Sistem terdistribusi 3

Interprocess Communication

Prinsip berkomunikasi

- Source
  - generates data to be transmitted
- Transmitter
  - Converts data into transmittable signals
- Transmission System
  - Carries data
- Receiver
  - Converts received signal into data
- Destination
  - Takes incoming data

Interprocess Communication

- Processes within a system may be **independent** or **cooperating**
- Reasons for cooperating processes:
  - Information sharing
  - Computation speedup
  - Modularity
  - Convenience
- Cooperating processes need **interprocess communication** (**IPC**)
- Two models of IPC
  - Shared memory
  - Message passing

Karakteristik IPC

- Synchronization dan Asynchronization.
- Message destination : Internet address dan local port.
- Reliability : validity dan integrity.
- Ordering.

Kategori IPC

- **Pipes** : merupakan fasilitas yang menyediakan komunikasi satu arah antar proses dalam sebuah system atau disebut half-duplex, yaitu data mengalir hanya terjadi satu arah.
- **FIFO** : fasilitas komunikasi secara FIFO (first in first out).
    - Mirip dengan Pipes
- **Shared memory** : suatu proses berbagi ruang dalam virtual address, sehingga proses manapun akan berbagi wilayah memory akan mampu menulis dan membacanya.
    - Dalam single processing

Kategori IPC

- **Mapped memory** : berhubungan dengan mapping sebuah file dalam file system sesuai dengan memory yang ada.
  - Konsep virtual memory
- **Message Queues** :mengirim pesan secara asynchronous.
  - Asynchronous berarti proses pengiriman data berlanjut disertai sebuah eksekusi tanpa harus menunggu penerima menerima atau mengenal informasi tersebut.

Kategori IPC

- **Semaphore** : struktur data yang di share ke beberapa proses untuk sinkronisasi
- **RPC**: adalah sebuah protokol yang memungkinkan program komputer berjalan pada satu host dan mengakibatkan kode dapat dieksekusi pada host yang lain tanpa kebutuhan programmer secara eksplisit pengkodekan ini.
- **Socket** : sebagai endpoint dari komunikasi dua proses pada dua buah sistem komputer. Dalam pengiriman dua buah proses tidak dapat melalui port yang sama

Message Passing

- Basic Operations
  - Send
  - Receive
- Variations
  - Connection-oriented vs Connectionless
  - Synchronous vs Asynchronous
  - Buffered vs Unbuffered
  - Reliable vs Unreliable
- Data representation
  - Marshalling

Communication Link

- Properties of communication link
  - Links are established automatically
  - A link is associated with exactly one pair of communicating processes
  - Between each pair there exists exactly one link
  - The link may be unidirectional, but is usually bi-directional

Direct Communication

- Processes must name each other explicitly:
  - **send** (*P, message*) – send a message to process P
  - **receive**(*Q, message*) – receive a message from process Q
- Properties of communication link
  - Links are established automatically
  - A link is associated with exactly one pair of communicating processes
  - Between each pair there exists exactly one link
  - The link may be unidirectional, but is usually bi-directional

Indirect Communication

- Messages are directed and received from ports
  - Each port has a unique id
  - Processes can communicate only if they share a port
- Properties of communication link
  - Link established only if processes share a common port
  - A link may be associated with many processes
  - Each pair of processes may share several communication links
  - Link may be unidirectional or bi-directional

Indirect Communication

- Operations
  - create a new/open port
  - send and receive messages through port
  - destroy a port
- Primitives are defined as:
  - **send**(*A, message*) – send a message to port A
  - **receive**(*A, message*) – receive a message from port A

Indirect Communication

- Mailbox sharing
  - P_1, P_2, and P_3 share port A
  - P_1, sends; P_2 and P_3 receive
  - Who gets the message?
- Solutions
  - Allow a link to be associated with at most two processes
  - Allow only one process at a time to execute a receive operation
  - Allow all processes receive the message
  - Allow the system to select arbitrarily the receiver. Sender is notified who the receiver was.

