1495 UNIVERSITY OF ABERDEEN

JC4001 – Distributed Systems

Wrap-up

Dr Yongchao Huang (yongchao.huang@abdn.ac.uk)
Dr Shahzad Mumtaz (shahzad.mumtaz@abdn.ac.uk)
Dr Jari Korhonen (jari.korhonen@abdn.ac.uk)

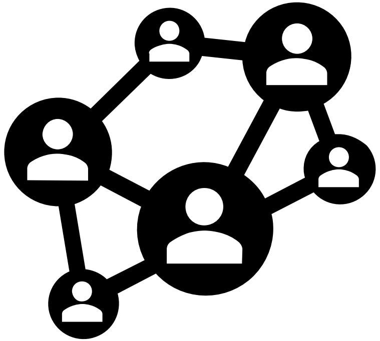
Oct-Dec 2025

Wrap-up: an overview

- This course covers the essential topics in distributed systems
  - *Distributed system* is a collection of independent computational entities (such as computers, servers, or devices) that are connected by a network and appear to the user as a single, coherent system
- Assessment based on in-class quizzes (20%), coursework assignment (30%), and exam (50%)
  - Coursework tests the ability to use distributed systems for a practical problem (i.e., federated learning)
  - Exam tests the general understanding of the course material

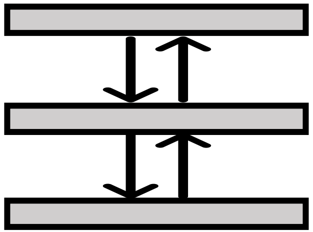
LC1: Introduction and fundamentals

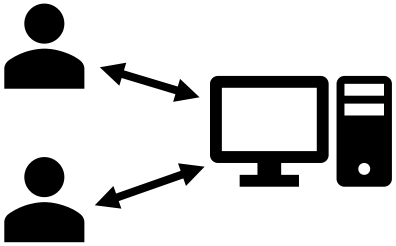
- Definitions and overall design goals of distributed systems
  - *Integrative view:* makes applications available for several users through networks of computers
  - *Expansive view:* expands existing systems to wider group of users and enhances usability for the existing users by adding computers
- Distributed system is often hidden from the users through middleware layer
  - Examples: web search engines, cloud computing

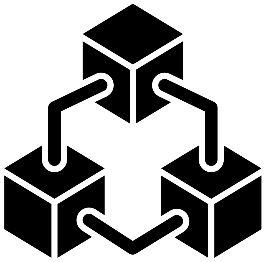


LC2: Architectures

- There are different architectural styles for distributed systems
  - *Layered system architectures* separate user interface, application, and low-level functionalities (like database handling) into layers
  - *Service-oriented architectures* are designed to build systems that provide services to other applications or consumers
  - In *peer-to-peer architectures*, each node can act as server and a client

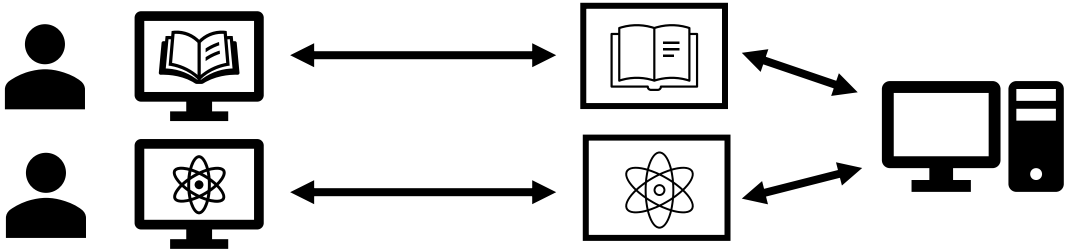






LC3: Processes

- *Process* is an instance of computer program that is being executed
  - In distributed systems, processes are the basic units of execution that run on different interconnected machines
- *Virtualisation* allows different users to share the same hardware resource in a distributed system
  - Different users can access the same computer and run their processes in virtual machines mimicking hardware/software interfaces

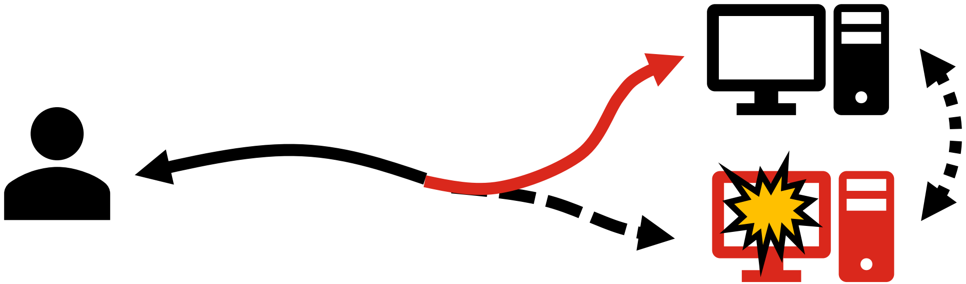


LC4: Replication and consistency

- *Replication* is the basic approach to improve availability of resources and fault tolerance
  - Same resource available from different servers
  - The challenge is to maintain *consistency* of data in different replicas: different clients should see the same data in each replica
- *Replica management* aims at finding the most appropriate locations for replicated resources
  - *Consistency protocols* aim at preserving sufficient level of consistency between the replicas

