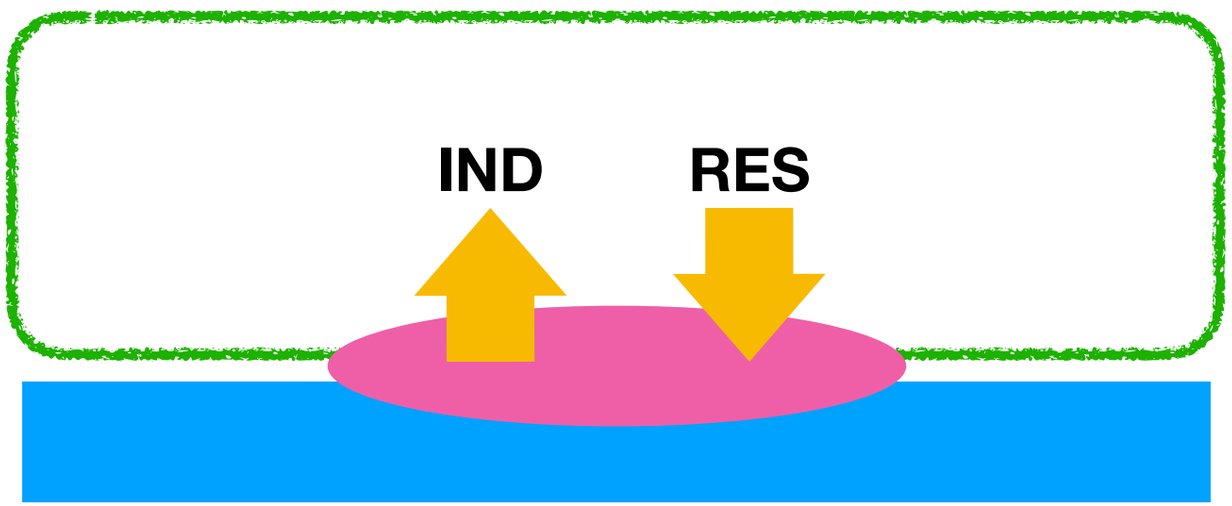




Protocol Model : Service Primitive

- Service primitive는 이름 / 형식 / 파라미터로 구성
  - 이름 : CONNECT, DATA, …
  - 형식 : request / indication / response / confirm
  - 파라미터 : called address, calling address, QoS, …

