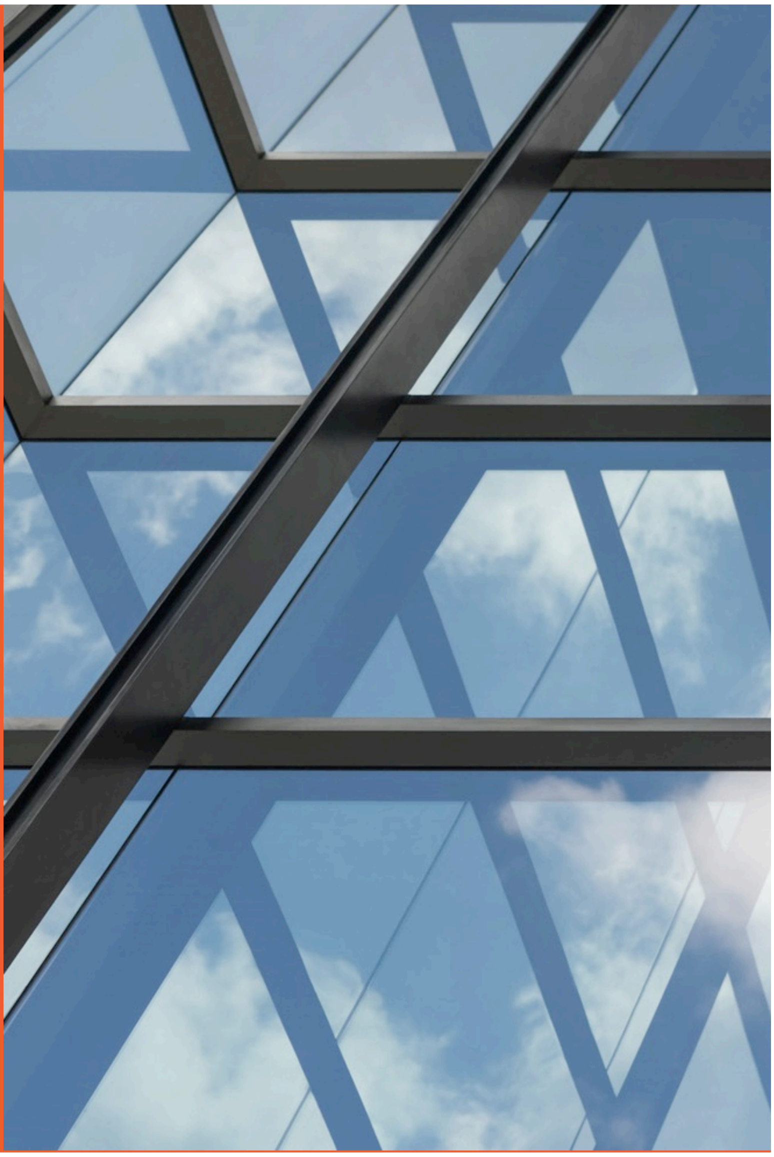
COMP3221: Distributed Systems

Architecture

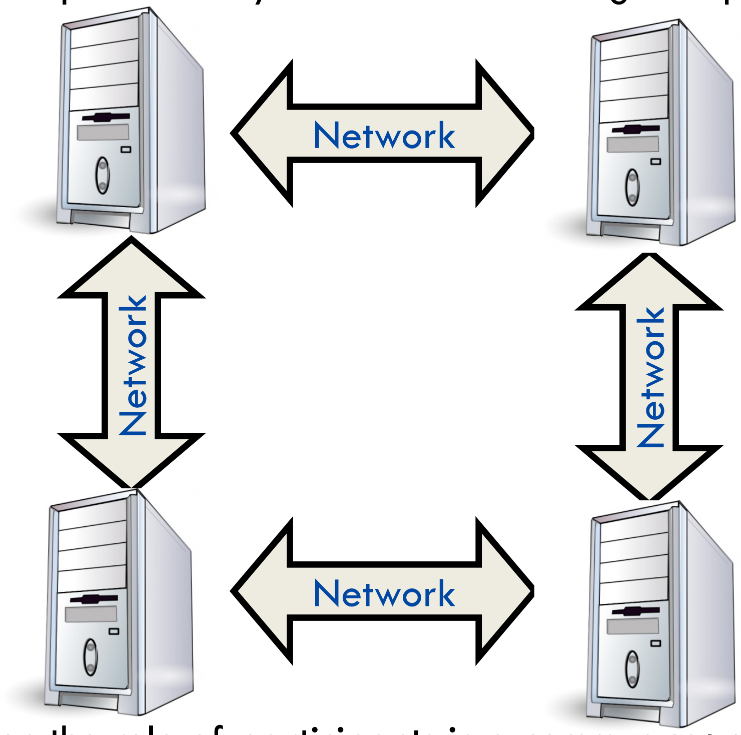
Unit coordinator Dr. Vincent Gramoli
Lecturer Dr. Guillaume Jourjon
School of Information Technologies

