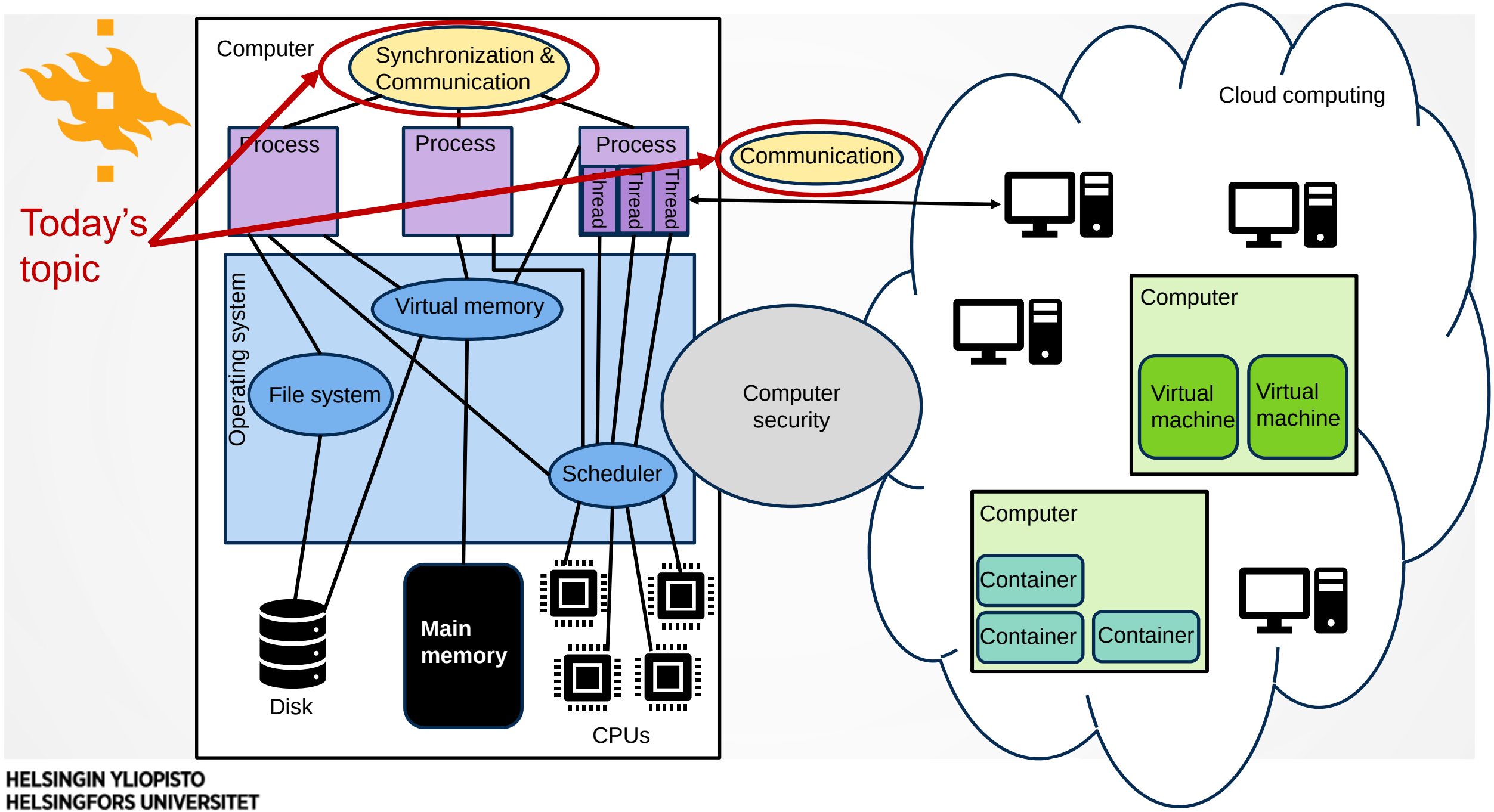
COMPUTING PLATFORMS

Interprocess communication:
Overview, background, alternatives, selection criteria