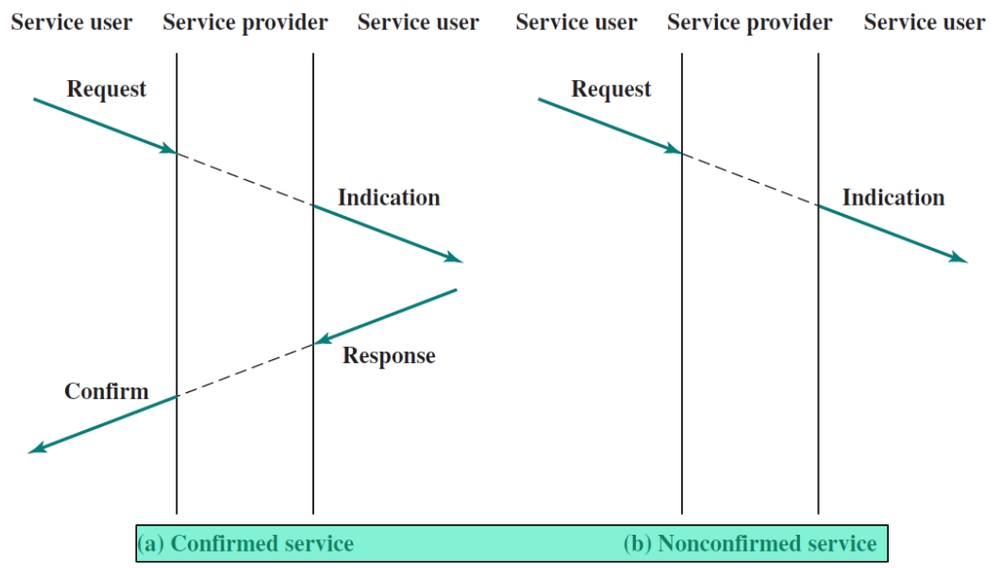




Protocol Model : Service Primitive

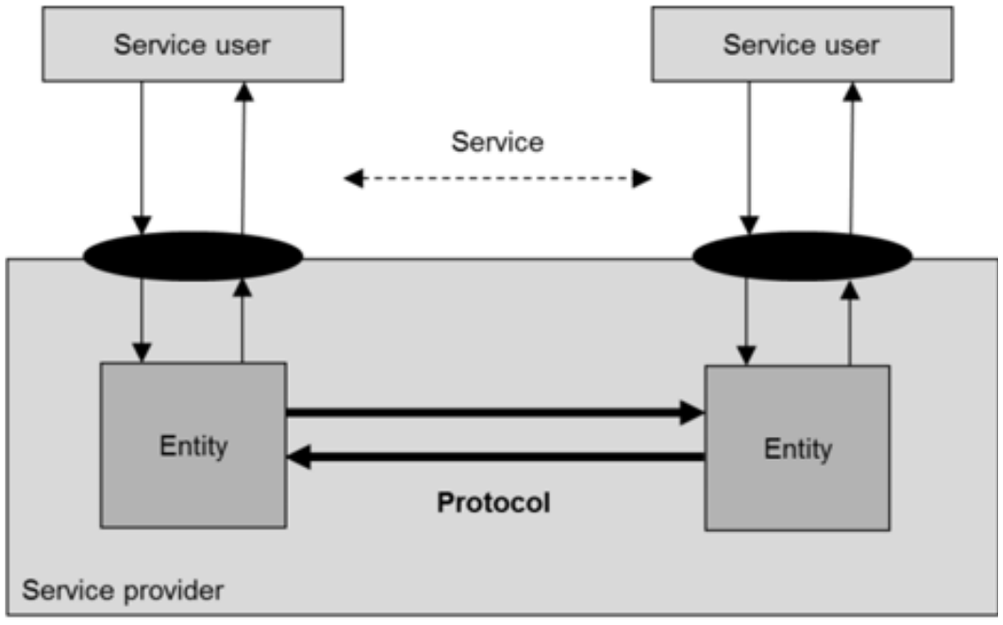
- Service primitive 종류

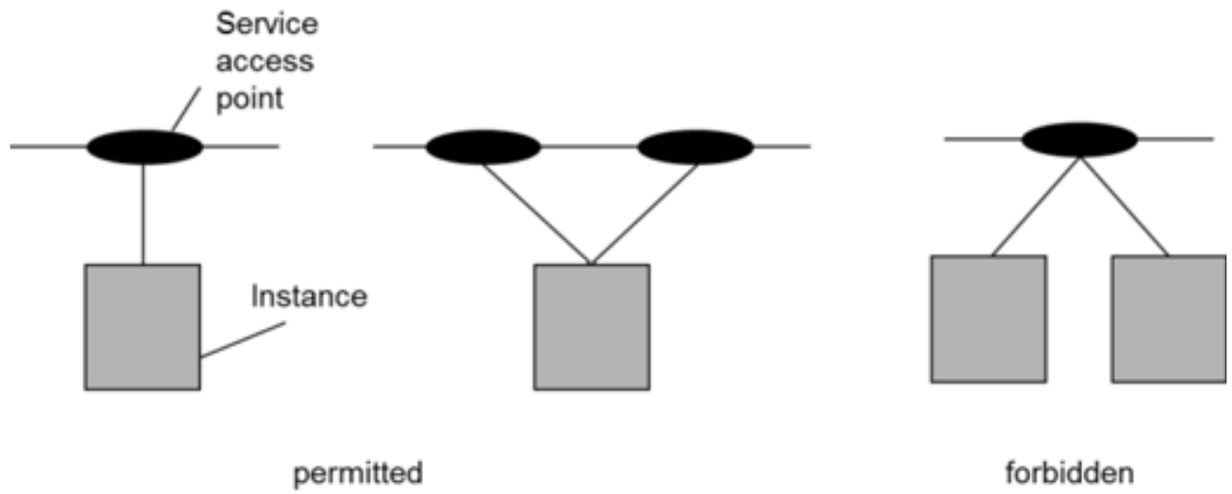
REQUEST	A primitive issued by a service user to invoke some service and to pass the parameters needed to specify fully the requested service
INDICATION	A primitive issued by a service provider either to 1. indicate that a procedure has been invoked by the peer service user on the connection and to provide the associated parameters, or 2. notify the service user of a provider-initiated action
RESPONSE	A primitive issued by a service user to acknowledge or complete some procedure previously invoked by an indication to that user
CONFIRM	A primitive issued by a service provider to acknowledge or complete some procedure previously invoked by a request by the service user



Protocol Model : Entity and SAP

- Entities : service provider 내에서 서로 interacting하면서 실제 동작을 하는 존재
    - communication service 실현을 위해 서로 message 를 주고 받음
    - serves SAPs through peer entity
        - SAP를 통해 받은 SP를 기반으로 동작



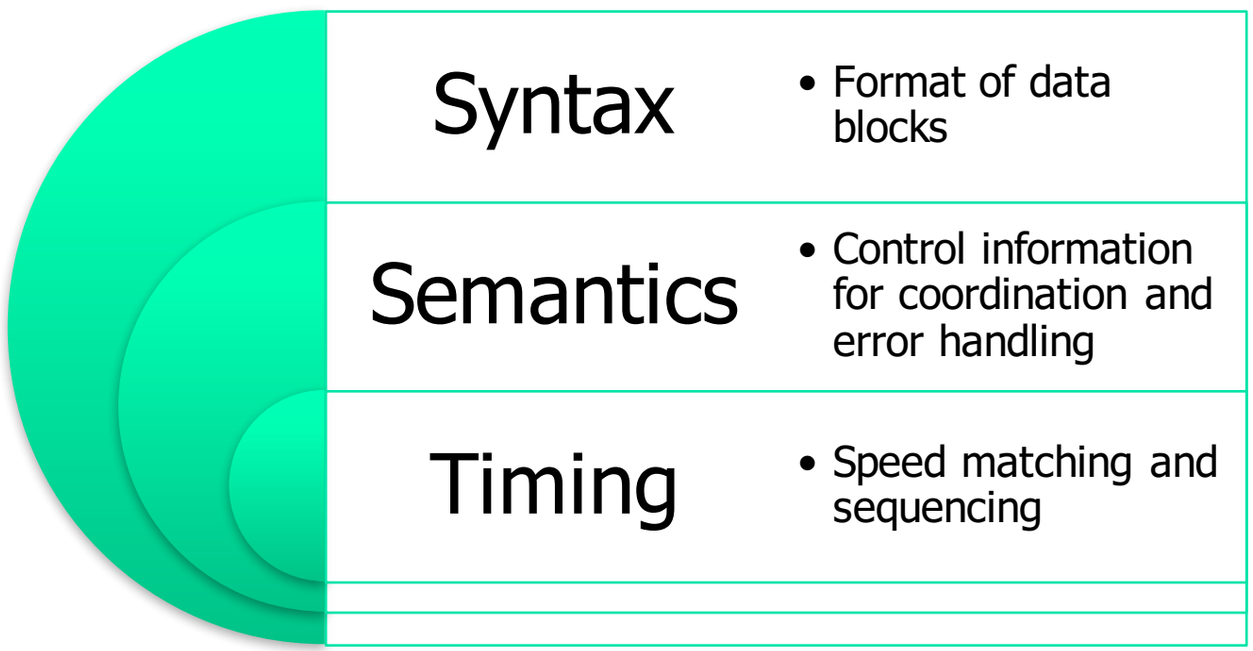


Protocol Model : Service vs. Protocol

- Communication protocol
  - peer entity들 간 상호작용을 하는 규칙
  - peer entity끼리 서로 message를 주고 받을 때 포멧이나 순서 등의 동작을 약속한 것
  - communication procedure that iterates
  - symmetric / asymmetric
- Service provider 내에서 service를 실현하기 위한 도구/규칙이 바로 protocol임

Protocol Model : Protocol 구성

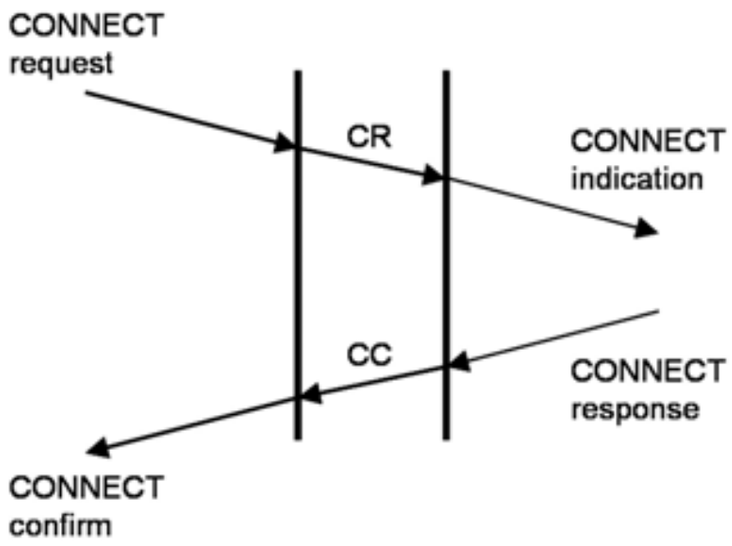
- Syntax : PDU에 대한 형식
- Semantics : protocol 동작을 위한 제어 정보
- Timing : procedure에 대한 동작 순서



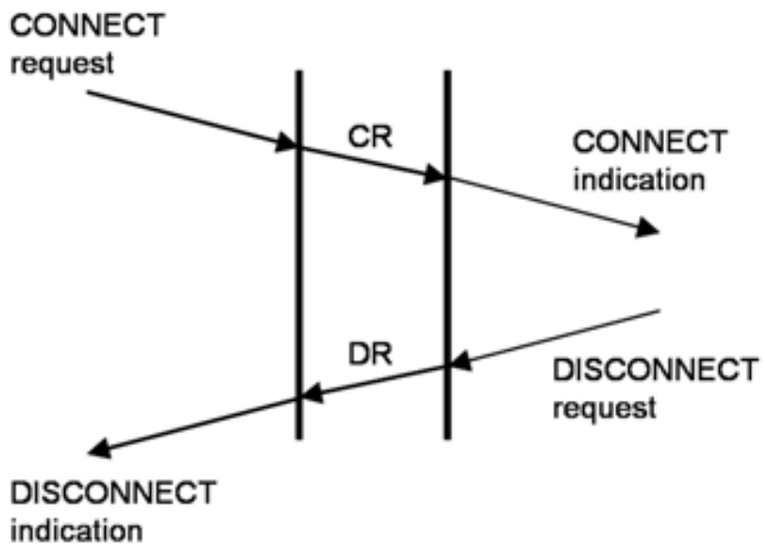
Protocol Model : Protocol Representation