Buffering

- Queue of messages attached to the link; implemented in one of three ways
  - Zero capacity – 0 messages
    Sender must wait for receiver
  - Bounded capacity – finite length of *n* messages
    Sender must wait if link full
  - Unbounded capacity – infinite length
    Sender never waits
  - Unbuffering means no message queue

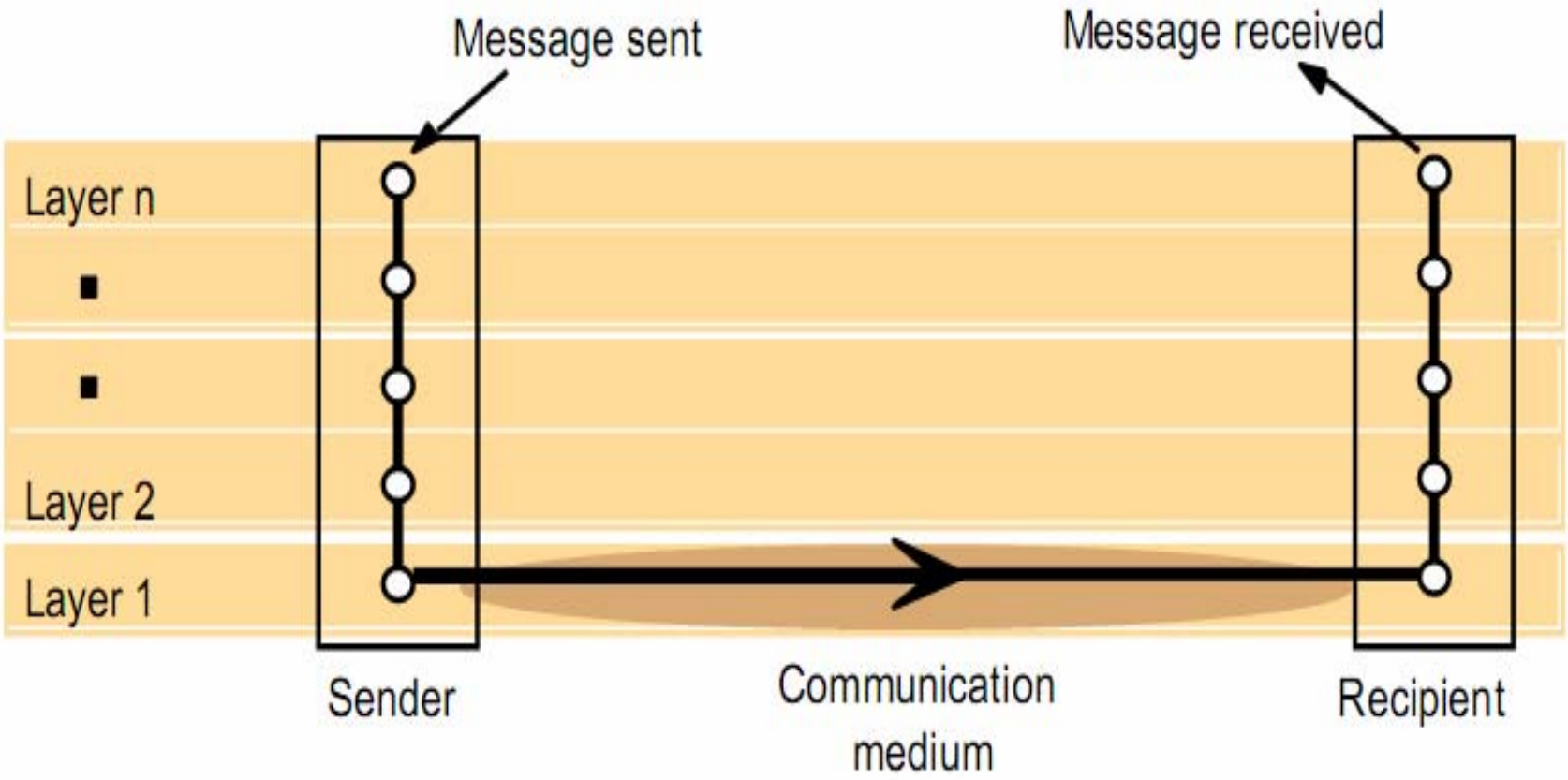
Syn & Asyn Comm

- Synchronous communication
  - Acknowledge must be received
  - Blocking communication
  - Sender/Recipient must both active
  - Example: Registration system @ UKDW
- Asynchronous communication
  - No acknowledge needed
  - Non-blocking communication
  - Message may be queued
  - Example: Email