THE UNIVERSITY OF
SYDNEY



Introduction

Previously: diverse components may be involved in a single request



- Now: let's focus on the role of participants in a communication

Outline

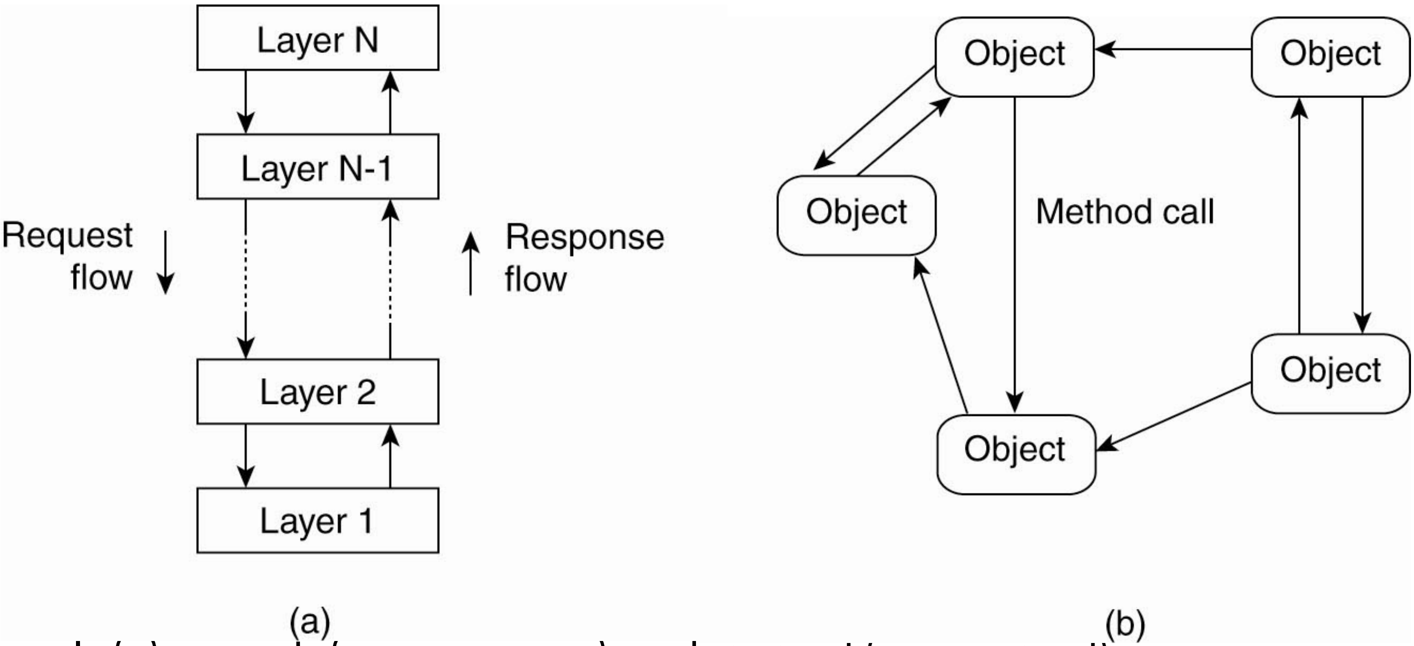
- Software Architecture
- The Client-Server Model
- The Layered Organization
- The Peer-to-Peer Organization
- Distributed Operating Systems

Software Architecture

THE UNIVERSITY OF SYDNEY

Component organization

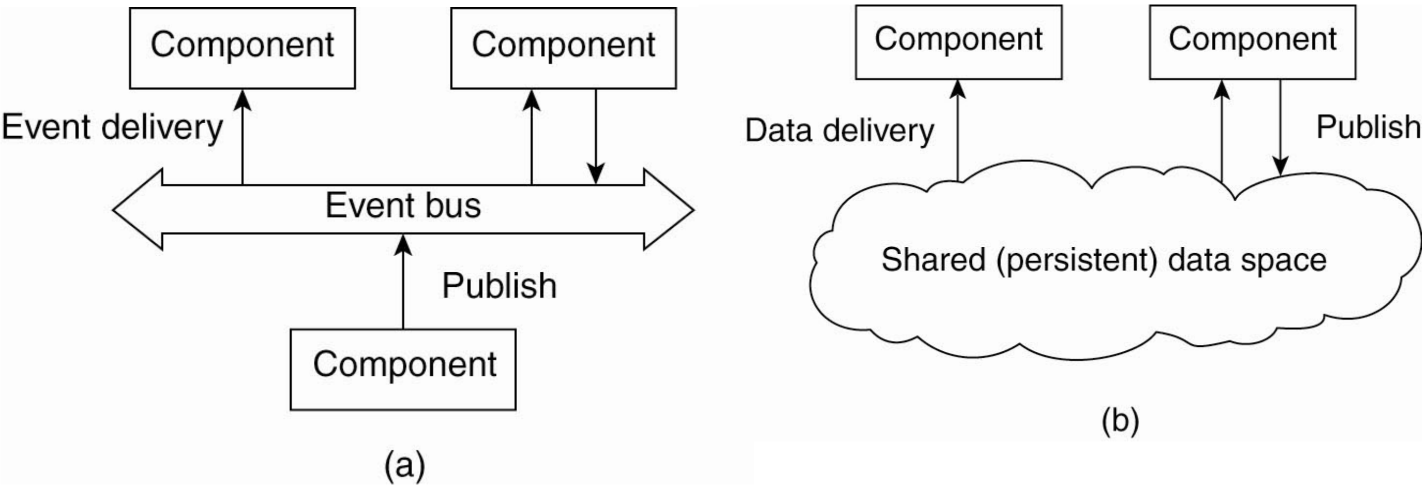
Layer-based architecture (a) vs. object-based architecture (b)



- In (a), requests (resp. responses) go downward (resp. upward)
- In (b), objects communicate through Remote Procedure Calls (RPCs)