Mon 10.2.2025 Tiina Niklander

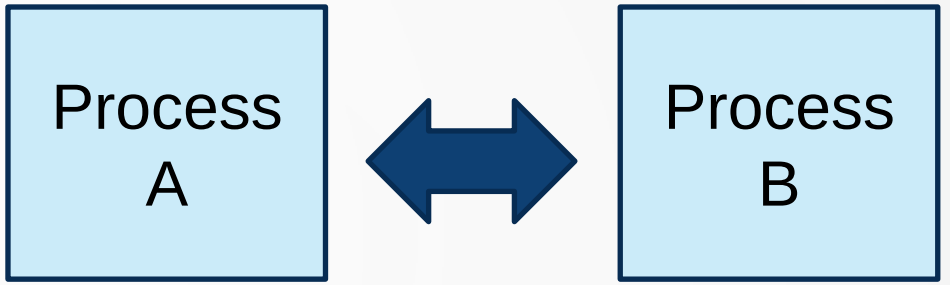

Today's topic
Computer
Synchronization & Communication
Process
Process
Process
Thread
Thread
Thread
Communication
Operating system
Virtual memory
File system
Scheduler
Computer security
Main memory
Disk
CPUs
Cloud computing
Computer
Virtual machine
Virtual machine
Computer
Container
Container
Container

CHECK YOUR COURSE EXAM REGISTRATION!

- Please remember to make separate registration to the course exam on time
- If you miss the registration deadline, you miss the exam

INTERPROCESS COMMUNICATION (IPC)

- Processes are isolated, but there is need for collaboration and coordination
- Several alternative methods and techniques that processes can use to exchange information
- Wikipedia: IPC are the mechanisms provided by an operating system for processes to manage shared data.



Coordinate actions

Synchronise operations

Exchange information

EXAMPLES

- Web-based services
  - Web browser needs to communicate with web server
  - Social media application needs to communicate with the server
- Parallel Computing
  - Used for example to model real-world events, like weather
  - One process can be running only on one core, but modern processors have multiple cores
- Others?

COORDINATION

Starting with
X: 10

Process A:

Read (X)
X = X+70
Write (X)

Process B:

Read (X)
X = X +5
Write (X)

- Value of X after execution?

15?

80?

85?

TYPES OF COMMUNICATION

- Shared memory *Same computer*
  - Normal memory operations, but memory area shared with other process(es)
  - Read and write memory content

- Message-passing *Same or different computer*
  - Information between processes is passed using special messages
  - Send and receive messages

Same or different computer

- Remote Procedure Call (RPC) / Remote Method Invocation (RMI)
  - One process can call a procedure or method of another process

TYPES OF COMMUNICATION, PART 2

- Signals
- Pipes / Named Pipes

Same computer

Must select one that is supported by API and OS

- Files

- Sockets
- Message queues

Same or different computer

Wed services have their own mechanisms, that are built using sockets

SYNCHRONOUS AND ASYNCHRONOUS

Synchronous

- Both present at the same time
- Real-world analogy: telephone call
- Rendezvous

Asynchronous

- Can be present at different times
- Real-world analogy: text message, letter

BLOCKING VS NON-BLOCKING

- Communication is blocking when a process has to wait for the other process to answer before it can continue
- Synchronous communication is always blocking: The first one arriving to synchronisation point has to wait for the other to arrive
- Asynchronous communication can be blocking or nonblocking

SHARED MEMORY

- Fast – direct access to memory by both processes (on same computer)

- Critical issues:
  - How to coordinate the operations by processes
  - Exclusive access

➡ Next lectures

- Python examples:
  https://docs.python.org/3/library/multiprocessing.shared_memory.html

SIGNALS

- Only in one computer
- Operating system specific

- Cannot be used to pass data, but just poke another process

- Needs process specific updates to signal handler on the receiving end

- Typically not used between user processes

FILES IN ONE COMPUTER

- Traditional way: One process writes to file, and then the other one reads from it
  - Very slow, if done frequently