- Time sequence diagram : 시간 흐름에 따른 entity간 상호 작용을 그래픽하게 나타냄

  - 상호작용은 곧 message를 주고받는 동작이므로 이것을 보기좋게 표현



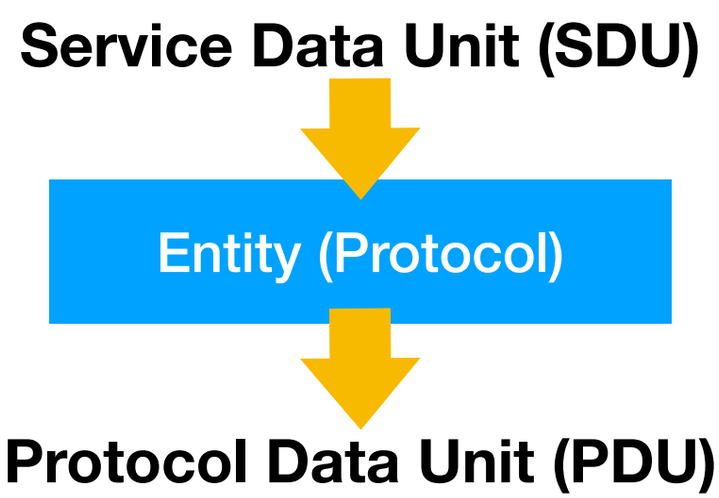
a) Successful connection establishment

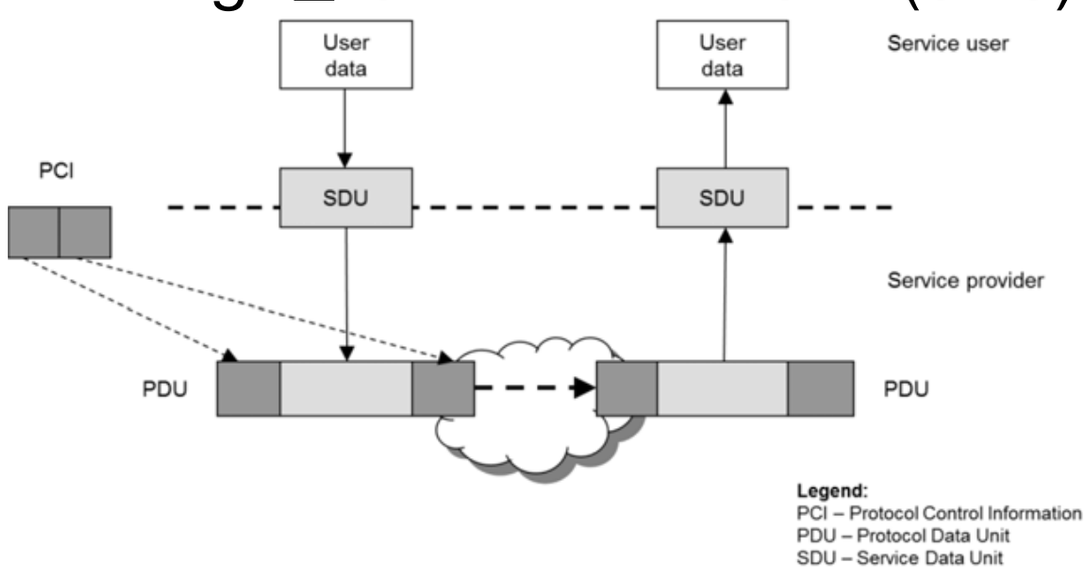


b) Rejected connection set up

Protocol Model : Protocol Data Units

- peer entity간 서로 주고받는 message를 PDU라고 함
  - PDU의 format (structure, semantics)은 protocol에 의해 정의됨
    - peer entity 양쪽에 format이 모두 알려져 있고, 한 PDU에 대해 양쪽에서 동일하게 해석
    - 보통 한 protocol은 여러 종류의 PDU 활용
- 반대로 SAP로 peer entity로 유입되는 message는 Service Data Unit (SDU)





Principle of Transparency

- service의 user data (SDU)에 대한 전달
    - Peer entity는 SP로 받아 PDU 형태로 전달
- service provider는 user data에 대해 principle of transparency 적용
    - user data를 조작없이 그대로 통과
    - user data 내용을 참고하여 동작하지 않음
- Protocol Control Information (PCI)
    - protocol 동작을 위해 SDU 앞뒤에 붙이는 제어 정보
    - Protocol header / trailer
    - 송신 측 entity에서 붙이고, 수신측 entity 에서 제거

Protocol Functions

- 여러 protocol에서 두루 사용되는 특정 procedure / mechanism
  - error control
    - PDU가 정상적으로 전달되지 않은 상황에서의 protocol entity의 동작
  - fragmentation, flow control
    - entity간 data를 서로 주고받는 속도 및 형태 조절