Communication organization

Communication through events or shared repository



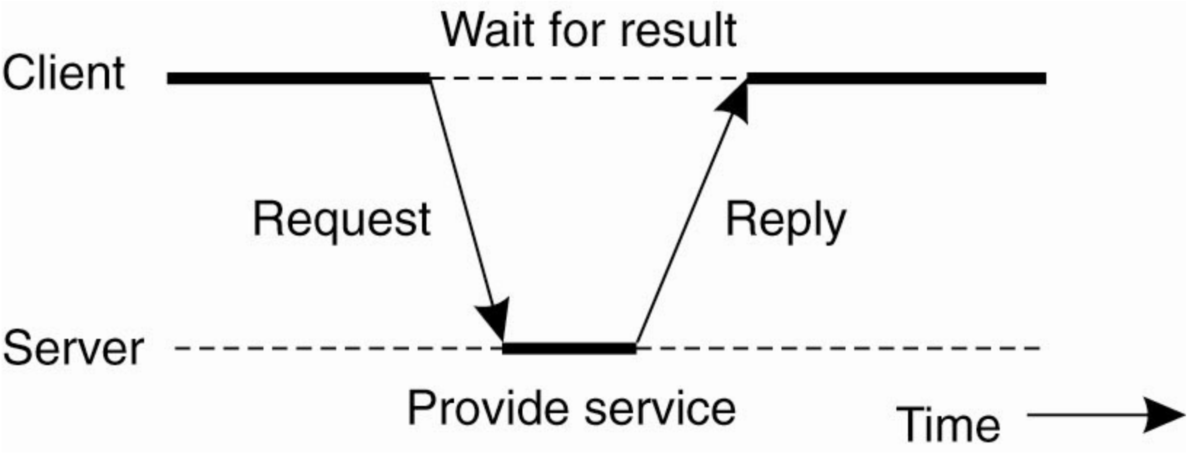
- (a) Event-based architecture: communication through events, that optionally carry data (subscribers get their desired events delivered)
- (b) Data-centered architecture: through a shared repository, that contains data (e.g., files in a distributed file system)

The Client-Server Model

The client server model

The basic client-server model

- The client requests the service whereas the server provides the service
- The client and the server can be hosted on different machines.

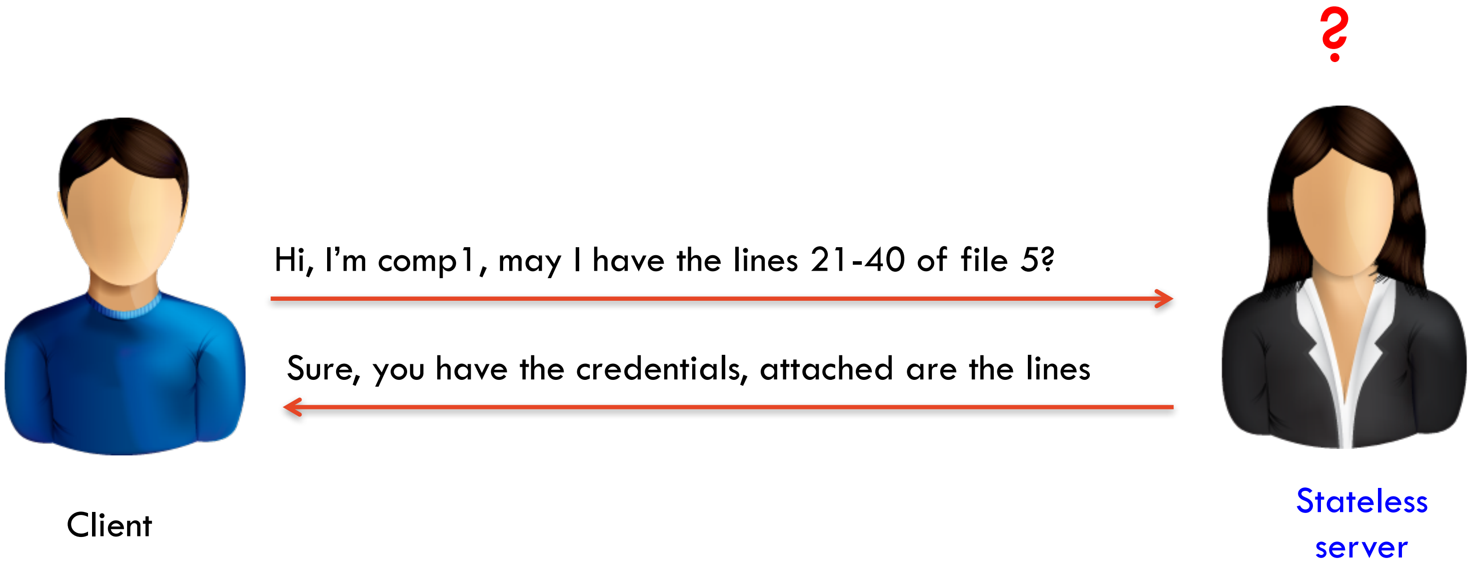


- The communication follows a request-reply model.

The client server model

Stateless vs stateful server

- Stateless server: does not record the state of its clients



?

Hi, I'm comp1, may I have the lines 21-40 of file 5?