- Memory-mapped files
  - Map the file to memory (like virtual memory page)
  - Operations on memory, but OS also updates the file to reflect the changes, but not immediately.
  - Can be used to implement shared memory
  - Python examples: https://docs.python.org/3/library/mmap.html

DISTRIBUTED FILE SYSTEMS

- Using same file in multiple computers, originally no concurrent access
- Distributed file systems (like NFS, SMB) allow file to be accessed from multiple computers
  - Network File System (NFS)
    - Developed by Sun microsystems in 1984
    - Used still in Unix-based systems like Linux
  - Server Message Block (SMB)
    - Developed at IBM in 1983
    - Used still in Windows to share files and other resources (like printers)
- Cloud-based services

PIPES AND NAMED PIPES (A.K.A. FIFO)

- Typical use on commandline: forward output of one process as input to another
  - ls | head 10
- Always unidirectional – One process writes and the other one reads
  - Stdin and stdout are examples of pipes used in linux

- Using pipes in python: https://docs.python.org/3/library/subprocess.html or https://docs.python.org/3/library/multiprocessing.html
- Named pipes (fifos): https://docs.python.org/3/library/os.html
  - Named pipe is created with command os.mkfifo
  - Used a lot like files

MESSAGE PASSING

Message-passing has multiple different alternatives

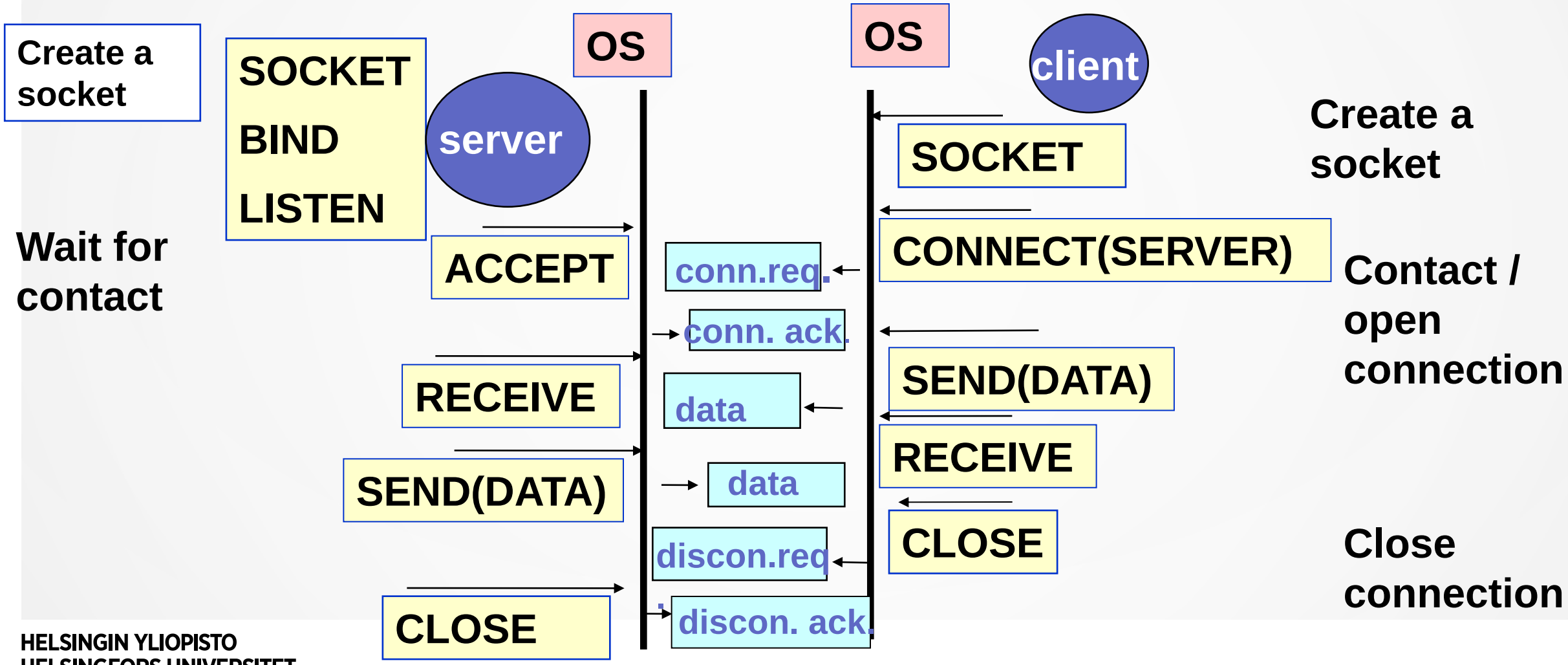
- Sockets are typically used with separate computers, but can be used in one computer
- Message queues have different implementations for use in one or more computers

