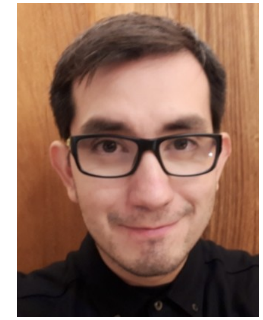
Distributed Systems

José Oramas

Overview of the course

Theory lectures

- **Prof. Dr. José Oramas** (Jose.Oramas@UAntwerpen.be)
- Thursdays 13h45 – 15h45 - G 005, CMI.



Who am I?

- Artificial Intelligence research at the Internet Data Lab (IDLab)
- Also teaching:
  - Operating Systems (1500WETOPS)
  - Artificial Neural Networks (2500WETANN)

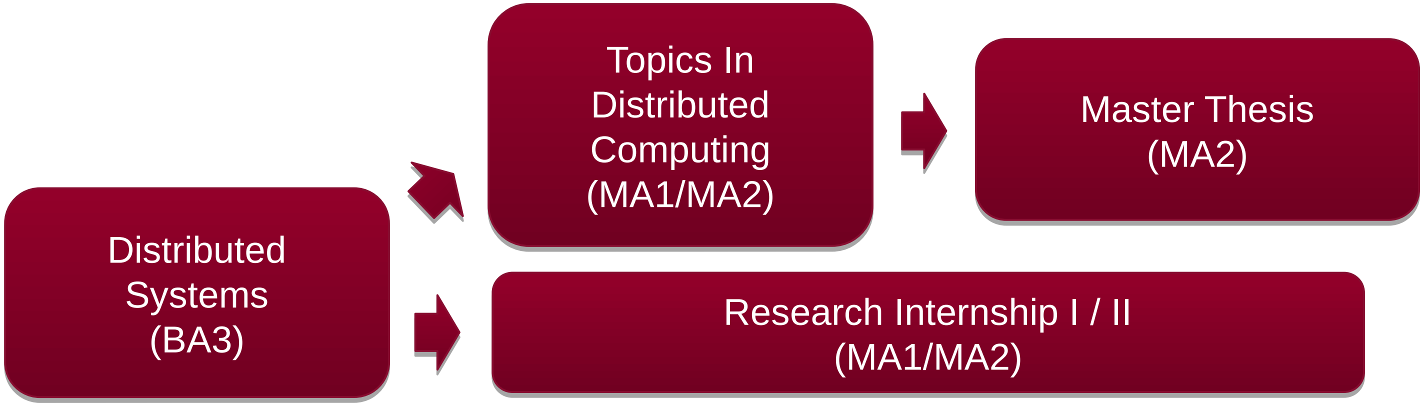
Course goals & context

Understand distributed systems

- Principles & nomenclature (what?)
- Challenges and desirable properties (why difficult?)
- Important principles, algorithms and design approaches (how?)

Obtain experience in practical design, development, debugging and testing of distributed system software

Course context



Mix of theoretical and practical knowledge on distributed systems

→ Foundations of distributed systems: what is a distributed system?
→ How would you design distributed systems?
→ What are the new and popular technologies (Docker, Kubernetes, Hadoop, etc.)

Theory lectures

Introduction

- Session 1 Course Overview & Introduction
- Session 2 Middleware

Theory lectures

Introduction

- Session 1 Course Overview & Introduction
- Session 2 Middleware

Services

- Session 3 Service-Oriented Architectures & Cloud Computing
- Session 4 Web Services
- Session 5 Microservices
- Session 6 Distributed Storage

Theory lectures

Introduction

- Session 1 Course Overview & Introduction
- Session 2 Middleware

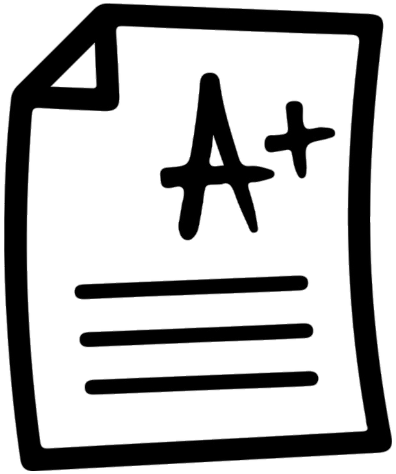
Services

- Session 3 Service-Oriented Architectures & Cloud Computing
- Session 4 Web Services
- Session 5 Microservices
- Session 6 Distributed Storage

Algorithms

- Session 7 Coordination
- Session 8 Replication

Evaluation



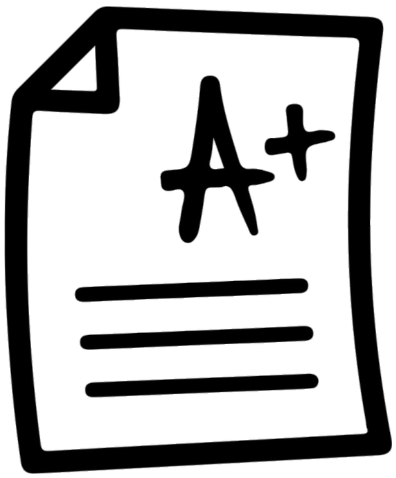
Theory (50%)

- Written exam
- Closed book
- No oral presentation

Projects (50%) - individual

- Two small programming projects on not so new, new and upcoming technologies: Hadoop, Web Services, microservices, etc.

Evaluation



Theory (50%)

- Written exam
- Closed book
- No oral presentation

Projects (50%) - individual

- Two small programming projects on not so new, new and upcoming technologies: Hadoop, Web Services, microservices, etc.

