VIA University College

Software Development of Distributed Systems

Autumn 2022

System Models

Learning Objectives

- ✓explain the different kinds of **communicating entities** and **communication paradigms**.
- ✓give an overview of different distributed system **architectures.**
- ✓explain the **n-tier architecture**.

Three Basic ways to Describe Distributed Systems

- **Physical models** – view distributed systems in terms of hardware – computers and devices that constitute a system and their interconnectivity, without details of specific technologies
- **Architectural models** – describe a system in terms of the computational and communication tasks performed by its computational elements. Client-server and peer-to-peer most commonly used
- **Fundamental models** – take an abstract perspective in order to describe solutions to individual issues faced by most distributed systems
  - interaction models
  - failure models
  - security models

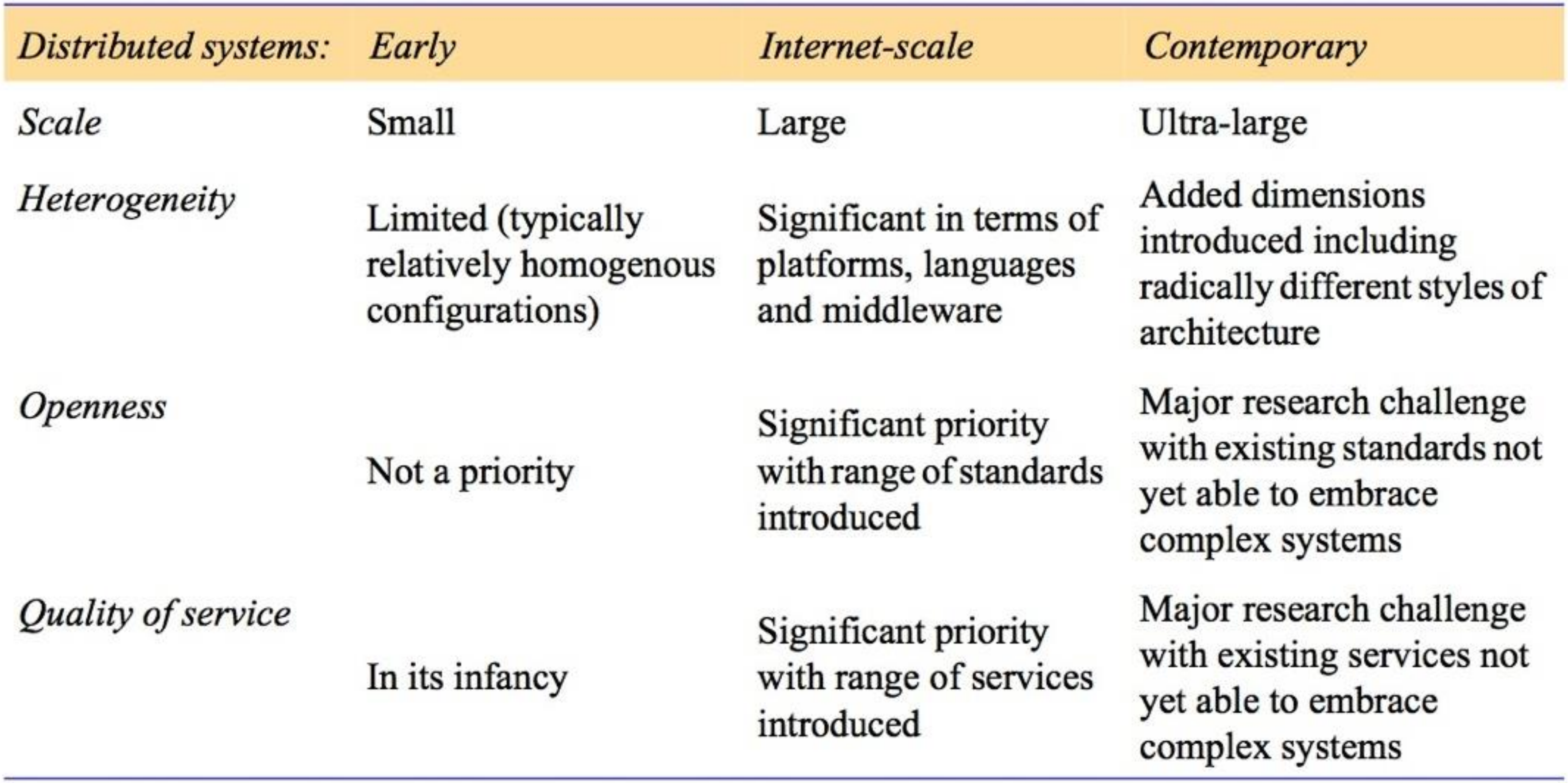
Recall: Difficulties and threats for Distributed Systems