Sure, you have the credentials, attached are the lines

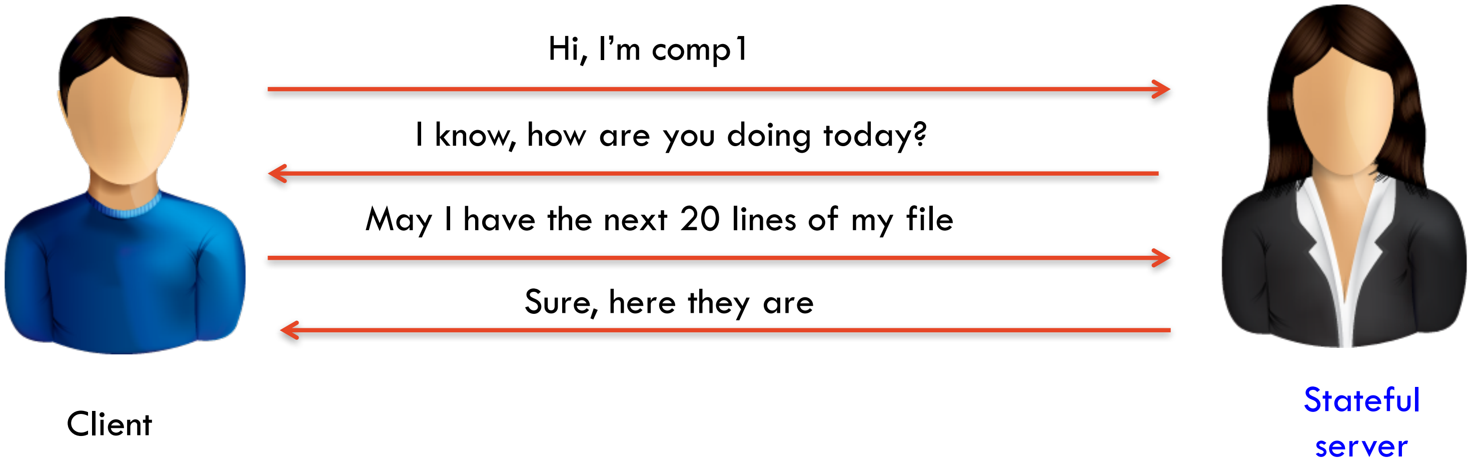
Client

Stateless server

The client server model

Stateless vs stateful server

- Stateless server: does not record the state of its clients
- Stateful server: maintains persistent information about its clients (client->file)



Hi, I'm comp1

I know, how are you doing today?

May I have the next 20 lines of my file

Sure, here they are

Client

Stateful server

The client server model

Stateless vs stateful server

- Stateless server: does not record the state of its clients
- Stateful server: maintains persistent information about its clients (client->file)

	Stateless server	Stateful server
State	No info kept	Persistent info
Request	Self-contained	Can be split, generally faster
Upon failure	No recovery needed	State recovery needed (explicit deletion)
Example	Network file system (NFSv3)	Andrew file system (AFS)

The Layered Organization

THE UNIVERSITY OF
SYDNEY

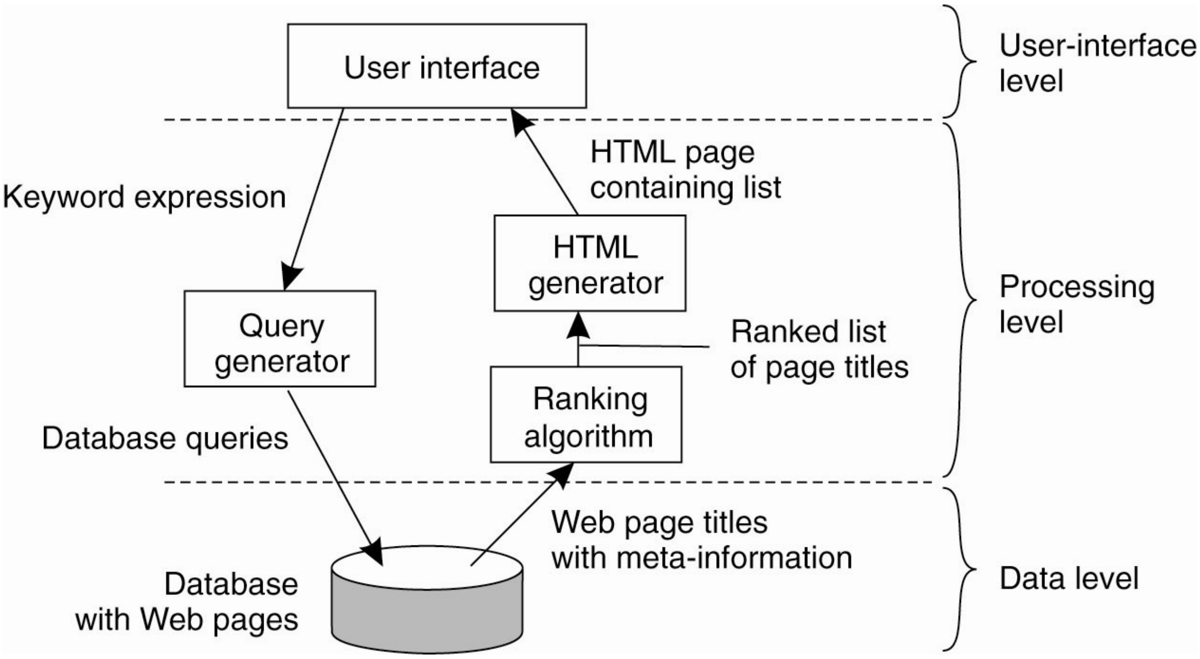
Application layering

Traditional three-layered view:

- The user interface layer: contains the feature to control the application
- The processing layer: contains the function of the application
- The data layer: contains the data of the application

Application layering (cont'd)