SOCKETS

- Sockets are the basic communication interface in Internet-based applications
  - Can be used in one computer also (IP 127.0.0.1)
- Note: Underlaying all modern Internet-related communication mechanisms, even if not used as such
- Require both parties to be active at the same time (synchronous), but handling one message can be sychronous or asynchronous

- Using in python: https://docs.python.org/3/library/socket.html with or without https://docs.python.org/3/library/ssl.html
- Covered in Network Programming -course

TCP SOCKETS (AND MESSAGES)



MESSAGE QUEUES

- Message queues provide asynchronous mechanism to pass information from one process to another
  - One process writes / sends / produces a message to the queue
  - Other process reads / receives / consumes a message from the queue
  - Access to the queue needs some synchronization (like locks) ➡ Next lectures

- One computer using python: https://docs.python.org/3/library/multiprocessing.html
- Multiple computers: must have a queue manager (process) or use a message queue middleware like RabbitMQ, Apache Kafka, …

MESSAGE QUEUES

- Stallings calls message queues (MQ) mailboxes
  - unidirectional
- Process attach itself to MQ, where it gets/reads/consumes messages
- Multiple processes consuming messages in one mail box
  - One message to all of them or
  - Each message to just one process
    - first reader,
    - select based on message type, …

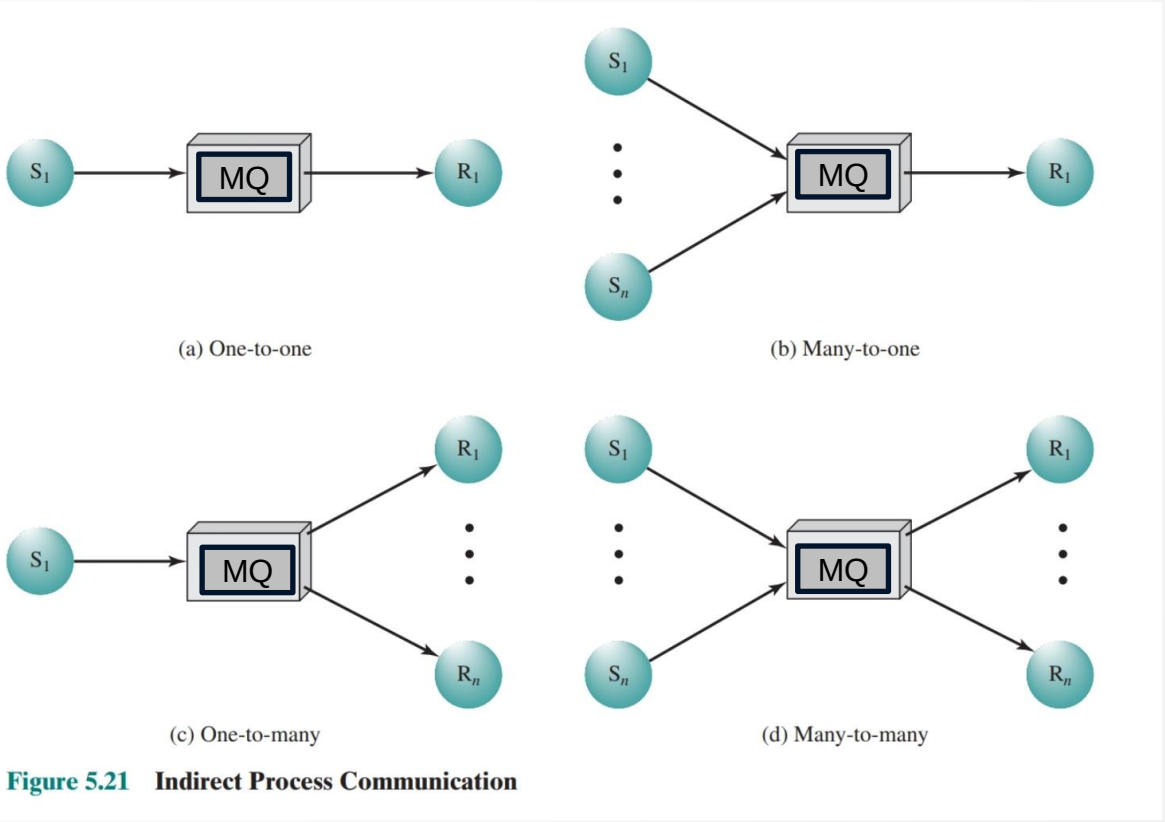


Figure 5.21 **Indirect Process Communication**

Stallings

REMOTE PROCEDURE CALL (RPC)/ REMOTE METHOD INVOCATION (RMI)

- Used to execute one procedure or method on a remote computer
- The caller process making the call is client
- The callee process executing the procedure is server

- Client needs to have a stub that
  - Can pack the argument to a message
  - Send the message to server
  - Wait for the reply
  - Unpack the result from message

- Server needs to have a stub that
  - Can wait for the incoming message
  - Unpack the arguments
  - Make the actual procedure call
  - Pack the result to message and send it