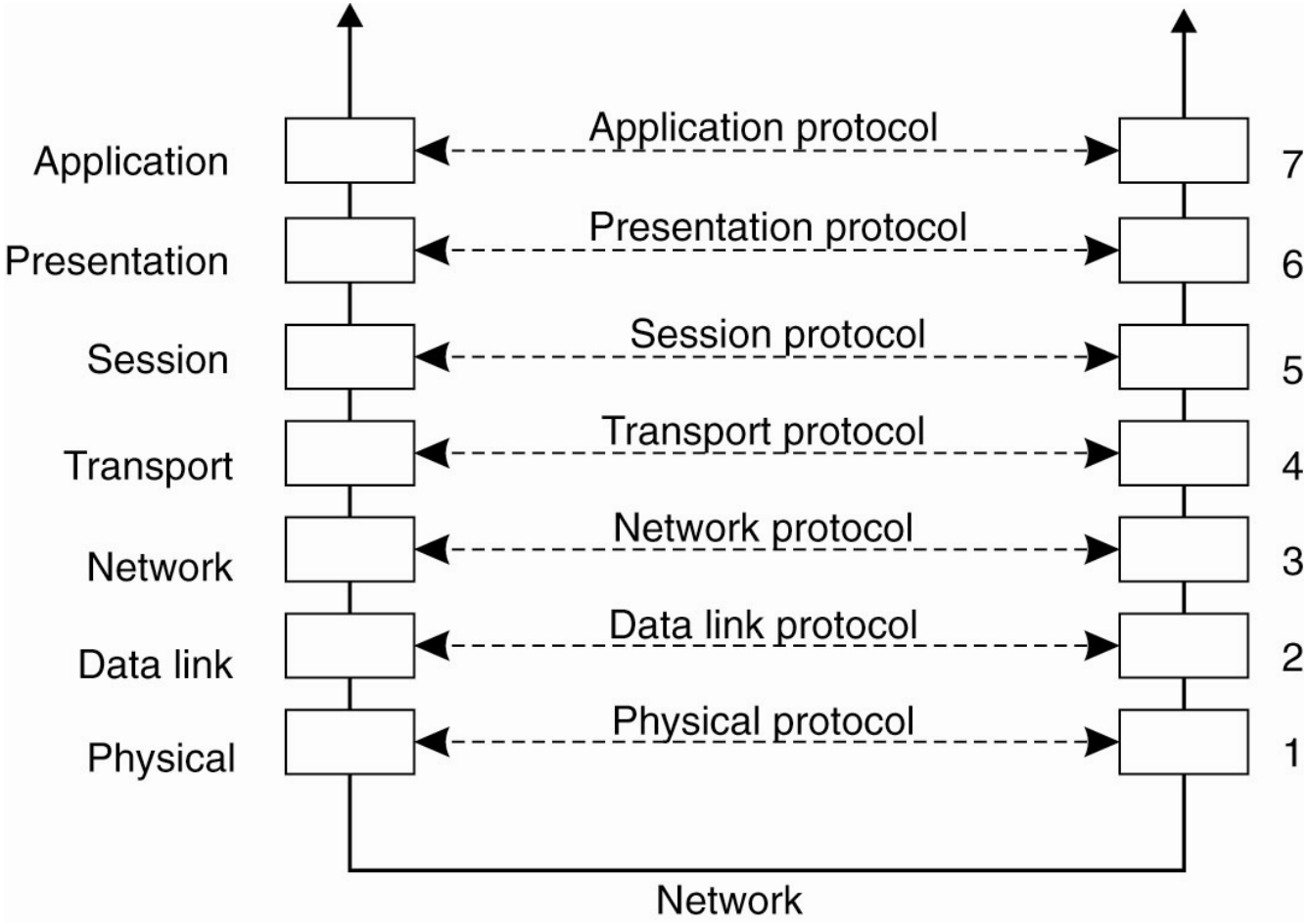
Transient & Persistent Comm.