Example: a search engine request spanning the traditional three layers



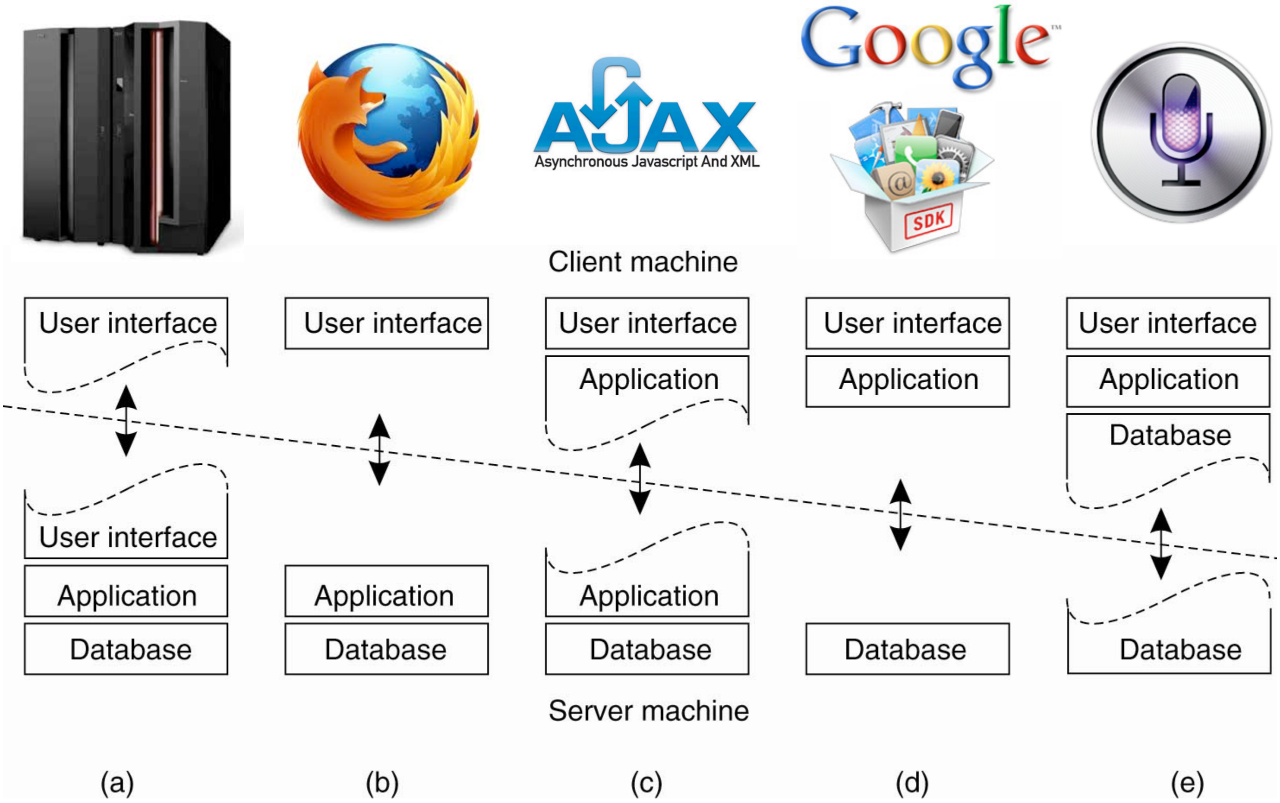
Application layering (cont'd)

Hosting different layers on different machines

- Three-tiered architecture:
  - each layer on separate machine
- Two-tiered architecture:
  - client
  - single server configuration
- Single-tiered architecture:
  - dumb terminal
  - mainframe configuration