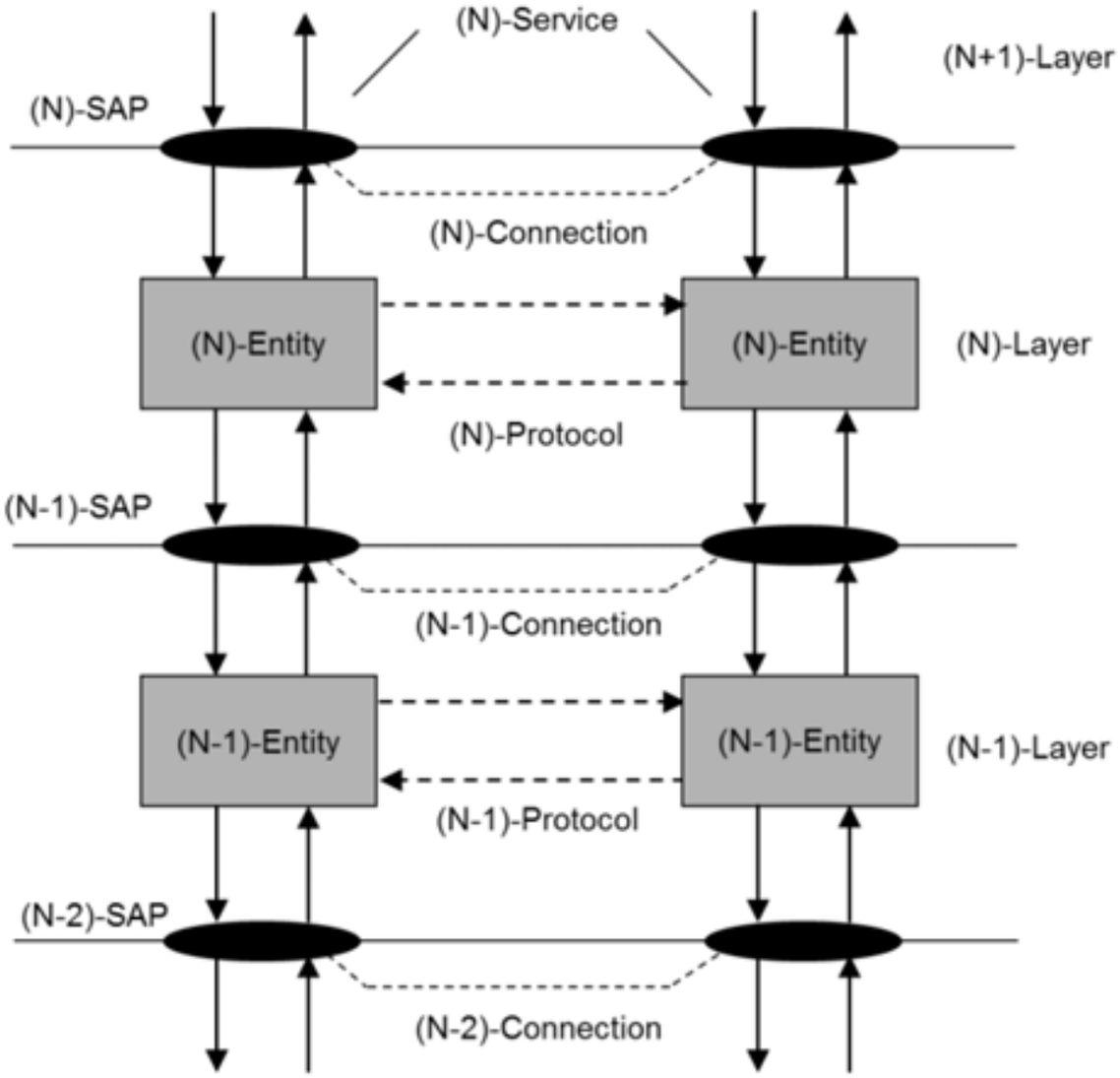
Protocol 동작의 특성

- Concurrency
  - entity는 어떤 순간에도 서로 다른 service demand 및 event에 대해 동시적으로 반응동작해야 함
  - 예시 : PDU coding 도중에 관련 연결이 끊어져서 재연결 동작을 자연스럽게 할 수 있어야 함
- Nondeterminism
  - 여러가지 event가 동시에 발생하는 경우 어떤 순서로 처리가 될지에 대한 예측이 되지 않는 특성
  - 어떤 event든 먼저 처리가 될 수 있어야 함

Layers

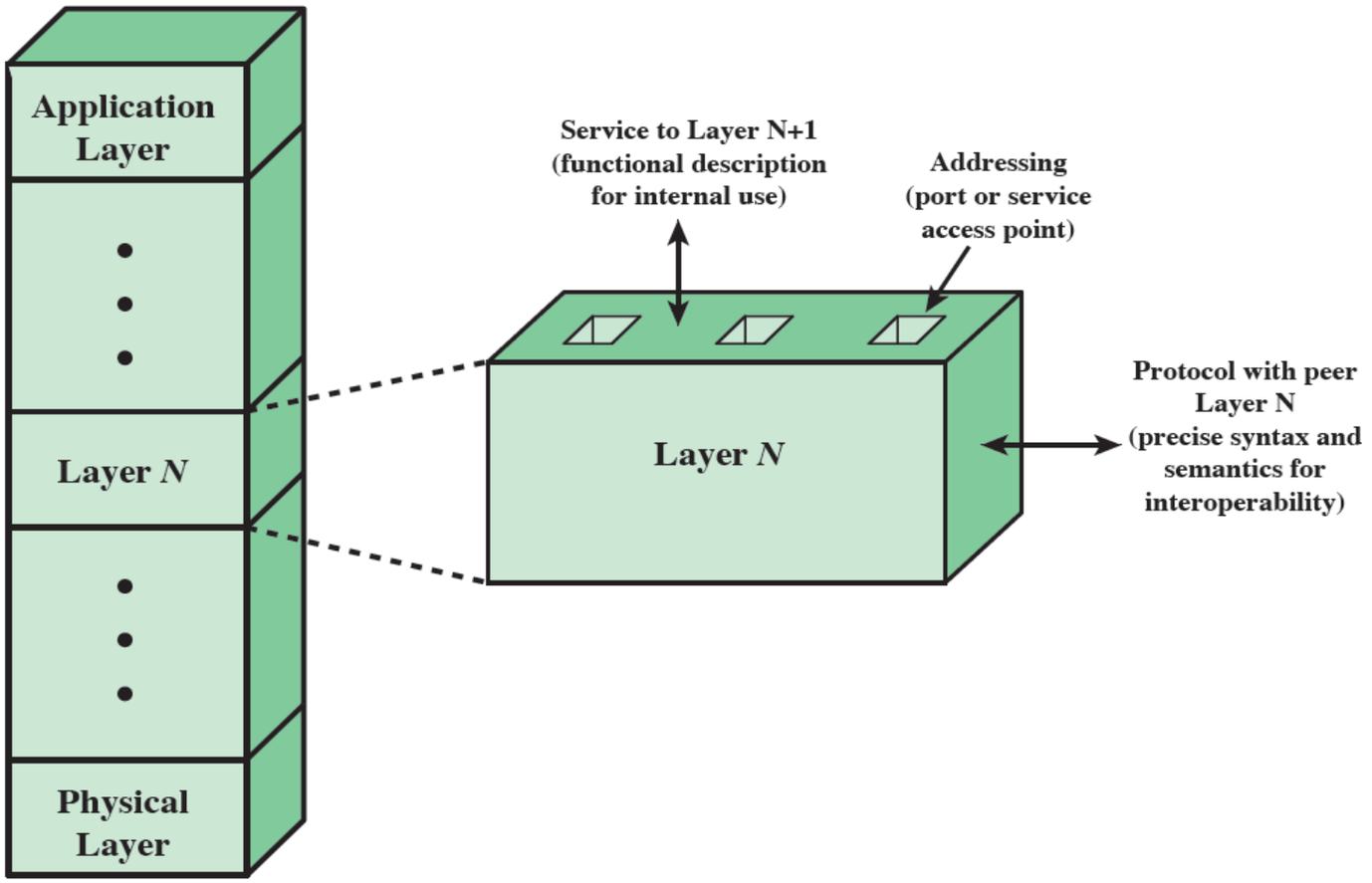
- layer : SAP-entity로 구성된 것
  - 각 layer는 하나 이상의 service를 제공

- Layered architecture
  - layer가 위아래로 존재
    - N-layer
  - peer entity간 message 교환은 인접한 하위 layer의 entity에 의해 이루어짐