- Widely varying modes of use
- Wide range of system environments
- Internal problems
- External threats

Physical Models

- Baseline physical model – minimal physical model of a distributed system as an extensible set of computer nodes interconnected by a computer network for the required passing of messages.

Generations of Distributed Systems



Distributed systems:	*Early*	*Internet-scale*	*Contemporary*
Scale	Small	Large	Ultra-large
Heterogeneity	Limited (typically relatively homogenous configurations)	Significant in terms of platforms, languages and middleware	Added dimensions introduced including radically different styles of architecture
Openness	Not a priority	Significant priority with range of standards introduced	Major research challenge with existing standards not yet able to embrace complex systems
Quality of service	In its infancy	Significant priority with range of services introduced	Major research challenge with existing services not yet able to embrace complex systems

Architectural Models

- Main concerns: make the system reliable, manageable, adaptable and cost effective.
- Architectural elements:
  - What are the entities that are communicating in the distributed system?
  - How do they communicate, or, more specifically, what communication paradigm is used?
  - What (potentially changing) roles and responsibilities do they have in the overall architecture?
  - How are they mapped on to the physical distributed infrastructure (what is their placement)?

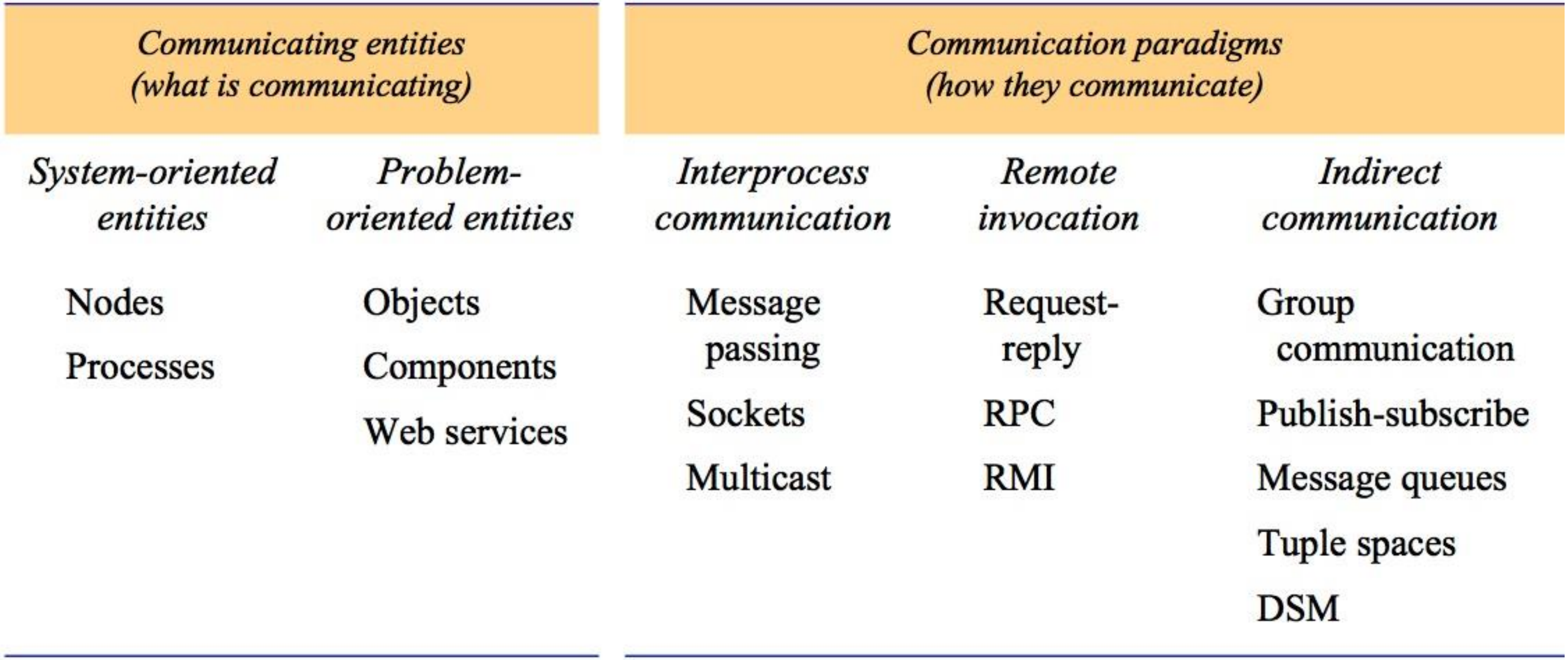
Communicating Entities

- From system perspective: processes
    - in some cases:
        - nodes (sensors)
        - threads (endpoints of communication)
- From a programming perspective:
    - Objects
        - computation consists of a number of interacting objects representing natural units of decomposition for the given problem domain
        - Objects are accessed via interfaces, with an associated interface definition language (or IDL)
    - Components
        - offer problem-oriented abstractions for building distributed systems
        - accessed through interfaces
    - Web services
        - closely related to objects and components
        - intrinsically integrated into the World Wide Web
            - · using web standards to represent and discover services

Communication Paradigms

- Interprocess communication: low-level support for communication between processes in distributed systems, including message-passing primitives, direct access to the API offered by Internet protocols (socket programming) and support for multicast communication
- Remote Invocation: calling of a remote operation, procedure or method
  - Remote procedure call (RPC)
  - Remote method invocation (RMI)
- Indirect communication:
  - Senders do not need to know who they are sending to (space uncoupling)
  - Senders and receivers do not need to exist at the same time (time uncoupling)
  - Techniques: Group communication, Publish-subscribe systems