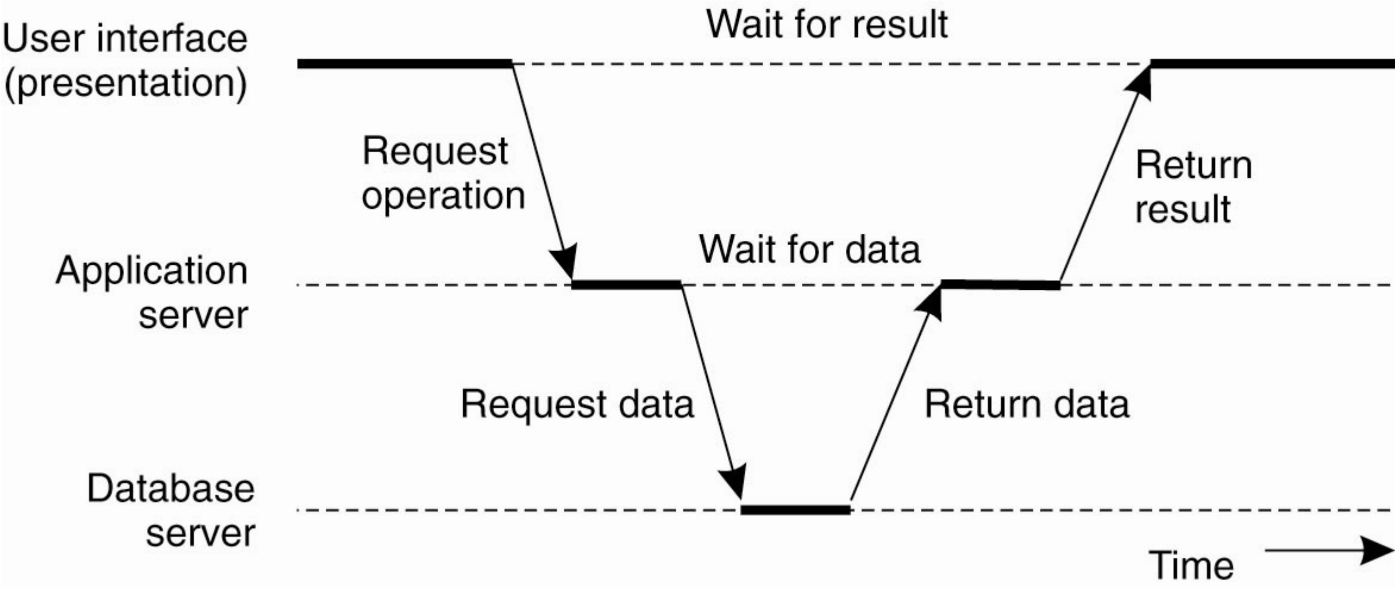
Multi-tiered architectures

Physical two-tiered architecture



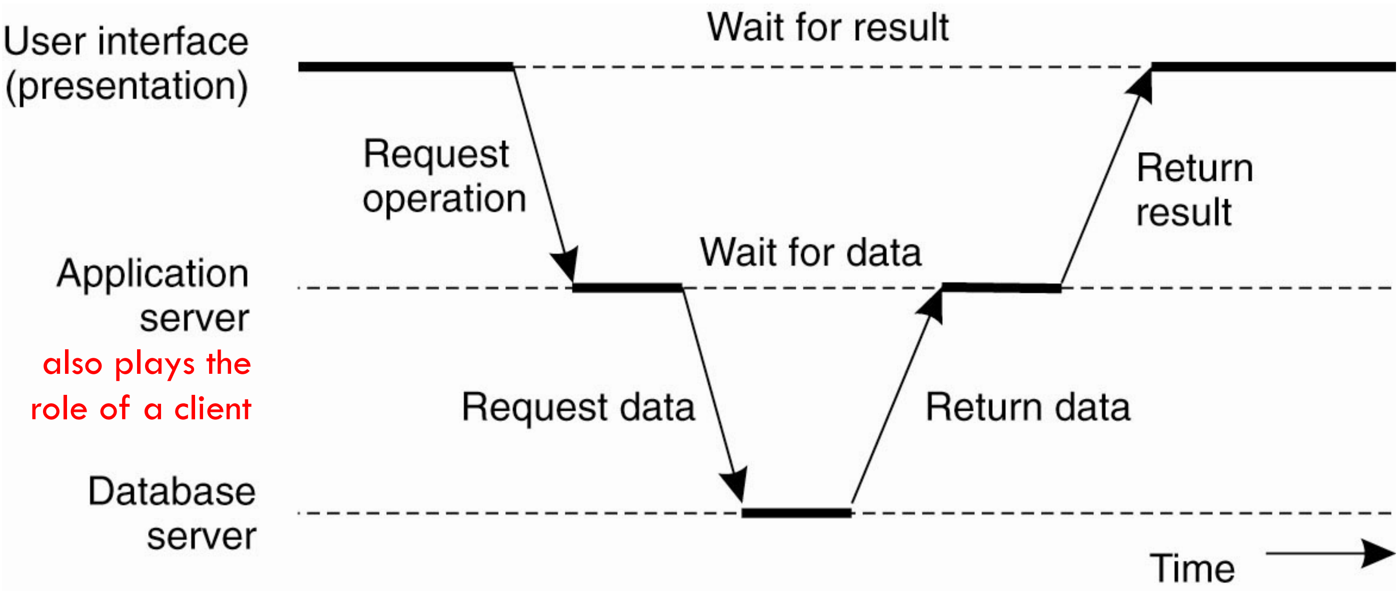
Multi-tiered architectures (cont'd)

A single machine can act both as a client and a server



Multi-tiered architectures (cont'd)

A single machine can act both as a client and a server



User interface (presentation)

Application server

also plays the role of a client

Database server

Wait for result

Request operation

Wait for data

Return result

Request data

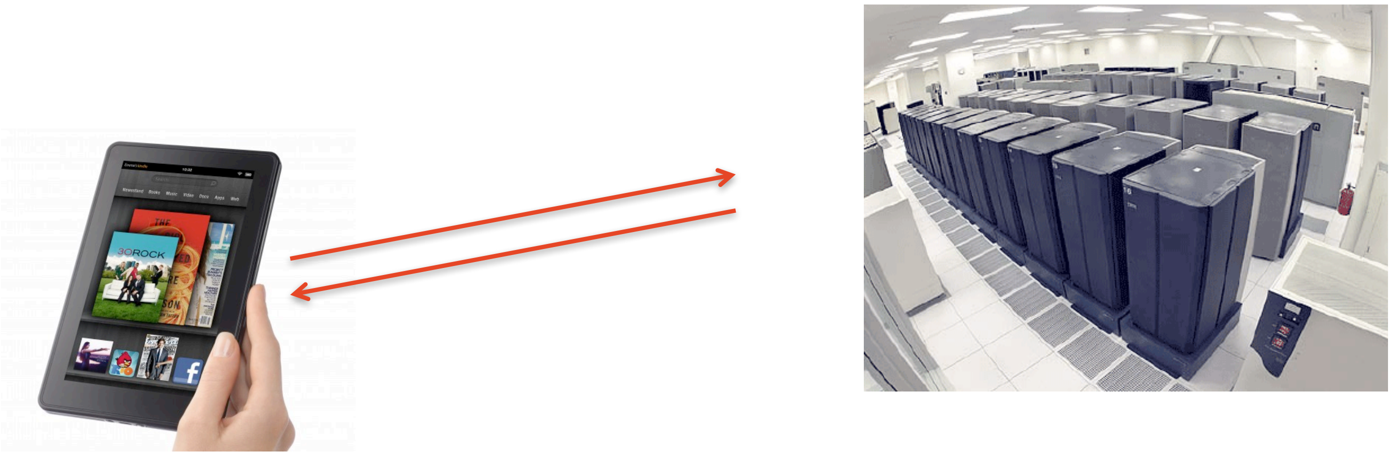
Return data

Time

Multi-tiered architectures (cont'd)

<u>Example</u>: Cloud computing

- *Cloud computing*: the delivery of computation or storage as a service to end-users.