LC5: Fault tolerance

- *Fault tolerance* allows system to continue working in spite of faults
  - In general, fault tolerance is achieved through *redundancy*: faulty server or process is replaced by a redundant version of the server or process
  - Fault tolerance requires coordination between processes through consensus systems and protocols like Raft and Paxos



The exam

- The “small” questions (50%-60% of the exam):
  - The “small” questions can be multiple choice questions, true/false statements, or combining terms to explanations
  - The “small” questions cover all the lectures
  - Some questions are easy, some of them more difficult
- Open-ended questions (40%-50% of the exam):
  - The open-ended questions require broader knowledge of a certain topic and can include more complex problem-solving elements
  - Can include smaller questions concerning the same topic/problem

Example questions: MSC 1

- Which type of system has multiple central points?
  - A) Centralised system
  - B) De-centralised system
  - C) Distributed system
  - D) None of those above

Example questions: MSC 2

- How are two processes coupled, if they communicate through mailbox coordination?
  A) Temporally and referentially coupled
  B) Temporally and referentially decoupled
  C) Temporally coupled, referentially decoupled
  D) Temporally decoupled, referentially coupled

Example questions: MSC 3

- Which of the following could be a reason to migrate code instead of data?
  A) To keep local data private
  B) To minimise communication
  C) To share processing workload fairly
  D) All of those above

Example questions: MSC 4

- What is the difference between notations $W_1(x_1;x_2)$ and $W_1(x_1|x_2)$?
  A) In $W_1(x_1;x_2)$, x_2 is sequentially dependent on x_1
  B) In $W_1(x_1;x_2)$, x_1 and x_2 are produced concurrently
  C) In $W_1(x_1;x_2)$, x_1 are x_2 written in different locations, in $W_1(x_1|x_2)$ they are in the same locations
  D) In $W_1(x_1|x_2)$, x_1 are x_2 written in different locations, in $W_1(x_1;x_2)$ they are in the same locations

Example questions: MSC 5

- If the leader server crashes, which server becomes the new leader in Raft protocol?
  A) Leader chosen randomly among the other servers
  B) The server chosen earlier as the secondary leader
  C) The server in the most central location
  D) The server with the most committed operations

Example questions: True/False

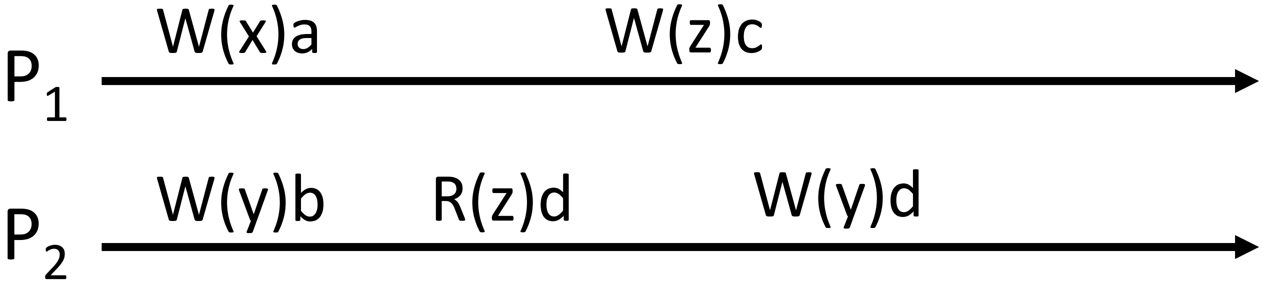
- Stateless servers do not keep information about their clients' states.
- Nonexclusive access gives a single process right to write but not read the data.
- When requests for a file F drop below deletion threshold, the server will always delete the file.
- Timing failures and crash failures can be confused.

Example questions: Combinations

- Connect terms A-D to correct locations 1-4: A) asymmetric; B) coordinator; C) layered; D) response
  - Value failure and state-transition failure are types of _1_ failure.
  - _2_ cryptosystems use a public key and a private key.
  - Network File System (NFS) is an example of _3_ system architecture.
  - Replicated-object invocations are avoided by using _4_.

Example question: Open-ended 1

- The diagram below shows the sequence of read (R) and write (W) operations by processes P1 and P2. Answer the questions:
  - Are operations in the diagram causally consistent? Why?
  - Which operations are causally related?
  - What values are possible for x, y, and z in the end?



P_1 W(x)a W(z)c

P_2 W(y)b R(z)d W(y)d

Example question: Open-ended 2

- In a 3-dimensional hypercube, what is the maximum distance between two nodes?
- In a 4-dimensional hypercube, what/which would be the shortest route(s) from node 11 (1011) to node 2 (0010)?
- In an *n*-dimensional hypercube, each data item is stored in one of the 2^n nodes, each connected to *n* other nodes. What are the benefits of this arrangement in comparison with unstructured P2P systems?

Example question: Open-ended 3

- Let us assume that you are designing a messaging system where the registered users can leave messages to the other registered users of the system.
  - How would you design the system so that it would remain usable even in case of a single server failure?
  - What kind of protocol you would use to make sure that messages don’t get lost in case of a sudden server failure?
  - What kind of compromises you may face in terms of reliability and latency?

That’s all, folks. Questions?