Communicating entities and Communication paradigms

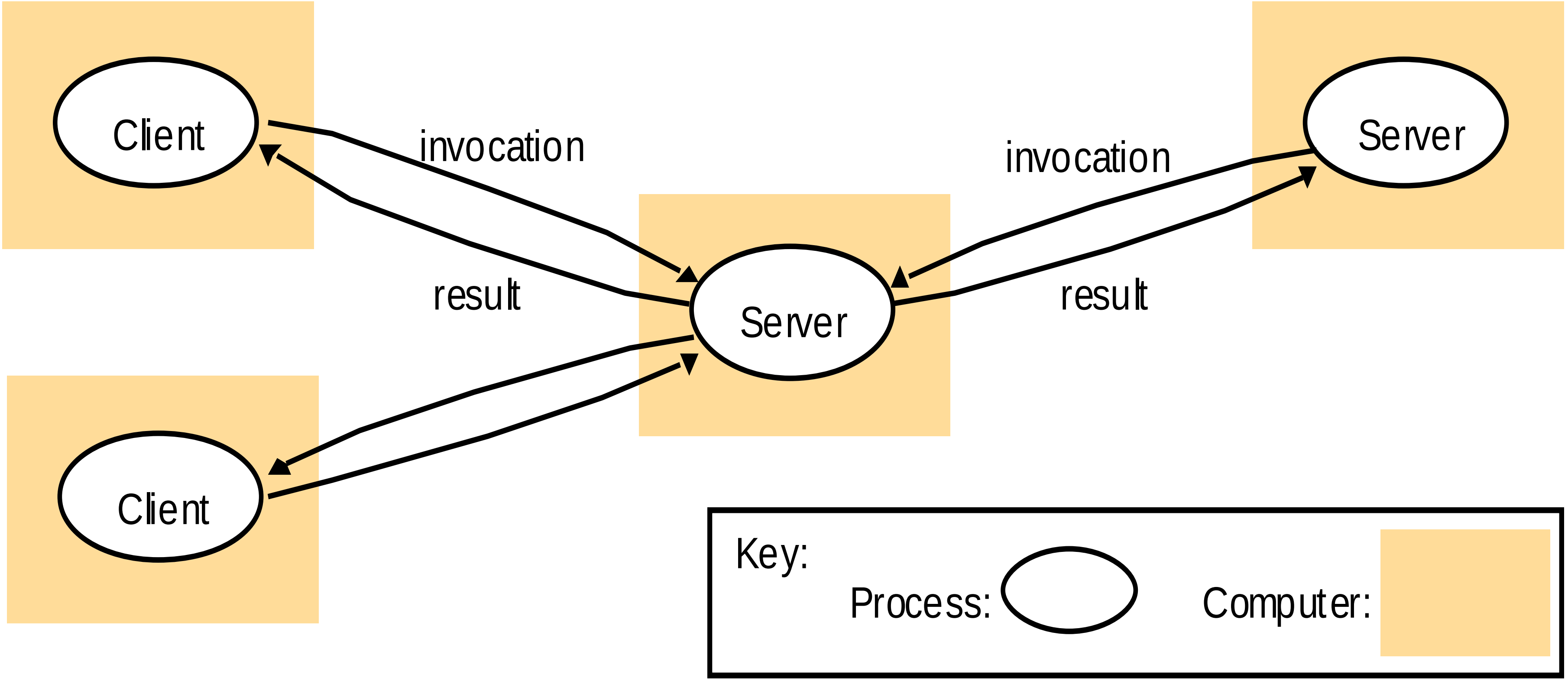


Communicating entities (what is communicating)		*Communication paradigms (how they communicate)*		
System-oriented entities	*Problem-oriented entities*	*Interprocess communication*	*Remote invocation*	*Indirect communication*
Nodes	Objects	Message passing	Request-reply	Group communication
Processes	Components	Sockets	RPC	Publish-subscribe
	Web services	Multicast	RMI	Message queues
				Tuple spaces
				DSM

Architectural Styles - Examples

- Client/Server
- Peer-To-Peer
- Model/View/Controller
- Three-tier, N-Tier Architecture
- Service-Oriented Architecture (SOA)
- Microservices
- Cloud
- Serverless, etc.