Multi-tiered architectures (cont'd)

Example: Cloud computing

- The client hosts the user interface to launch the computation and prints the results
- The servers handle most of the computation upon request and sends back the results to the client
  - One server asks data to another server
  - Another does the computation

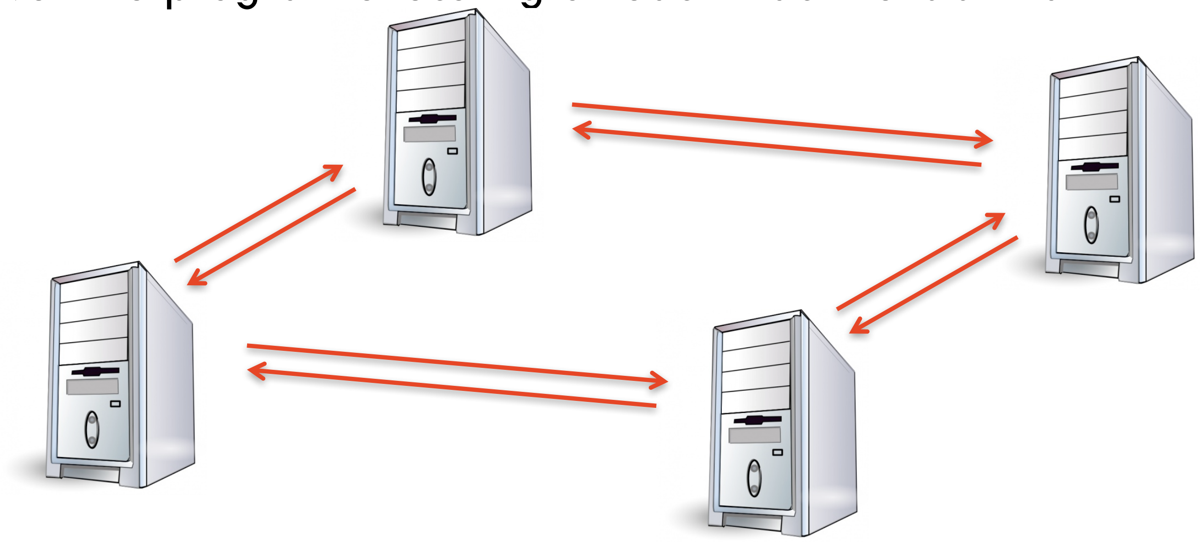
The Peer-to-Peer Organization

THE UNIVERSITY OF SYDNEY

The peer-to-peer model

Every machine acts similarly

- Every machine is both a client and a server
- No centralized control: the responsibility is distributed evenly
- Even the program executing on each machine is similar

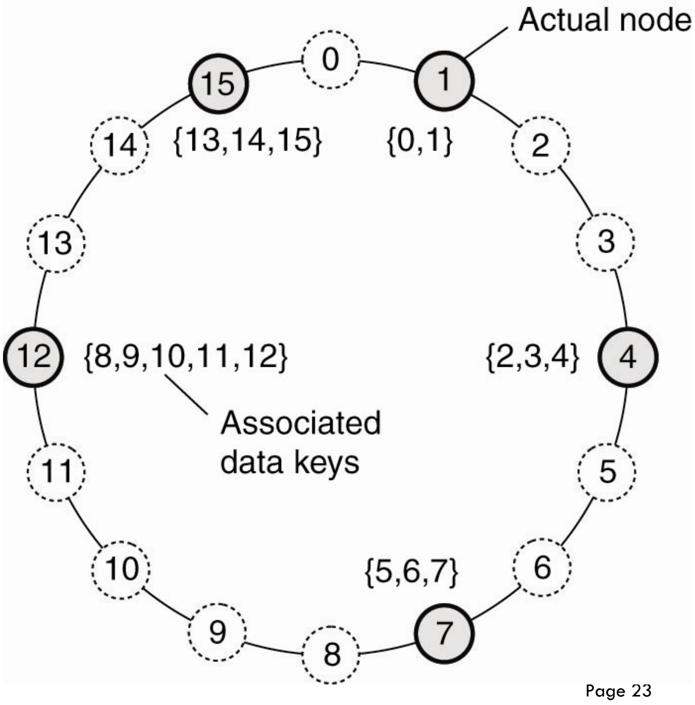


The peer-to-peer model (cont'd)

Example 1: Chord is an example of a Distributed Hash Table (DHT)

As a node:

- I have a successor peer
- I have a predecessor peer
- I have some shortcuts to other nodes to speedup delivery of requests
- I am responsible of a subset of the system data items (based on my unique identifier)

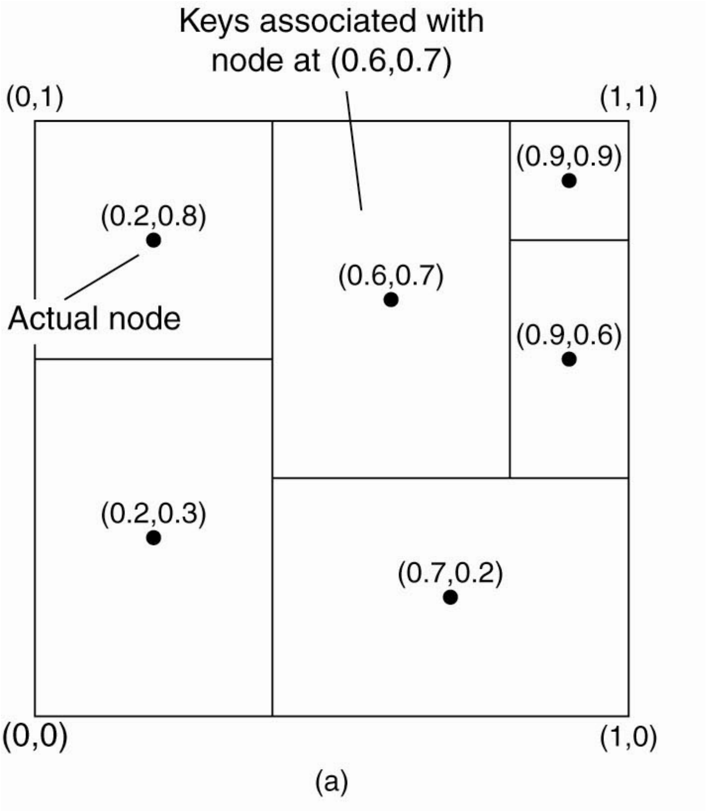


The peer-to-peer model (cont'd)