RPC

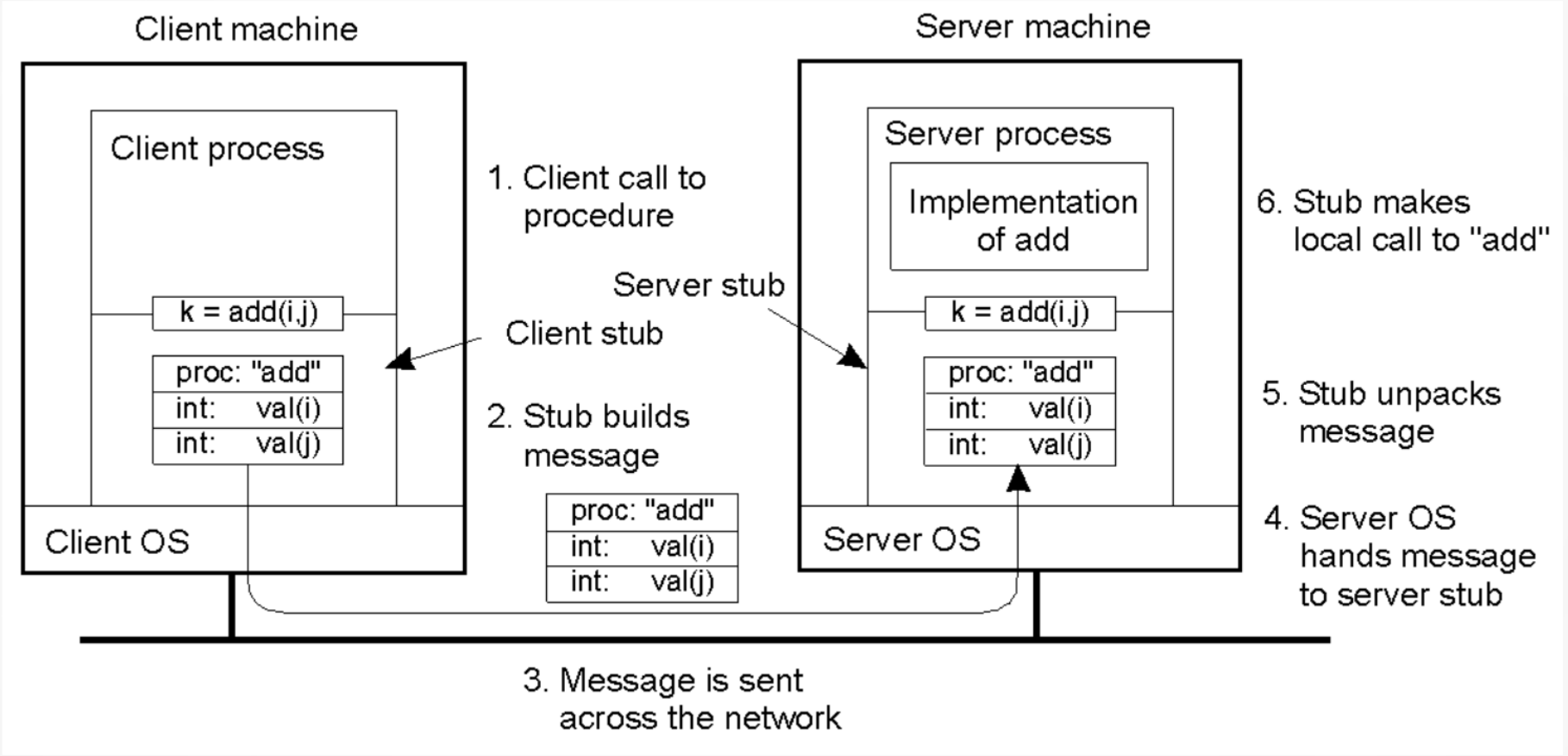


Figure from Tanenbaum and van Steen: Distributed Systems

DISCUSSION: FIRST PAIRS, THEN LARGER GROUPS

- Features

Synchronous
Asynchronous

Blocking
Non-blocking

and methods:

Shared-memory
Signals
Pipes
Files
Sockets
Message queues

- Explain the listed features and methods
- Which features describe which methods? Why / why not?

- If you have already used some of these, explain when and how
- If you have not used, think about situations when you might use each of them

MESSAGE PASSING VS SHARED MEMORY

- Shared memory
  - Used in one computer, but
  - Distributed Shared memory solutions have been studied

    https://en.wikipedia.org/wiki/Distributed_shared_memory

- Message passing
  - Must identify the recipient
  - Must define protocol including message formats
  - Value encodings and marshalling

MESSAGE PASSING: ADDRESSING

- Direct addressing
  - Sockets: (*IP address, port number*)
    - IP address identifies the destination computer (and source computer)
    - Local computer can always use IPv4 address 127.0.0.1 (loopback address)
    - Other computer's IP address must be known or searched (e.g. DNS, directory service, etc)
    - Port number identifies the socket (and receiving/sending process)
- Indirect addressing
  - Message queues are identified instead of processes
  - Messages are sent to MQ, not to the process
  - Access to MQ can be exclusive (just one process) or shared (multiple processes)

MIDDLEWARE (VÄLIOHJELMISTO)

- Wikipedia: Middleware is a type of computer software program that provides services to software applications beyond those available from the operating system.
- IETF (in 2000): services found above the transport (i.e. over TCP/IP) layer set of services but below the application environment
- One example: Object Request Brokers (ORB)

- Message Oriented Middleware (MOM)
  - Create uniform interface for application
  - Use message brokers that
  - store (buffer), route, and/or transform messages

See Wikipedia for more information:
- Message brokers
- Message Oriented Middleware

SECURITY ISSUES

- Who can access message content?
  - Only the sending and receiving processes?
  - Also message oriented middleware?
  - Also other message passing entities between sender and receiver?
- Use encryption and proper key management
  - Message protected, but
  - All marchalling between different mappings have to be done by sender and receiver
    - Data types and sizes, Big-Endian, Little-Endian, different character encoding, …
- What is the priority?