- Transient Communication
  - Message discarded if failed to delivered immediately
  - Example: HTTP Request
- Persistent Communication
  - Message stored until receiver can accept it
  - Example: Email

Komunikasi Jaringan



Remember: OSI



OSI Layers (1)

- Physical
  - Physical interface between devices
    - Mechanical
    - Electrical
    - Functional
    - Procedural
  - Contoh: Ethernet CARD
- Data Link
  - Means of activating, maintaining and deactivating a reliable link
  - Error detection
  - Contoh: PPP

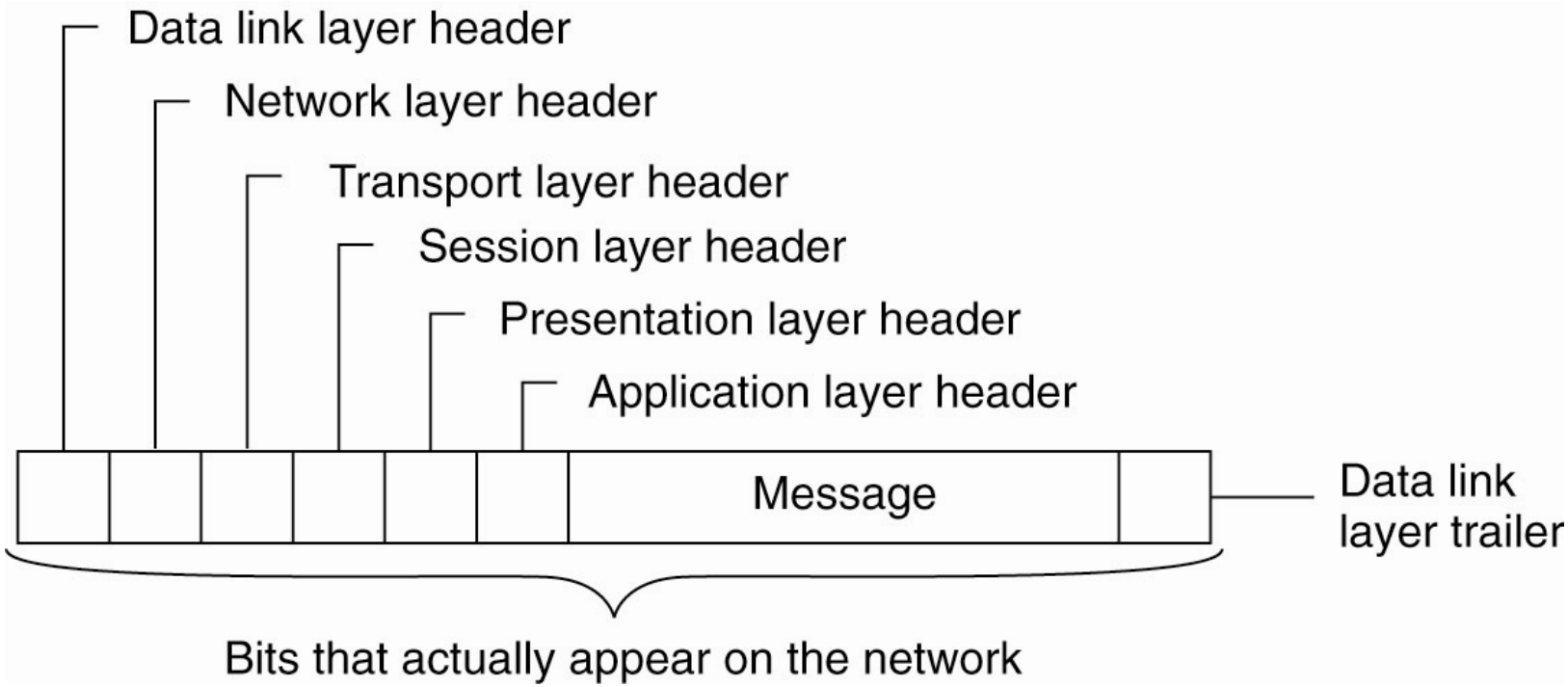
OSI Layers (2)

