CSE 601: Distributed Systems

Toukir Ahammed

System Architectures

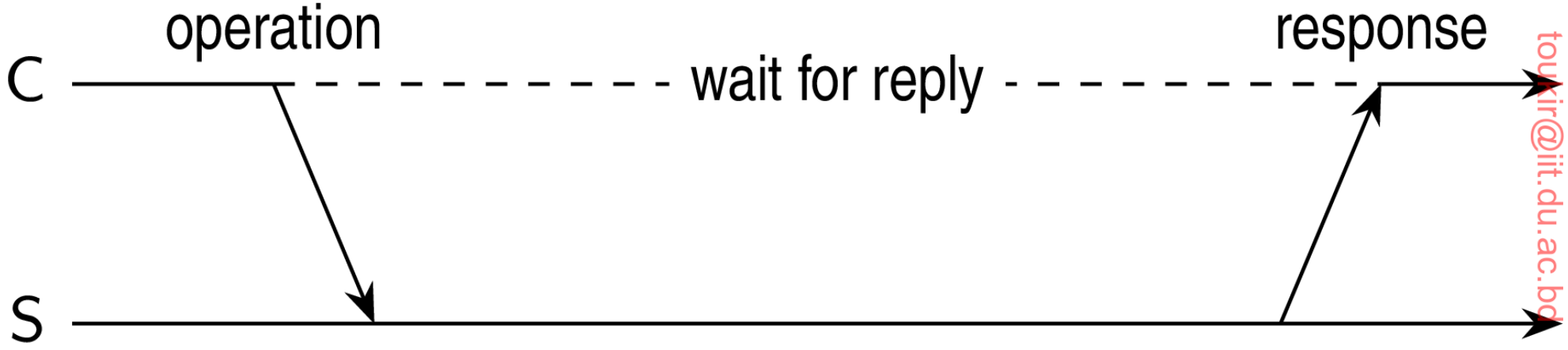
- Let us now take a look at how distributed systems are actually organized by considering where software components are placed.
- Deciding on software components, their interaction, and their placement leads to an instance of a software architecture, also known as a **system architecture**
- Thinking in terms of clients that request services from servers helps to understand and manage the complexity of distributed systems

Simple Client-server Architecture

- A **server** is a process implementing a specific service, for example, a file system service or a database service
- A **client** is a process that requests a service from a server by sending it a request and subsequently waiting for the server's reply.
- This client-server interaction is also known as request-reply behavior

Simple Client-server Architecture

- When a client requests a service, it simply packages a message for the server, identifying the service it wants, along with the necessary input data.
- The message is then sent to the server which will always wait for an incoming request subsequently process it, and package the results in a reply message that is then sent to the client.