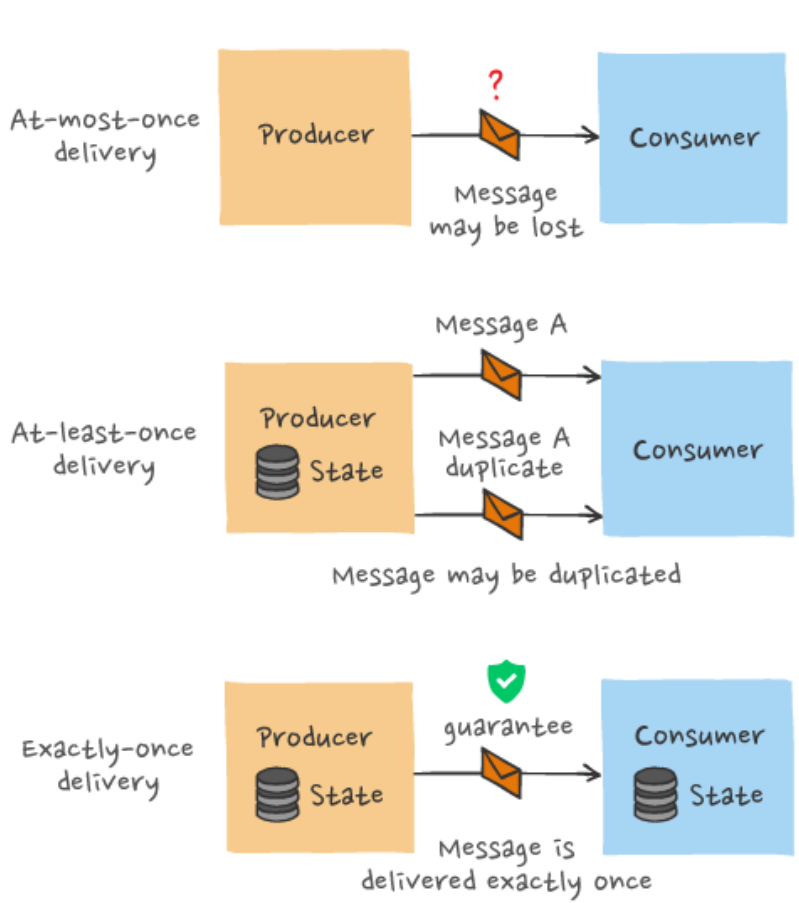
MESSAGE PASSING AND COMMUNICATION SEMANTICS

- Typical assumption: All messages transfered in the sending order
- However, a message in transit can
  - corrupt
  - be lost
  - dublicate
  - change order with other messages

- Typical solution: Detect and Retransmit
  - How many times the message is received?

MESSAGE DELIVERY SEMANTICS

- At-most-once semantics
  - A sent message might not be delivered
  - No message is delivered multiple times
- At-least-once semantics
  - A sent message is (eventually) delivered
  - A message might be delivered multiple times (duplicates allowed)
- Exactly-once semantics
  - Ideal target, but not always possible to reach



https://ably.com/blog/achieving-exactly-once-message-processing-with-ably

HOW TO SELECT?

- Based on communication style:
  - One-to-one, many-to-one, one-to-many, many-to-many
  - Synchronous vs asynchronous, blocking vs non-blocking, direct vs indirect
- Based on availability of methods
  - Not all OSs or programming languages provide all alternatives
- Based on number of computers
  - Single computer, multiple networked computers
- Based on efficiency / performance / security /semantics goals
- Based on …

SUITABLE READING

- Inter-process communication: https://en.wikipedia.org/wiki/Inter-process_communication
- Other pages linked from that
  - Message passing
  - Shared memory
  - Pipe, socket, message queue

- Stallings has just few pages and use slightly different terminology

TRANSIENT AND PERSISTENT MESSAGE PASSING

Transient

- Message in system only as long as both processes are executing
- Pipes and sockets are transient

Persistent

- Sent message is stored in the system as long as it takes to deliver it to the recipient
- Files and message queues are persistent

EXTRA DISCUSSION TOPICS

- Time overlap?
  - Processes must be running same time or can be running different times
- Speed of the mechanism
  - Which might be slowest? Why?
- How much data can be passed conveniently in one contact?
  - Differences? Can you order mechanisms based on the data amount?

Shared-memory
Signals
Pipes
Files
Sockets
Message queues