- Network
  - Transport of information
  - Contoh: Virtual Circuit & IP
- Transport
  - Exchange of data between end systems
  - Error free
  - In sequence / No sequence
  - No losses / losses
  - No duplicates
  - Quality of service
  - Contoh: TCP dan UDP

OSI Layers (3)

- Session
  - Control of dialogues between applications
  - Recovery
- Presentation
  - Data formats and coding
  - Data compression
  - Encryption
  - Contoh: SSL
- Application
  - Means for applications to access OSI environment
  - Contoh: HTTP, FTP, SMTP

The Message


Data link layer header
Network layer header
Transport layer header
Session layer header
Presentation layer header
Application layer header
Message
Data link layer trailer
Bits that actually appear on the network

Konsep Pengiriman Data

- Data dikirim dalam bentuk paket
- Setiap paket memiliki header untuk keperluan administrasi routing
- Data disimpan dalam body sebuah paket
- Ukuran paket sangat bervariasi
  - Ethernet: 64 – 1518 byte
- Bisa dikirimkan dengan TCP/UDP

TCP dan UDP

- Dua protokol untuk transport layer
- Menggunakan konsep port (16 bit) untuk membedakan aplikasi
  - HTTP: 80, HTTPS: 443, FTP: 21, ...
  - Port 1-1023 : well-known port
  - Port 1024-49151 : registered port
  - Port lain bisa digunakan secara bebas
  - Di Linux: /etc/services