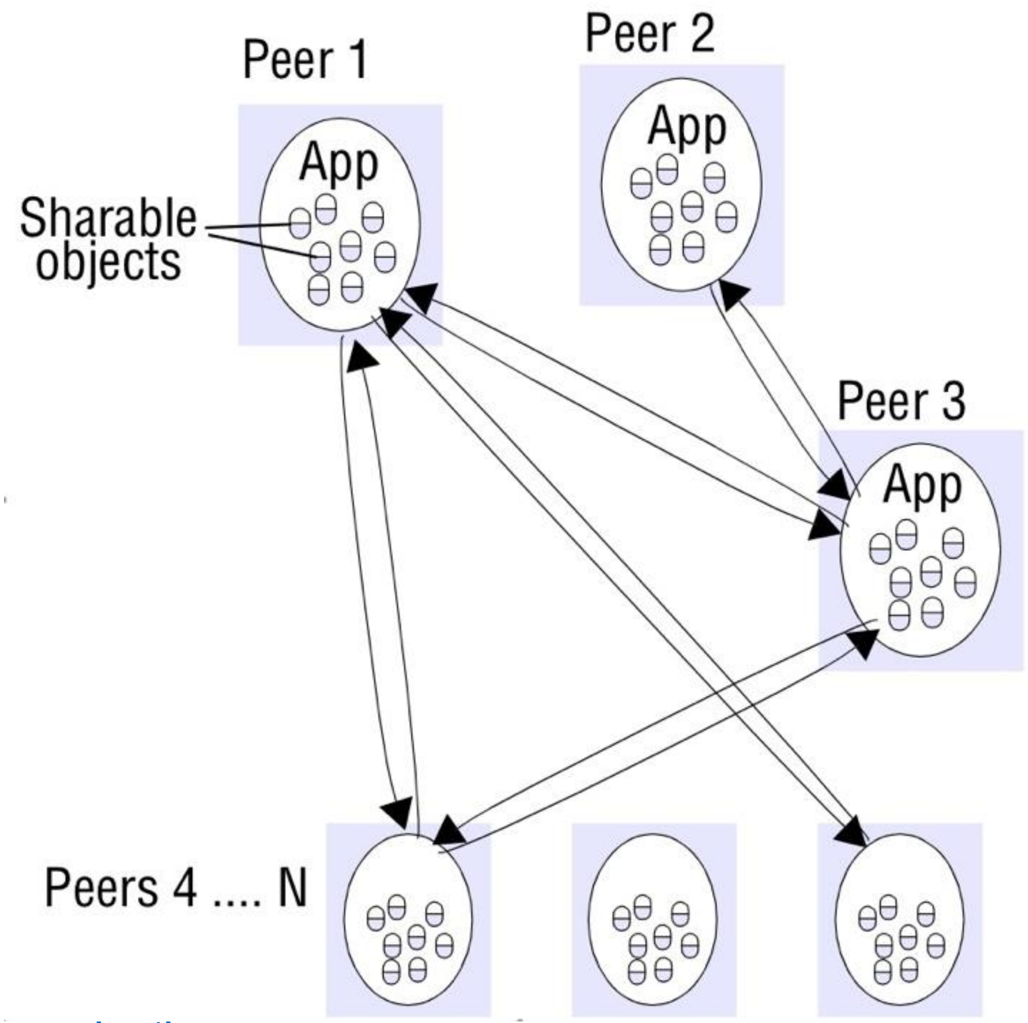
Roles and responsibilities: Client-server