Example 2: CAN (Content Adressable Network), another DHT

As a node:

- I am responsible of a region of the system (based on my unique identifier)
- I have few neighbors, the nodes with adjacent regions, with which I can communicate



The peer-to-peer model (cont'd)

Example 3: BitTorrent, a file sharing application

- 35% of world-wide internet traffic in 2015.
- Used for Linux distribution, software patches, distributing
- Goal: quickly replicate large files to large number of clients

- Web server hosts a .torrent file (w/ file length, hash, tracker's URL…)
- A tracker (server or a DHT) tracks downloaders/owners of a file
- Files are divided into chunks (256kb-1MB)
- Downloaders download chunks from themselves (and owners)
- Tit-for-tat: the more one shares (server), the faster it can download (client)

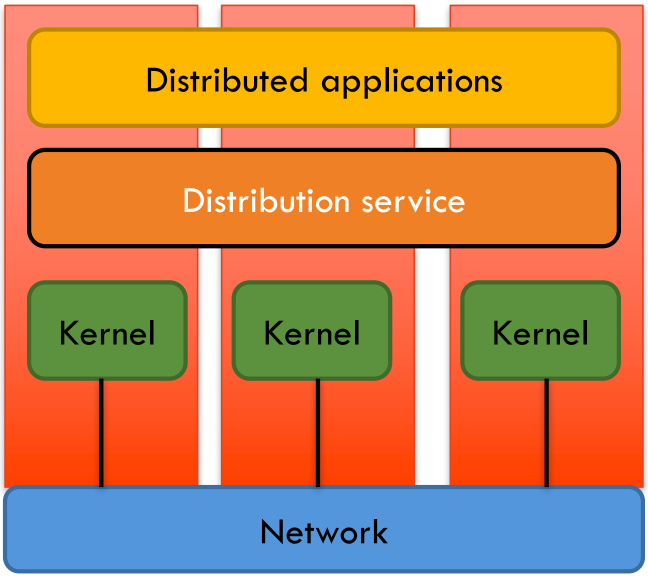
Distributed Operating Systems

THE UNIVERSITY OF SYDNEY

Distributed operating systems

<u>Distributed operating system</u>

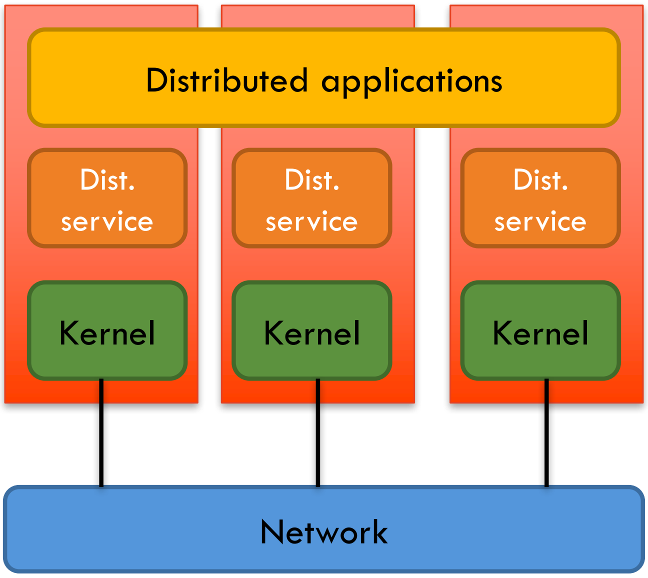
- This is a single system image, the system maintains a single copy of the resources



Distributed operating systems (con't)

<u>Network operating system</u>

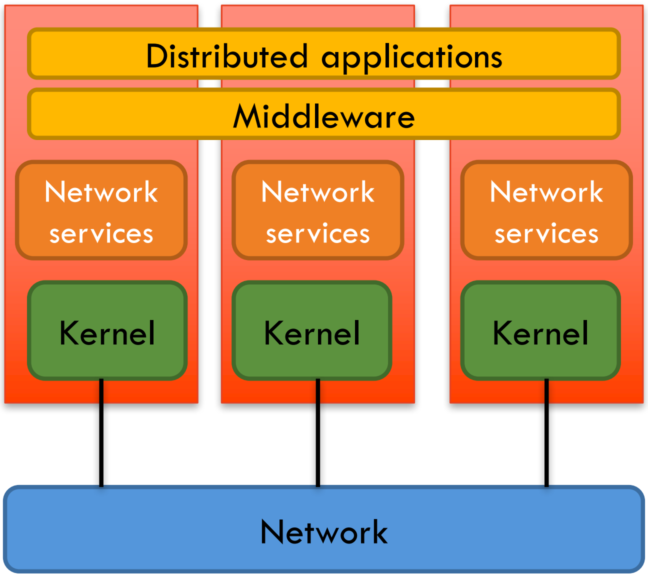
- Machines provide resources to other machines (e.g., UNIX rlogin)
- The OS can vary from one machine to another, essentially file sharing



Distributed operating systems (con't)

Middleware

- A layer over the network services providing general services to applications in a very transparent manner (systems can differ)



Conclusion

- Client and server are used to identify the role of communication participant

- Client and server roles may run on:
  - The same machine
  - Distinct machines with very different resources
  - Distinct machines with similar resources

- In operating systems, applications may run on top of a single distributed operating system, of network operating systems (multiple OSes), or a middleware (multiple OSes looking like a single OS).