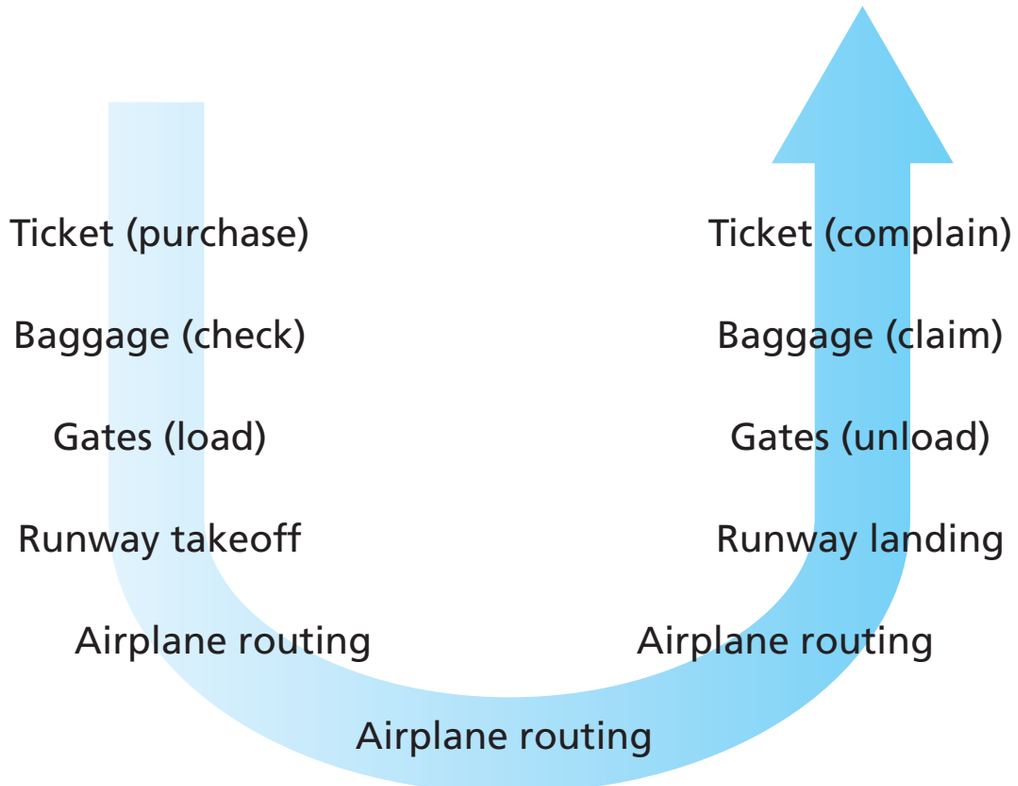
Layers

- Layer간 상호 작용은 service primitive를 통해 이루어짐
  - primitive : function to be performed
  - parameter : passing data/ control information



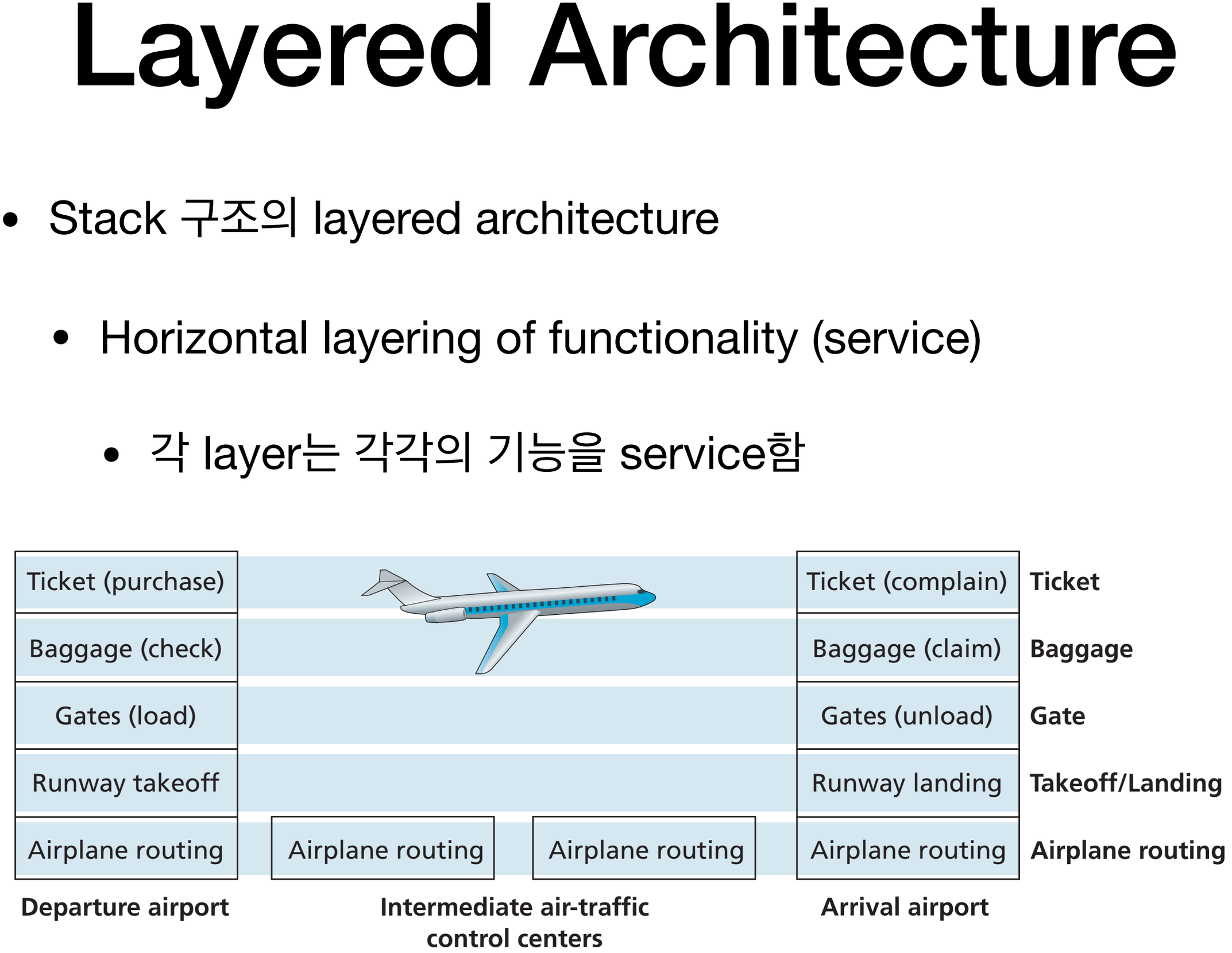
Layered Architecture

- Protocol stack도 결국 layered architecture의 한 사례
- 일상에서도 layer architecture를 찾아볼 수 있음
- 예시 : 항공 서비스에 대한 시스템
  - 티켓팅, 수화물, 게이트 통과, …
  - series of actions



Layered Architecture

- Stack 구조의 layered architecture
  - Horizontal layering of functionality (service)
  - 각 layer는 각각의 기능을 service함