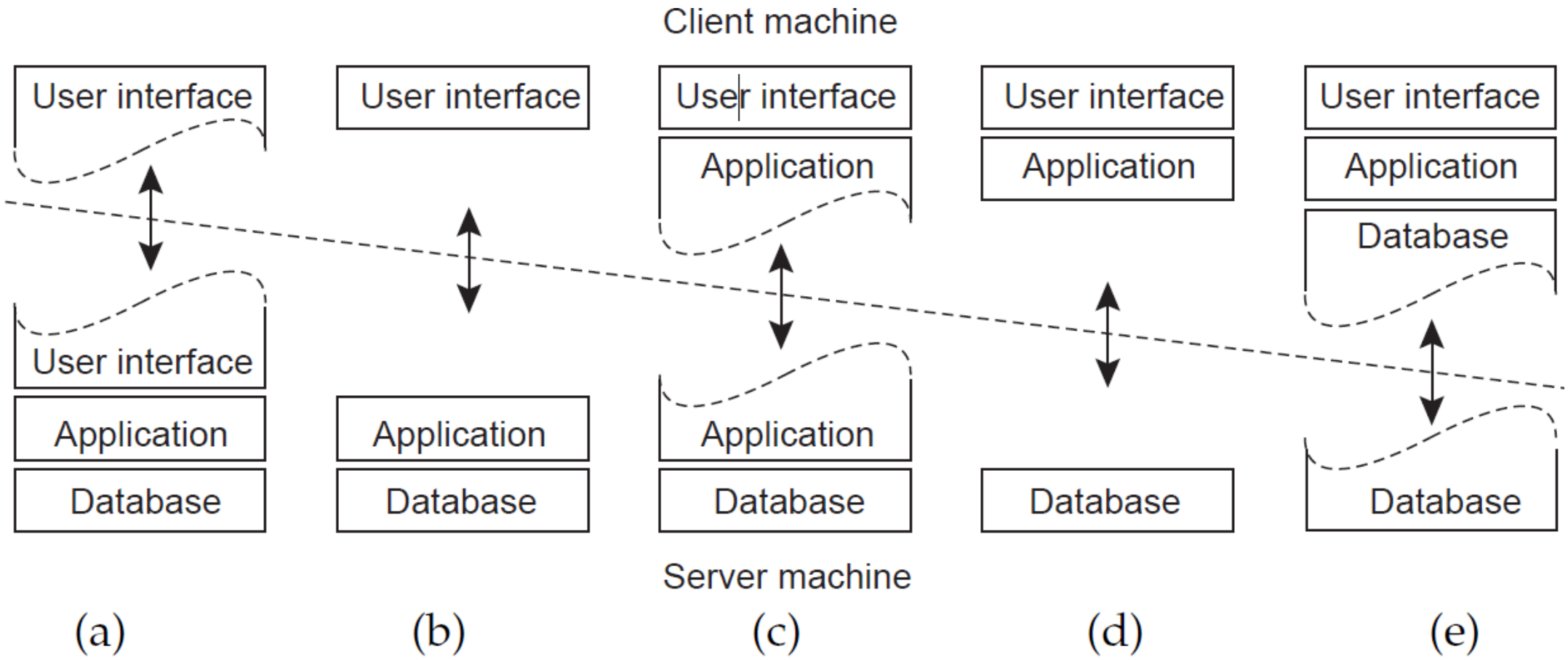
Transmission Failures

- The problem is that the client cannot detect whether the original request message was lost, or that transmission of the reply failed.
- If the reply was lost, then resending a request may result in performing the operation twice
  - If the operation was something like "*transfer $10,000 from my bank account*," then clearly, it would have been better that we simply reported an error instead
  - If the operation was "*tell me how much money I have left*," it would be perfectly acceptable to resend the request.
- When an operation can be repeated multiple times without harm, it is said to be idempotent.

Multitiered Architectures

- The distinction into three logical levels, (1) a user-interface layer, (2) a processing layer, and (3) a data layer, suggests several possibilities for physically distributing a client-server application across several machines.
- The simplest organization is to have only two types of machines:
  - A client machine containing only the programs implementing (part of) the user-interface level
  - A server machine containing the rest, that is, the programs implementing the processing and data level

Two-tiered Architectures



Two-tiered Architectures

a) Only the terminal-dependent part of the user interface on the client machine, and give the applications remote control over the presentation of their data

b) Place the entire user-interface software on the client side
   - divide the application into a graphical front end, which communicates with the rest of the application (residing at the server) through an application-specific protocol.
   - the front end (the client software) does no processing other than necessary for presenting the application's interface.

Two-tiered Architectures

c) Move part of the application to the front end

- For example, the application makes use of a form that needs to be filled in entirely before it can be processed
- The front end can then check the correctness and consistency of the form, and where necessary interact with the user.
- A word processor in which the basic editing functions execute on the client side where they operate on locally cached, or in-memory data, but where the advanced support tools such as checking the spelling and grammar execute on the server side.