Clients invoke individual servers

Roles and responsibilities: Peer-to-peer

Peer-to-peer architecture



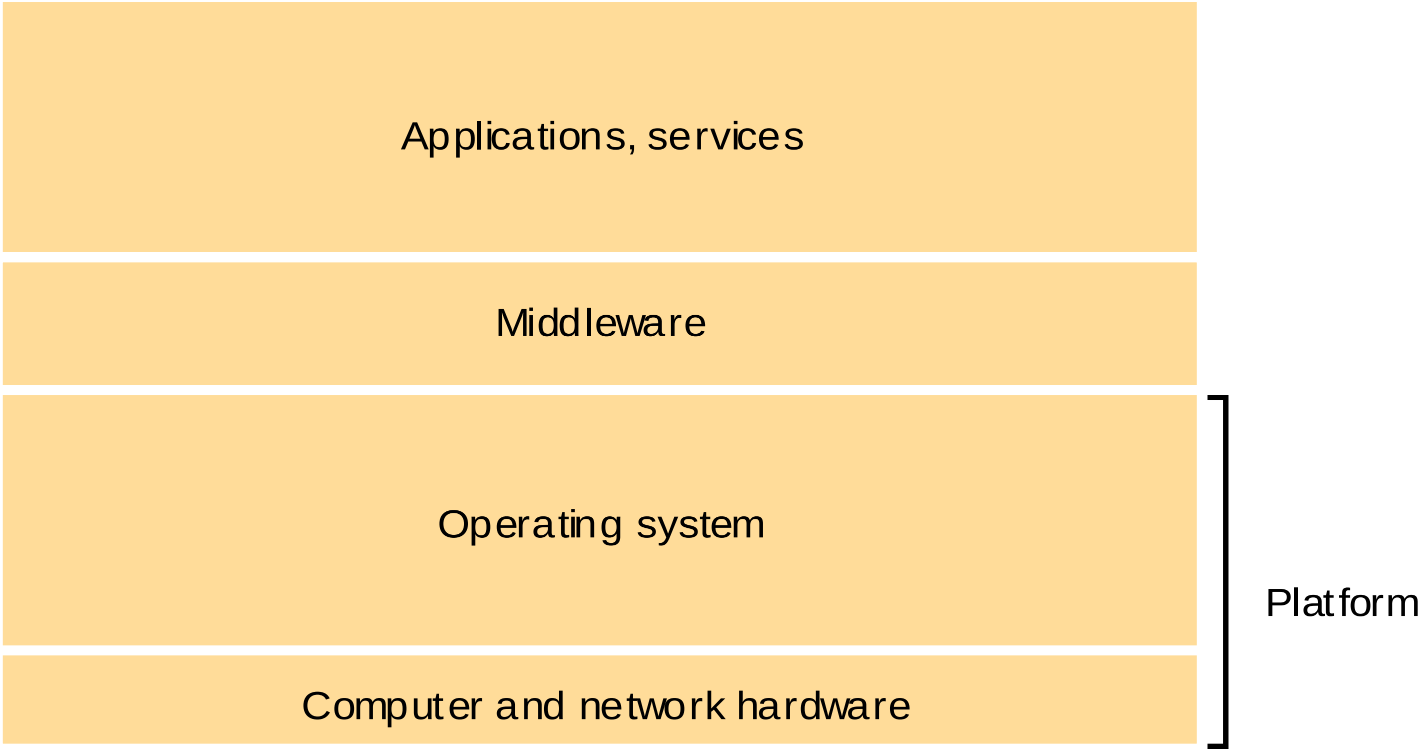
* same set of interfaces to each other

Architectural patterns - Layering

- Layered approach – complex system partitioned into a number of layers:
  - vertical organisation of services
  - given layer making use of the services offered by the layer below
  - software abstraction
  - higher layers unaware of implementation details, or any other layers beneath them