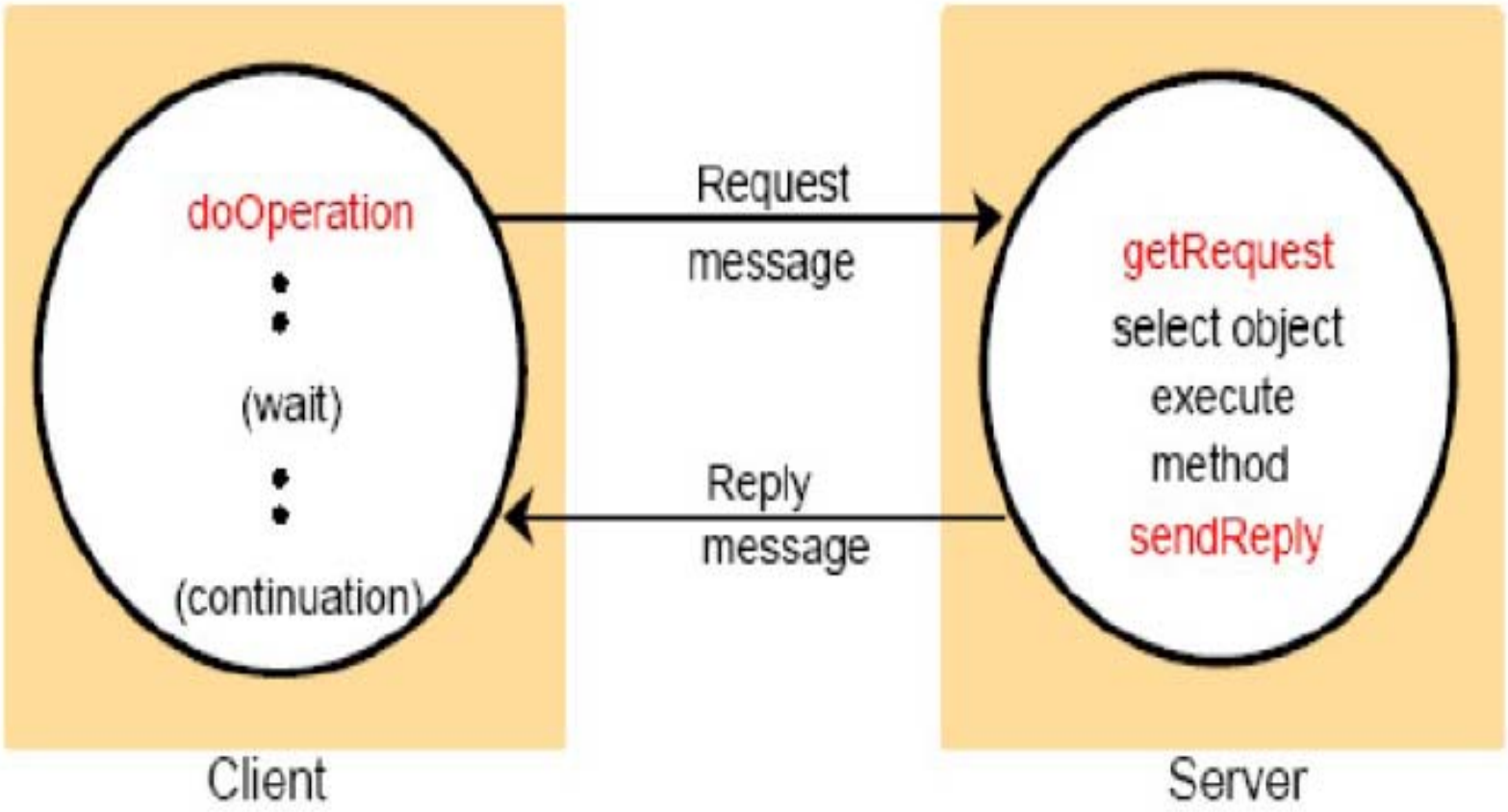
TCP

- Transmission Control Protocol, pada transport layer
  - Reliable connection
- Adanya pengecekan error
- Dijaga urutan message
- Komunikasi duplex – dua arah
- Segmentasi - TCP PDU
  - Called TCP segment
  - Includes source and destination port
    - Identify applications
    - Connection refers to pair of ports
- TCP tracks segments between entities on each connection

UDP

- User Datagram Protocol
- Not guaranteed delivery
- No preservation of sequence
- No protection against duplication
- Minimum overhead
- Adds port addressing to IP
- Contoh: DNS, streaming

Request / Reply



Failure Model of IPC

- **Timeout**, jika tidak dapat balasan, method doOperation akan mengirim terus request message sampai timeout.
- **Duplicate request message**, server menerima lebih dari sekali request message sehingga memprosesnya berulangkali.
  - solusi : request identifier & filter out duplicate.