Layered Architecture 특징

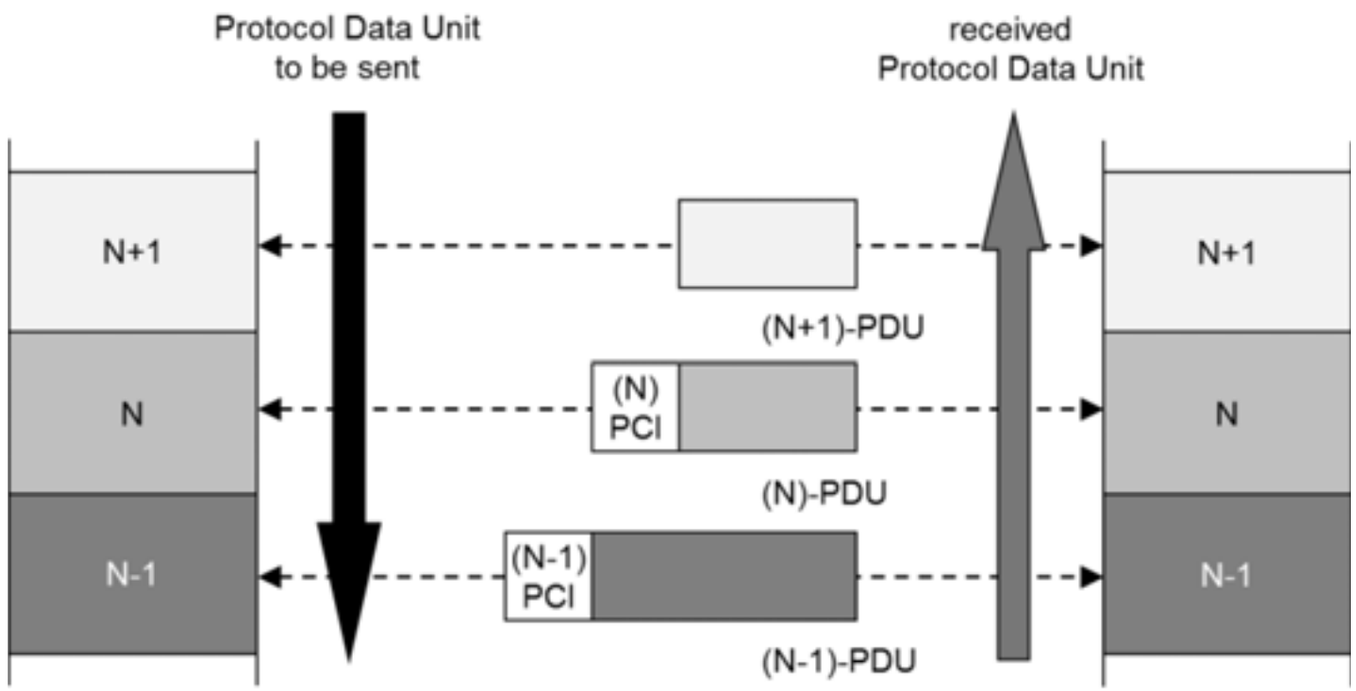
- horizontal interaction : peer entities끼리만 interaction이 이루어지며, 위아래 layer간 interaction은 없음
    - 하위 layer로부터 service는 받으나, 하위/상위 간 동작에 영향을 서로 주지는 않음
- principle of transparency : SDU를 그대로 통과시키며 protocol 동작이 SDU에 의해 영향을 받지 않음
- 상위 layer의 interaction이 하위 layer의 동작을 줄여주지 않음
- 하위 layer의 동작상 문제는 상위 layer에게 알려지지 않을 수 있음
    - 보통 자체적으로 문제 해결이 이루어지며, 치명적 문제 경우에만 inform을 줌

Layered Architecture 특징

- 크고 복잡한 기능들의 시스템을 표현하기 적절함
  - Modularity : 기능 단위로 시스템을 분해
    - 각 layer의 service에 대해 비교적 쉽게 구현
    - 일부 기능에 대한 수정이 용이
- 기능의 series 형태로 표현되므로 동작 흐름이 한눈에 보임
  - 동작 분석 및 디버깅이 용이

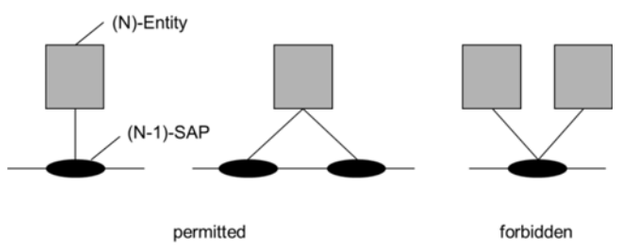
PDUs of Layered Architecture

- N Layer에서 발생한 PDU가 N-1 layer의 SDU가 됨
- 하위로 갈수록 PDU 크기가 일반적으로 커짐
    - N -> N-1 layer로 갈수록 PCI가 붙으면서 PDU가 커짐
- fragmentation이 일어날 수 있으나 principle of transparency는 보존됨



Mapping on the (N-1) Layer

- N layer의 entity는 여러개의 N-1 layer의 entity와 연결될 수 있음
  - 아래 방향으로 여러 SAP로 분기되는 형태는 가능
- N layer 의 여러 entity가 하나의 N-1 layer entity에 연결되면 안됨
  - N-1 layer entity가 routing을 하려면 SDU 내용을 알수 있어야 하며, 이는 principle of transparency 위반