Platform and Middleware

Software and hardware service layers in distributed systems



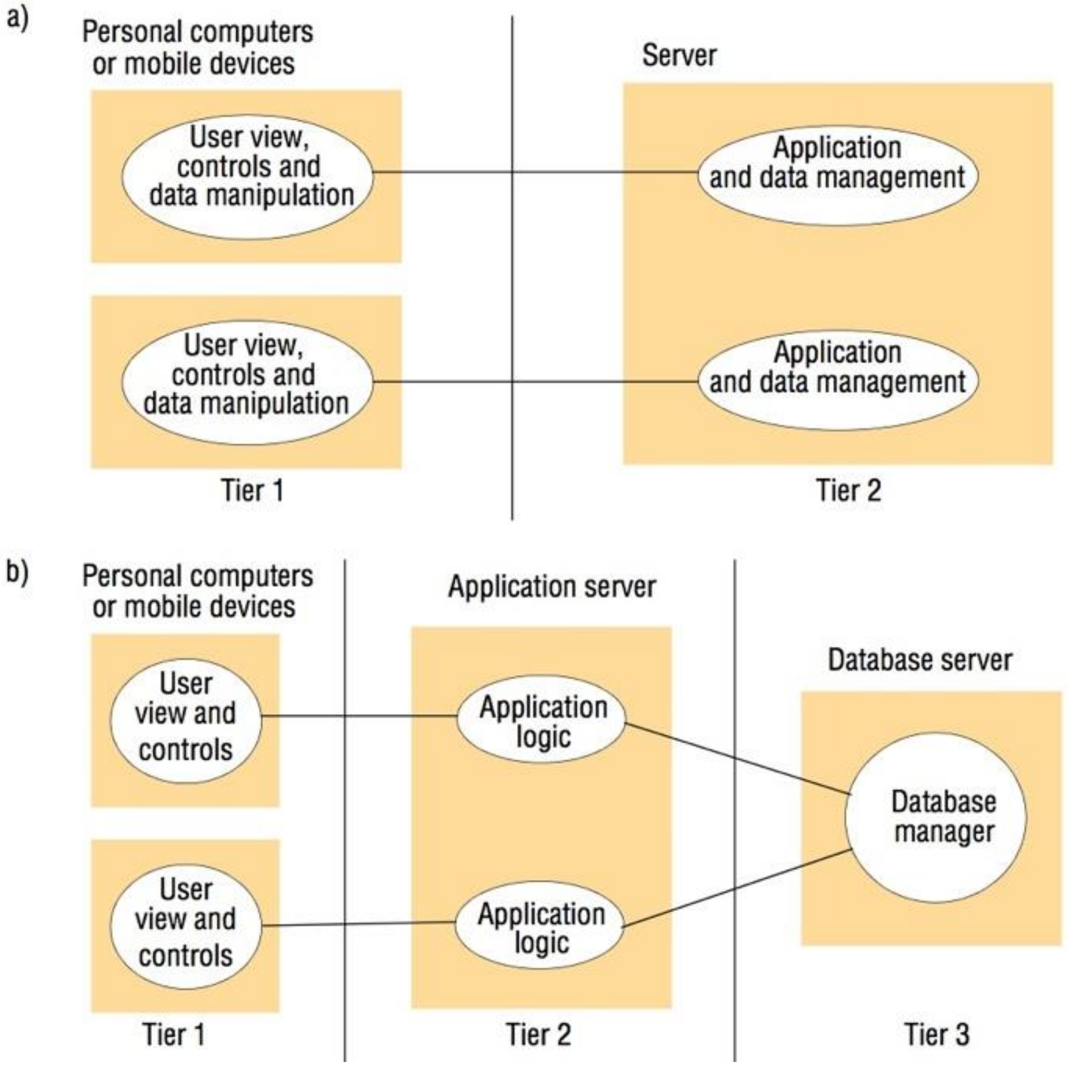
- A platform for distributed systems and applications consists of the lowest-level hardware and software layers.
- Middleware – a layer of software whose purpose is to mask heterogeneity and to provide a convenient programming model to application programmers.

Tiered Architecture

- Tiering is a technique to organize functionality of a given layer and place this functionality into appropriate servers and, as a secondary consideration, on to physical nodes.

- For instance functional decomposition of an application into two-tier and three-tier architecture:
  - presentation tier
  - Application/business logic tier
  - data tier