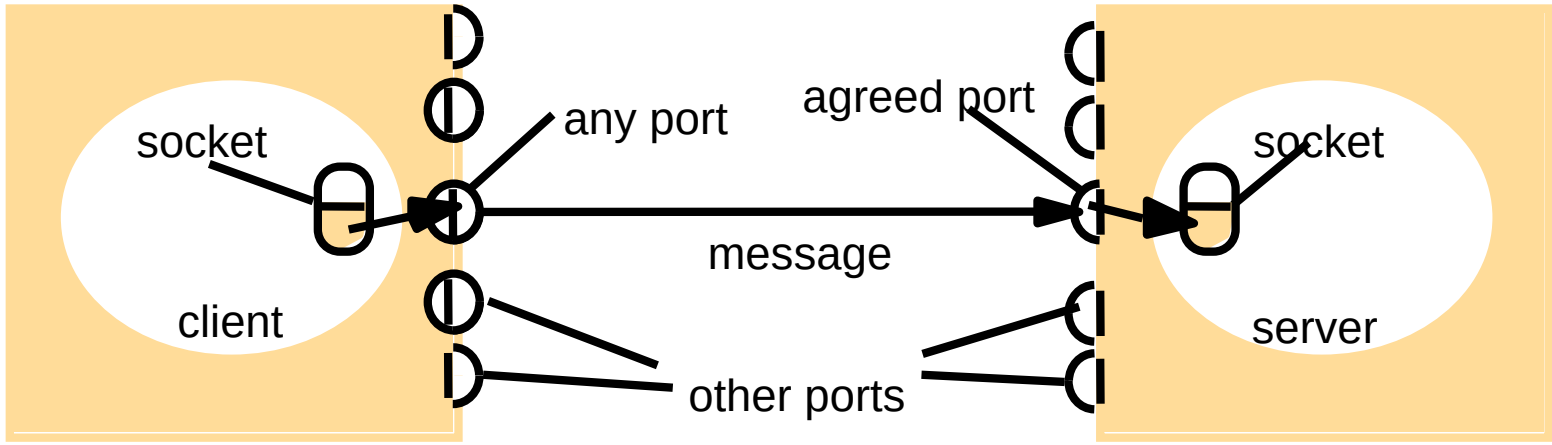
Failure Model of IPC

- **Lost reply message**, server dapat menyimpan hasil proses request message, jika ada request message yang sama tidak perlu diproses ulang, server mengirim reply message berupa hasil proses dari request message yang telah disimpan.
  - History, server menyimpan struktur rekaman reply message yang telah dikirim.

Socket

- Menyediakan jembatan komunikasi antar proses
- Komunikasi antar proses: mengirimkan pesan antar socket pada satu proses menuju sebuah socket pada proses lain
- Bisa menggunakan TCP/UDP
- Melakukan binding ke sebuah port tertentu

Sockets and ports



Internet address = 138.37.94.248

Internet address = 138.37.88.249

Operasi Socket

- Socket dapat melakukan operasi:
  - Koneksi ke mesin remote
  - Mengirim data
  - Menerima data
  - Mentutup koneksi
  - Bind to a port
  - Menerima koneksi dari mesin remote pada port tertentu
- Di tiap mesin yang saling berinterkoneksi, harus terpasang socket.