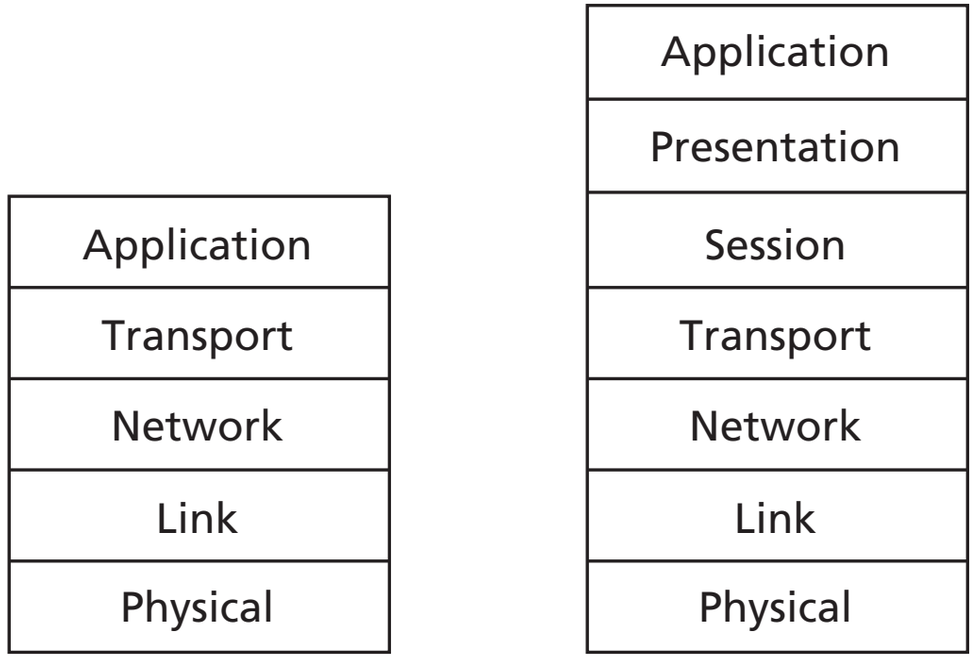


Open Systems Interconnection (OSI) Model

- International Organization for Standardization (ISO)에서 제안한 통신 네트워크의 표준 계층 구조

  - 1970년대 만들어질 당시는 7계층 구조

  - 현 internet은 5계층 모델

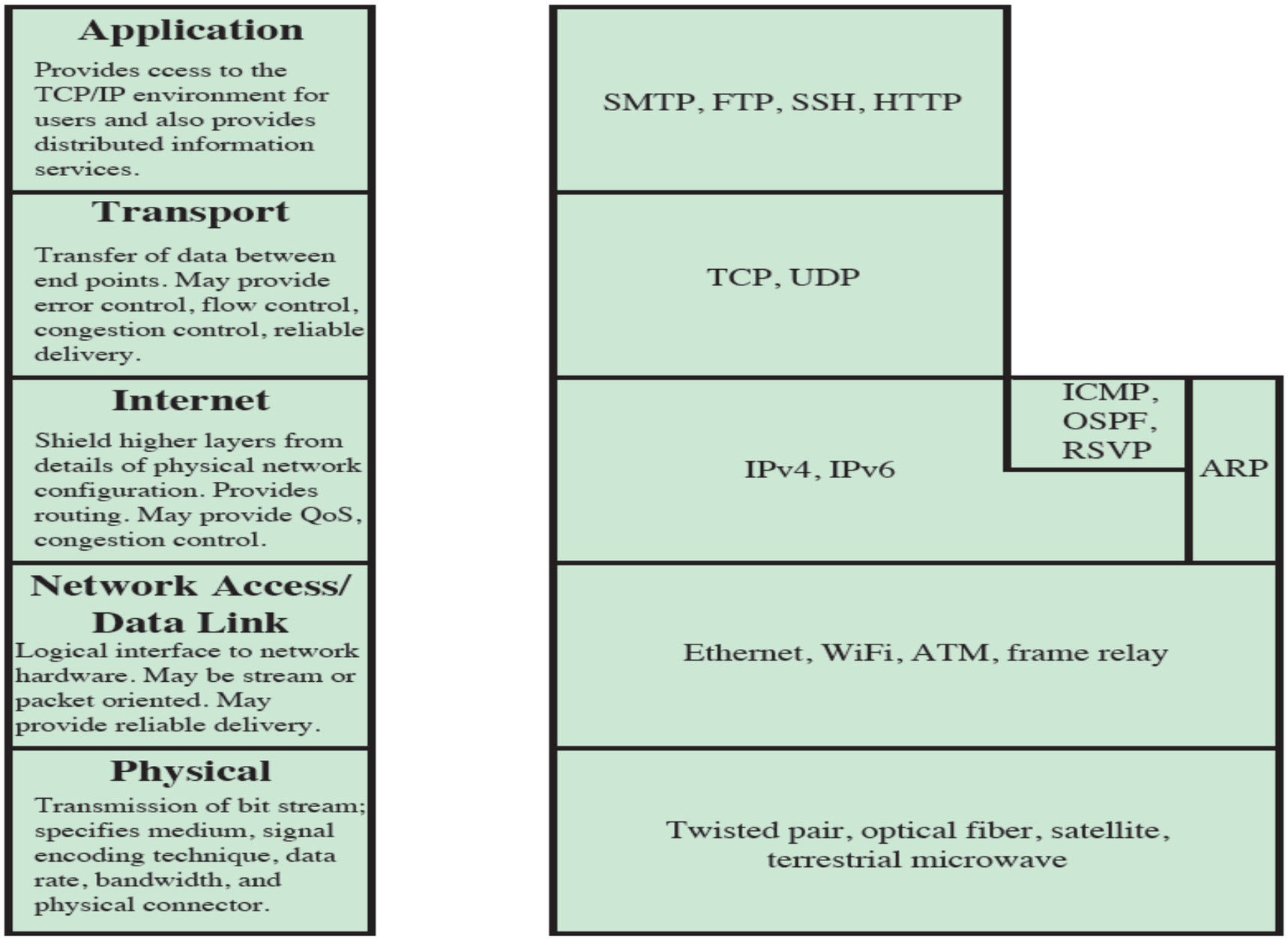


a. Five-layer Internet protocol stack

b. Seven-layer ISO OSI reference model

Open Systems Interconnection (OSI) Model

- Example protocols



OSI Model

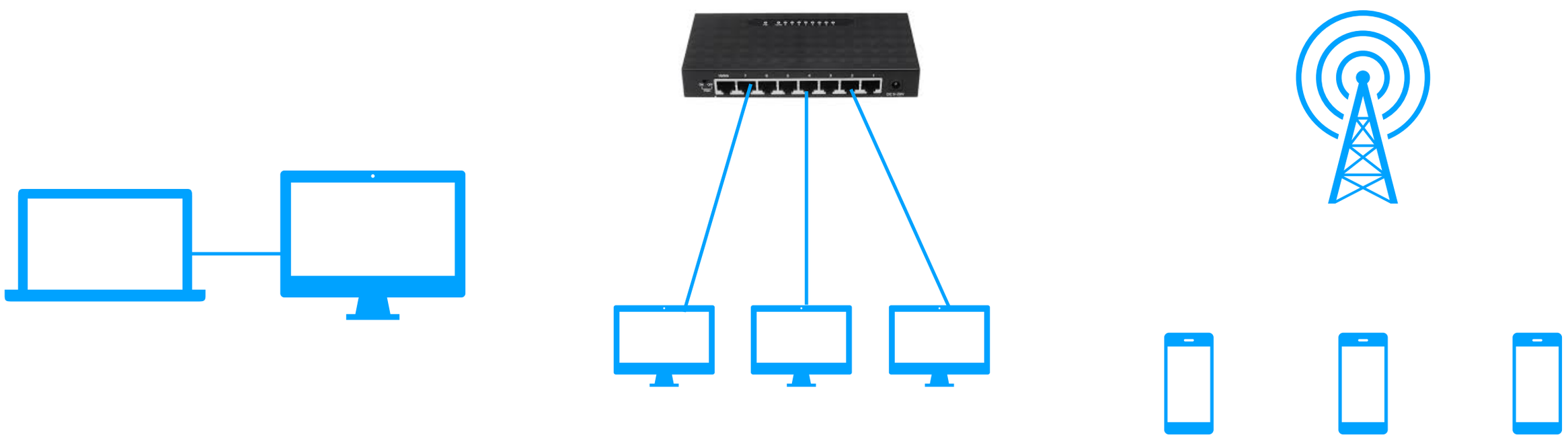
- L1 : Physical layer (link layer)
  - 물리적으로 연결된 링크에 맞는 실제 물리 신호를 생성
  - 1 hop으로 연결된 상대방에게 직접 정보 전달
- 링크나 물리 매체에 따라 상이함
  - Ethernet : twisted-pair copper wire, coaxial cable
  - WiFi, LTE, … : wireless signal

OSI Model

- Physical layer issues
  - Characteristics of transmission medium
  - Nature of the signals (신호및시스템, 통신이론)
  - Data rates

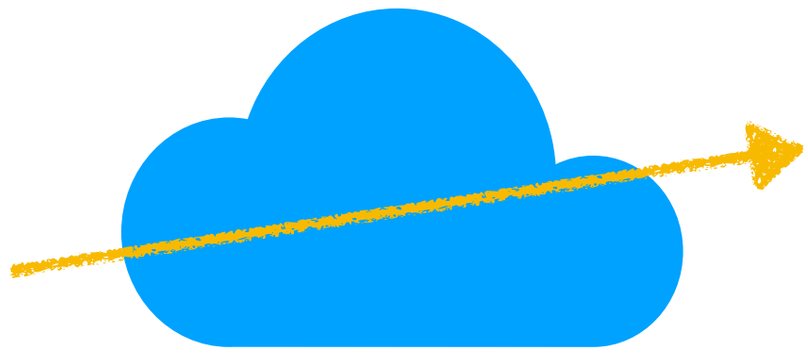
OSI Model

- L2 : Data link layer, Medium Access Control (MAC) layer
  - 통신 매체에 대한 접근 제어
  - 여러 개체가 하나의 통신 매체를 공유하는 것에 대한 교통정리
  - L1 과 보통 세트이며, 링크의 물리 매체 특성에 영향을 많이 받음



OSI Model

- L3 : Network layer (Internet Protocol (IP)) -> 우편 배달 시스템
  - 네트워크 내에서의 packet 흐름을 결정하는 핵심 역할
  - 우편 서비스에서 보내는 사람 -> 받는 사람으로 우편물 전달 역할과 동일
    - Routing protocol
- L4 : Transport layer (TCP, UDP) -> 우체통
  - application endpoint 간 packet 을 전달하는 역할
  - TCP : connection-oriented, flow control
  - UDP : connectionless service (던지고 끝)