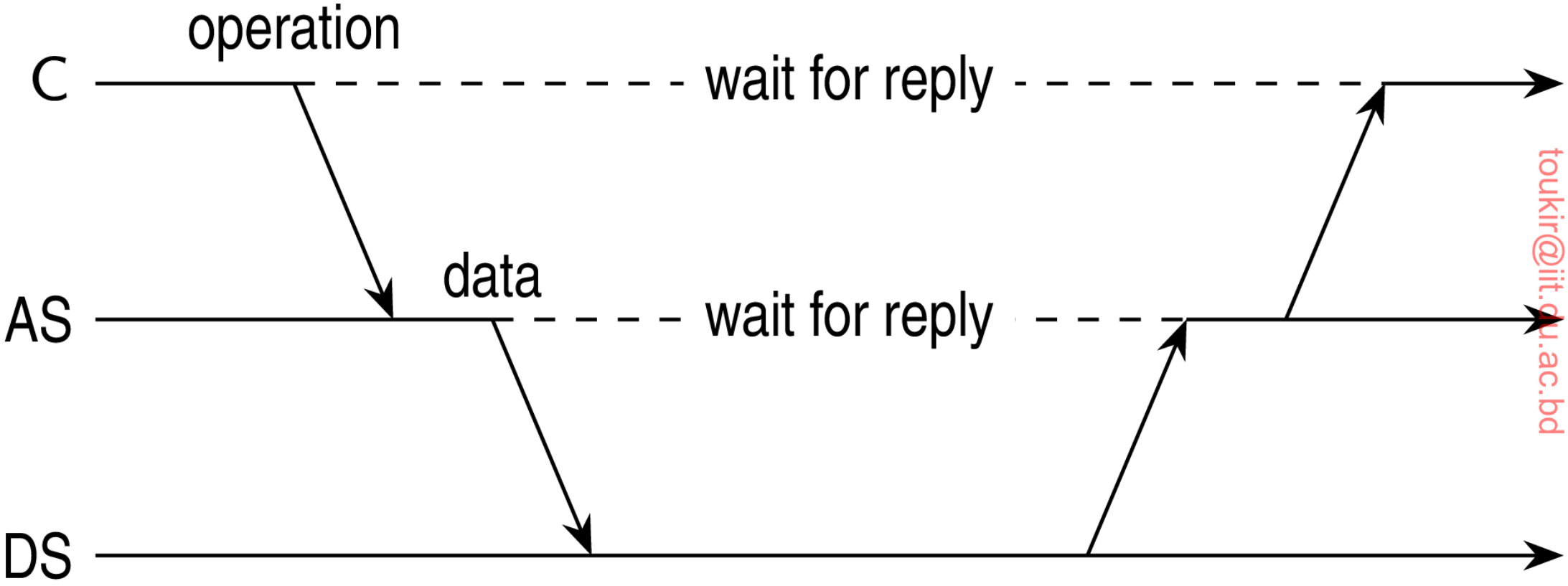
Two-tiered Architectures

d) These organizations are used where the client machine is a PC or workstation, connected through a network to a distributed file system or database.
   - Banking applications run on an end-user's machine, where the user prepares transactions and such. Once finished, the application contacts the database on the bank's server and uploads the transactions for further processing

e) Represents the situation where the client's local disk contains part of the data.
   - For example, when browsing the Web, a client can gradually build a huge cache on local disk of most recent inspected Web pages

Three-tiered Architecture

Server may sometimes need to act as a client



Vertical vs Horizontal Distribution

- In many business environments, distributed processing is equivalent to organizing a client-server application as a multitiered architecture. This type of distribution is referred as **vertical distribution**. It is achieved by placing logically different components on different machines
- In **horizontal distribution**, a client or server may be physically split up into logically equivalent parts, but each part is operating on its own share of the complete data set, thus balancing the load

Peer-to-Peer (P2P) Architecture

- The processes that constitute a peer-to-peer system are all equal
- This means that the functions that need to be carried out are represented by every process that constitutes the distributed system
- Each process will act as a client and a server at the same time

Cloud Computing

- Organizations in charge of running data centers have been seeking ways for opening up their resources to customers.
- This led to the concept of **cloud computing** by which a customer could upload tasks to a data center and be charged on a per-resource basis.
- Cloud computing is characterized by an easily usable and accessible pool of virtualized resources.
- Clouds are organized into four layers: Hardware, Infrastructure, Platform and Application

Hardware

- The lowest layer is formed by the means to manage the necessary hardware:
  - processors, routers, but also power and cooling systems.
- It is generally implemented at data centers and contains the resources that customers normally never get to see directly.

Infrastructure

- The infrastructure layer forms the backbone for most cloud computing platforms.
- It deploys virtualization techniques to provide customers an infrastructure consisting of virtual storage and computing resources.
- Indeed, nothing is what it seems: cloud computing evolves around allocating and managing virtual storage devices and virtual servers.

Platform

- The platform layer provides to a cloud computing customer what an operating system provides to application developers, namely the means to easily develop and deploy applications that need to run in a cloud.
- For example, the Amazon S3 storage system is offered to the application developer in the form of an API allowing (locally created) files to be organized and stored in buckets. By storing a file in a bucket, that file is automatically uploaded to the Amazon cloud.

Application

- Actual applications run in this layer and are offered to users for further customization.
- Well-known examples include those found in office suites (text processors, spreadsheet applications, presentation applications, and so on).
- It is important to realize that these applications are again executed in the vendor's cloud.
- As before, they can be compared to the traditional suite of applications that are shipped when installing an operating system.

Cloud-computing Services

Cloud-computing providers offer these layers to their customers through various interfaces (including command-line tools, programming interfaces, and Web interfaces), leading to three different types of services:

1. **Infrastructure-as-a-Service (IaaS)** covering the hardware and infrastructure layer.
2. **Platform-as-a-Service (PaaS)** covering the platform layer.
3. **Software-as-a-Service (SaaS)** in which their applications are covered.

Organization of Clouds