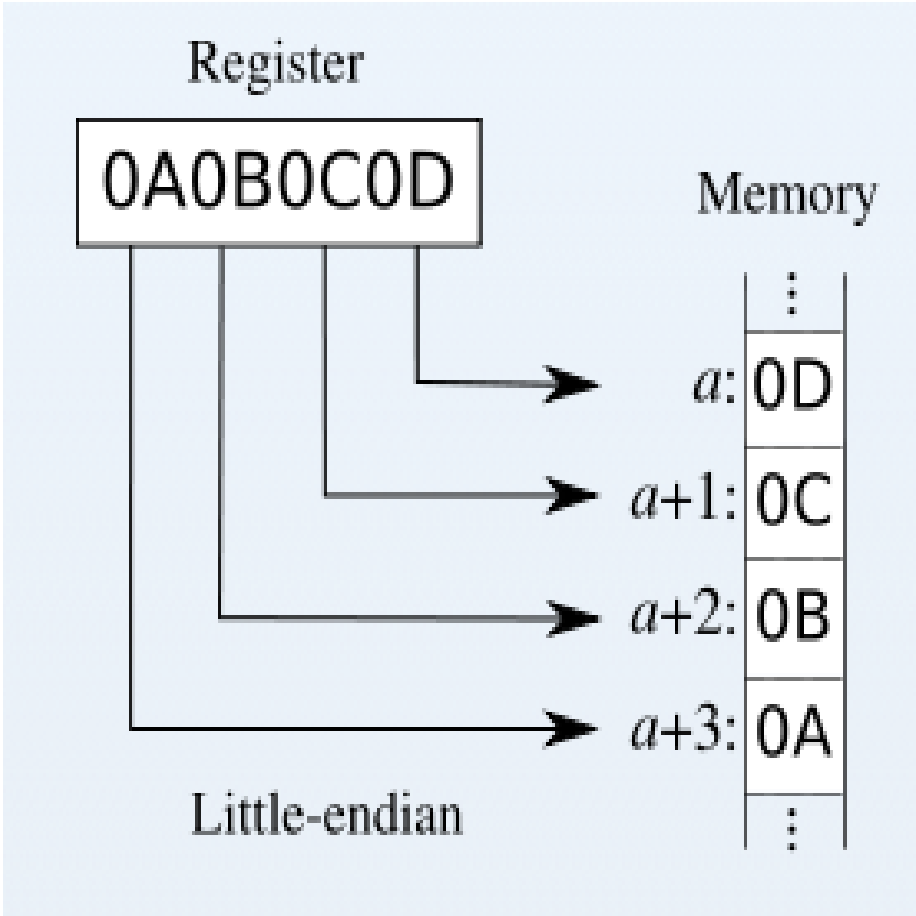
Socket API in Java

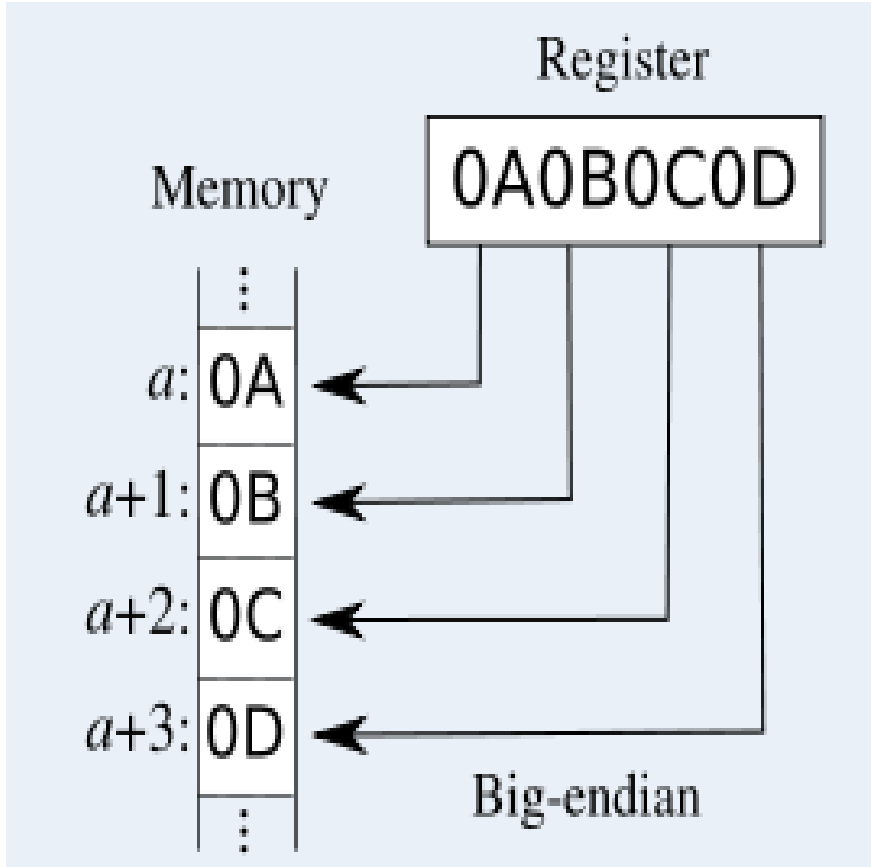
- Pada J2SE telah disediakan paket java.net yang berisi kelas- kelas dan interface yang menyediakan API (Application Programming Interface):
  - level rendah (Socket, ServerSocket, DatagramSocket)
  - level tinggi (URL, URLConnection).
- Disimpan pada package **java.net.***

Masalah Socket

- Informasi pada program : Struktur data
- Informasi pada message : Urutan byte
- Data harus dikonversi sebelum dan sesudah pengiriman agar bisa diproses dua pihak
- Masalah : representasi pada sistem bisa berbeda-beda
  - ASCII vs Unicode
  - Big-endian vs Little-endian

Big vs Little Endian system





Marshalling / Unmarshalling

- Proses konversi data menjadi bentuk yang cocok untuk transmisi pesan
  - Unmarshalling : proses kebalikannya
- Pendekatan yang umum digunakan:
  - CORBA dan RMI Marshalling
  - Java serialization

NEXT

- Distributed Object (CORBA)
- Remote Invocation (RMI)