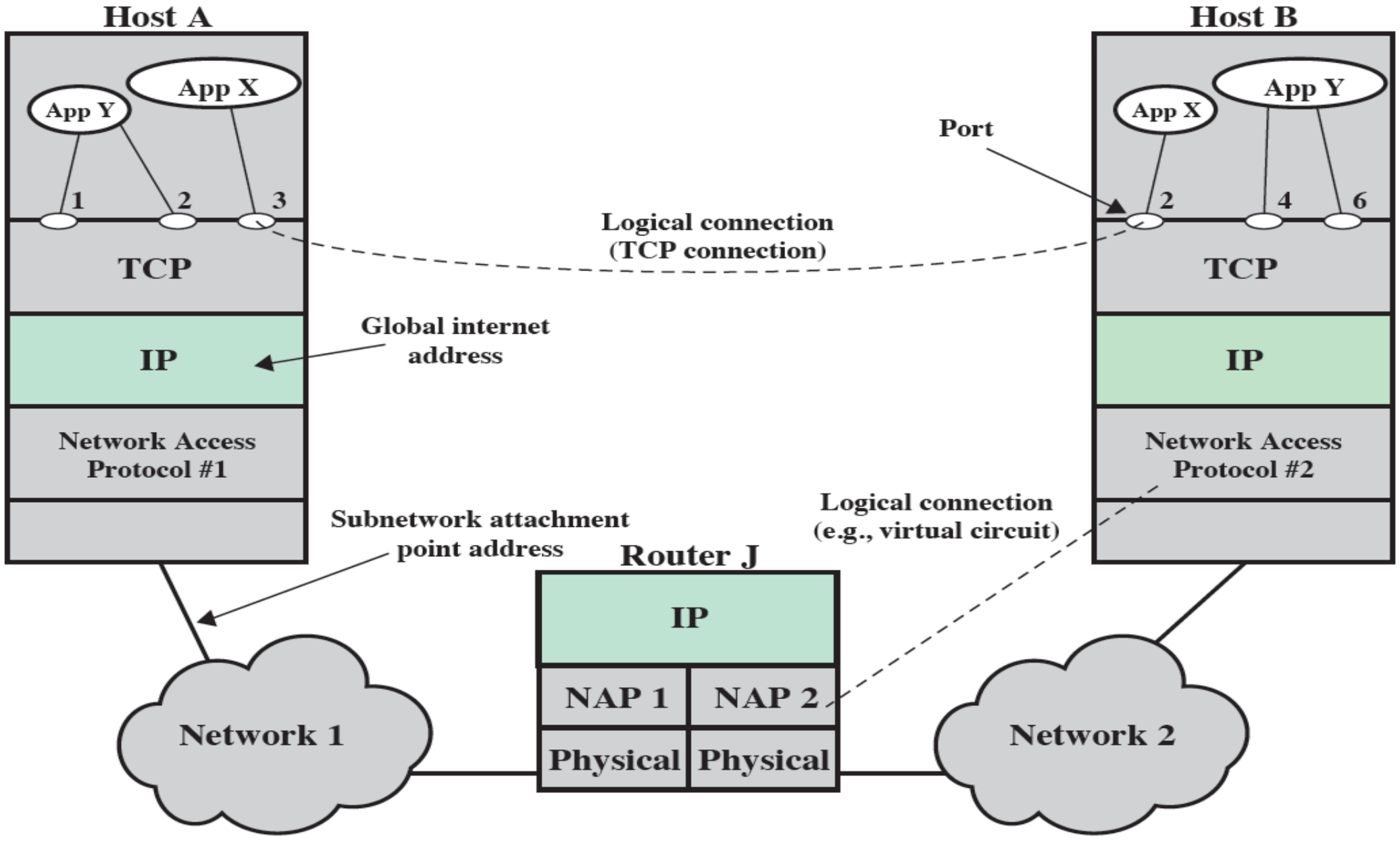


OSI Model

- Application layer
  - 통신 네트워크를 사용하는 실제 프로그램 혹은 app.
    - HTTP
    - SMTP
    - FTP
    - DNS

OSI Model

- TCP/IP 기반 protocol stack 예시

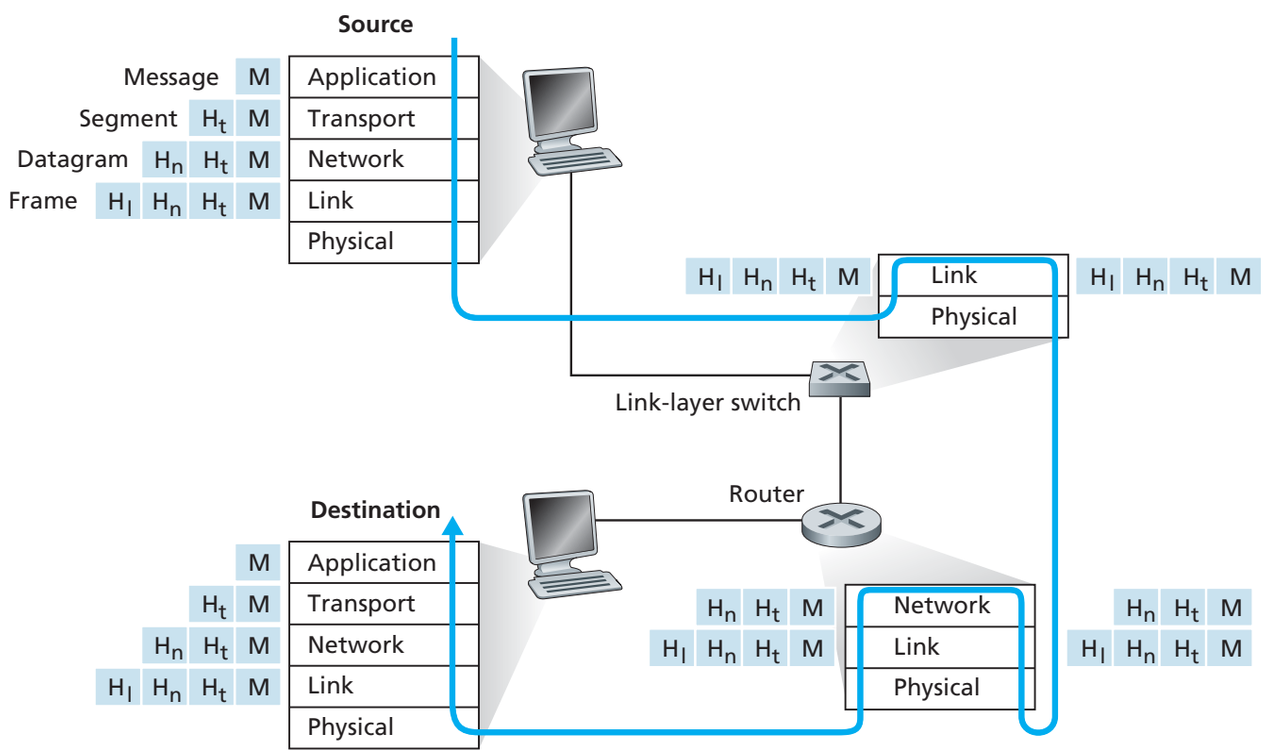


OSI Model의 실물 구현

- Application, transport layers
  - 보통 SW이며, end-system에만 존재
- Physical, data link layers : 특정 통신 link에 국한된 기능
  - network interface card에 구현됨 (HW + SW)
- Network layer : 네트워크를 통한 packet 전달
  - end-system / packet switch 등에 분산되어 동작
  - HW + SW

Encapsulation

- OSI model 관점에서 본 Internet 상에서의 packets 전달 예시
  - switch/router는 모든 layer 가 없고, 하위 layer들만 가짐
- 각 layer가 자신만의 정보(header)를 붙이면서, 상위 layer의 packet을 encapsulation함



Encapsulation

- PDU 전달 예시