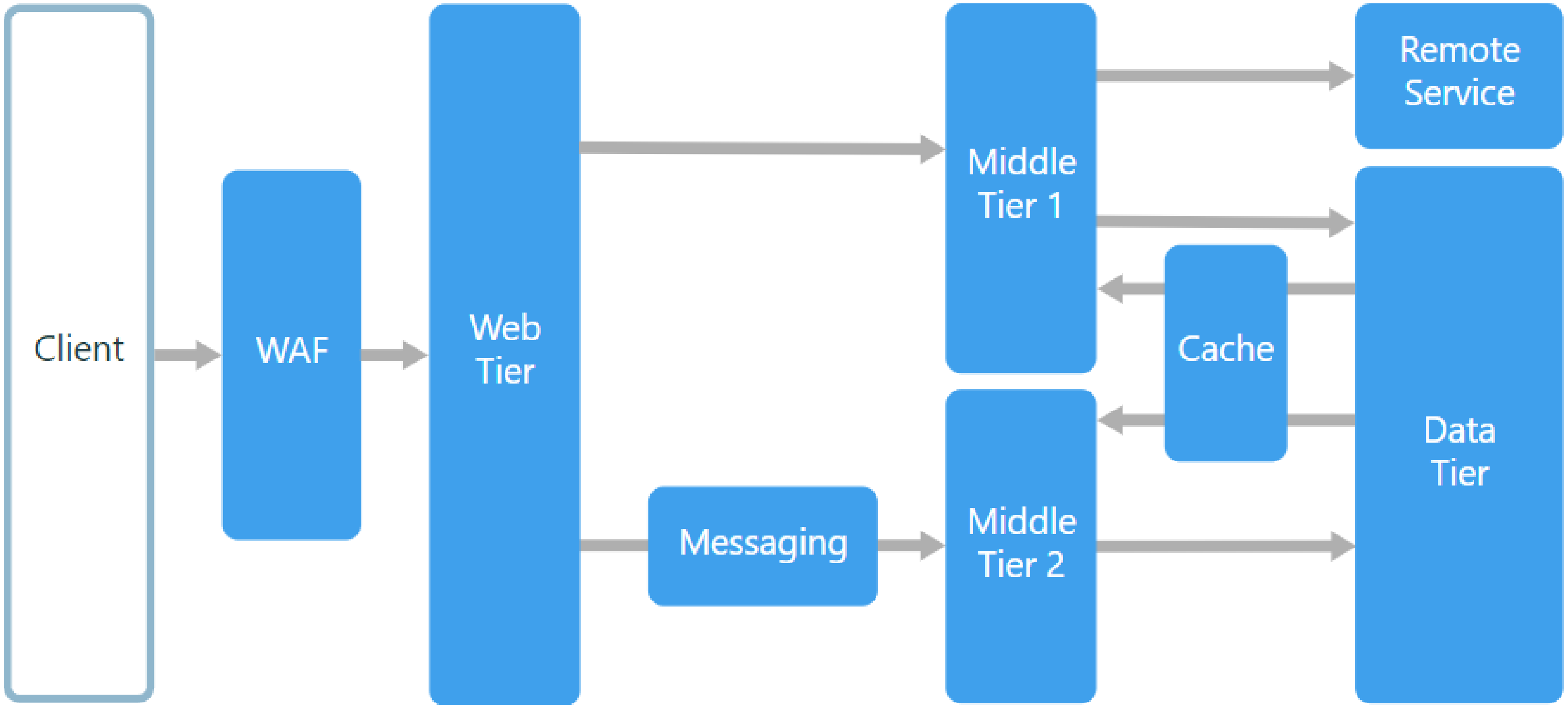
Two-tier and Three-tier Architectures


a)
Personal computers or mobile devices
User view, controls and data manipulation
User view, controls and data manipulation
Tier 1
Server
Application and data management
Application and data management
Tier 2
b)
Personal computers or mobile devices
User view and controls
User view and controls
Tier 1
Application server
Application logic
Application logic
Tier 2
Database server
Database manager
Tier 3

- three aspects partitioned into two processes
- (+) low latency
- (-) splitting application logic

- (+) one-to-one mapping from logical elements to physical servers
- (-) added complexity, network traffic and latency

N-tier Architecture



- divides an application into:
  - Logical layers
  - Physical tiers

https://docs.microsoft.com/en-us/azure/architecture/guide/architecture-styles/n-tier#:~:text=An%20N%2Dtier%20architecture%20divides,layer%20has%20a%20specific%20responsibility.&text=A%20traditional%20three%2Dtier%20application,tier%2C%20and%20a%20database%20tier.

Group Discussion

- Discuss two key points from the lesson so far, that could be useful in SEP3
- Discuss ideas for your architecture design