Important:

- You need to **succeed on all these two parts** in order to pass the course.
- You will need a grade of at least 10/20 for each part.
- Partial exceptions are possible, but **only within the same academic year**.

Lab sessions

Benjamin Vandersmissen
Benjamin.Vandersmissen@
UAntwerpen.be

Fabian Denoodt
Fabian.Denoodt@
UAntwerpen.be

- **Wednesdays 08h30 – 10h30, G 025**
- **Presentation and discussion** related to the projects.
- **~ 4 Sessions** during the semester.
- Sessions **date/time to be announced via Blackboard.**

Communication

Blackboard

- Announcements
- Course material
- Projects and other assignments
- Etc.

Email

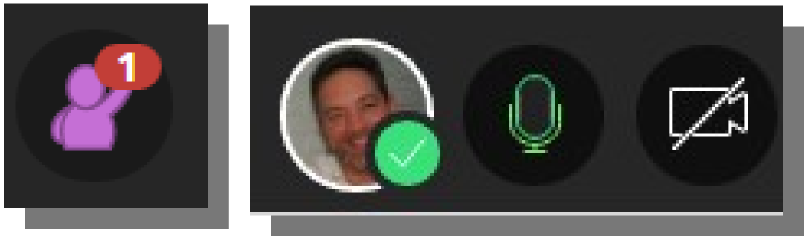
- General questions

Q&A Lab Session

- Under request via email

What if I cannot come to the class?

Recordings will be made available “eventually” via Blackboard



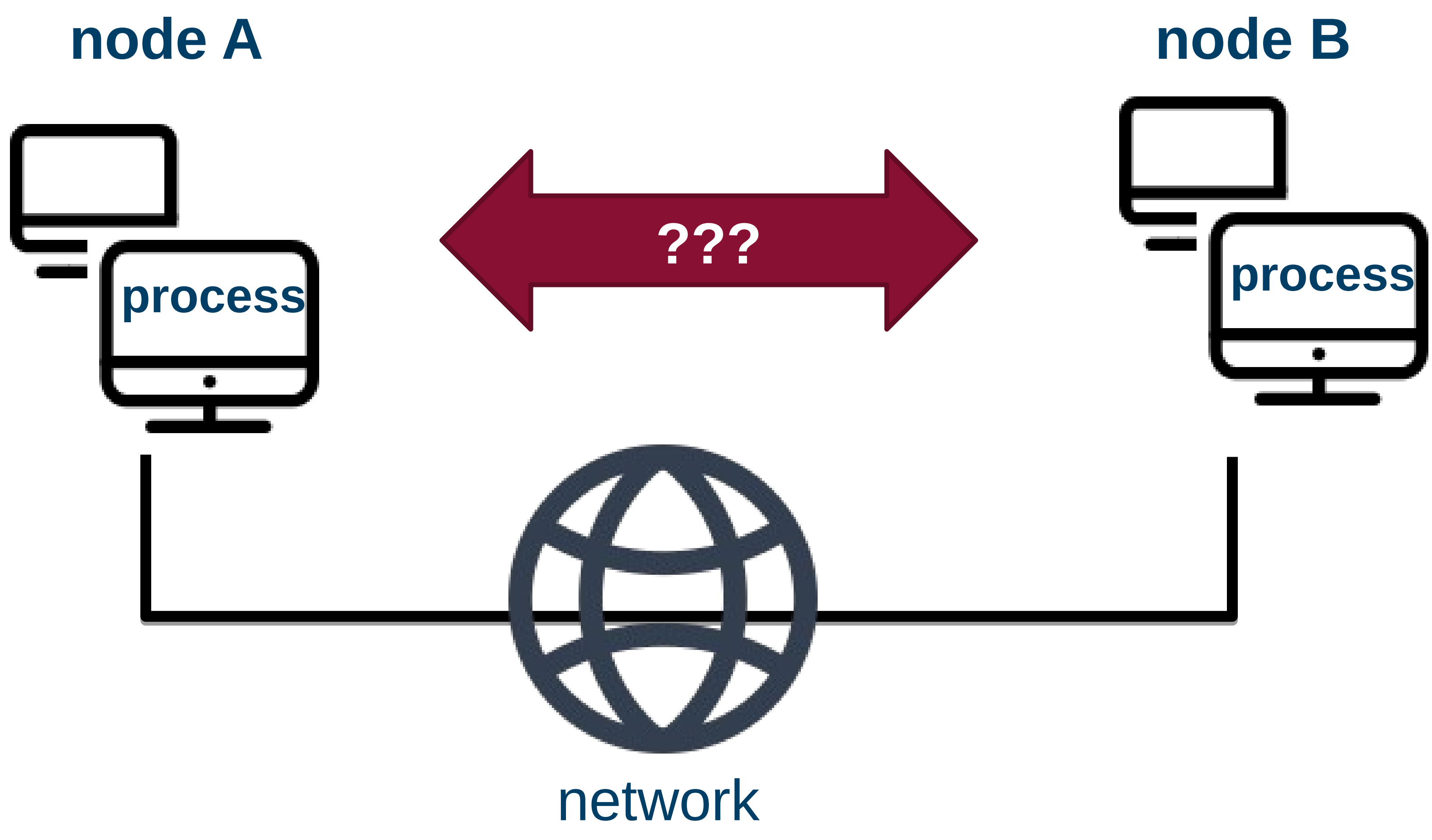
Ready?

Let’s start

Introduction

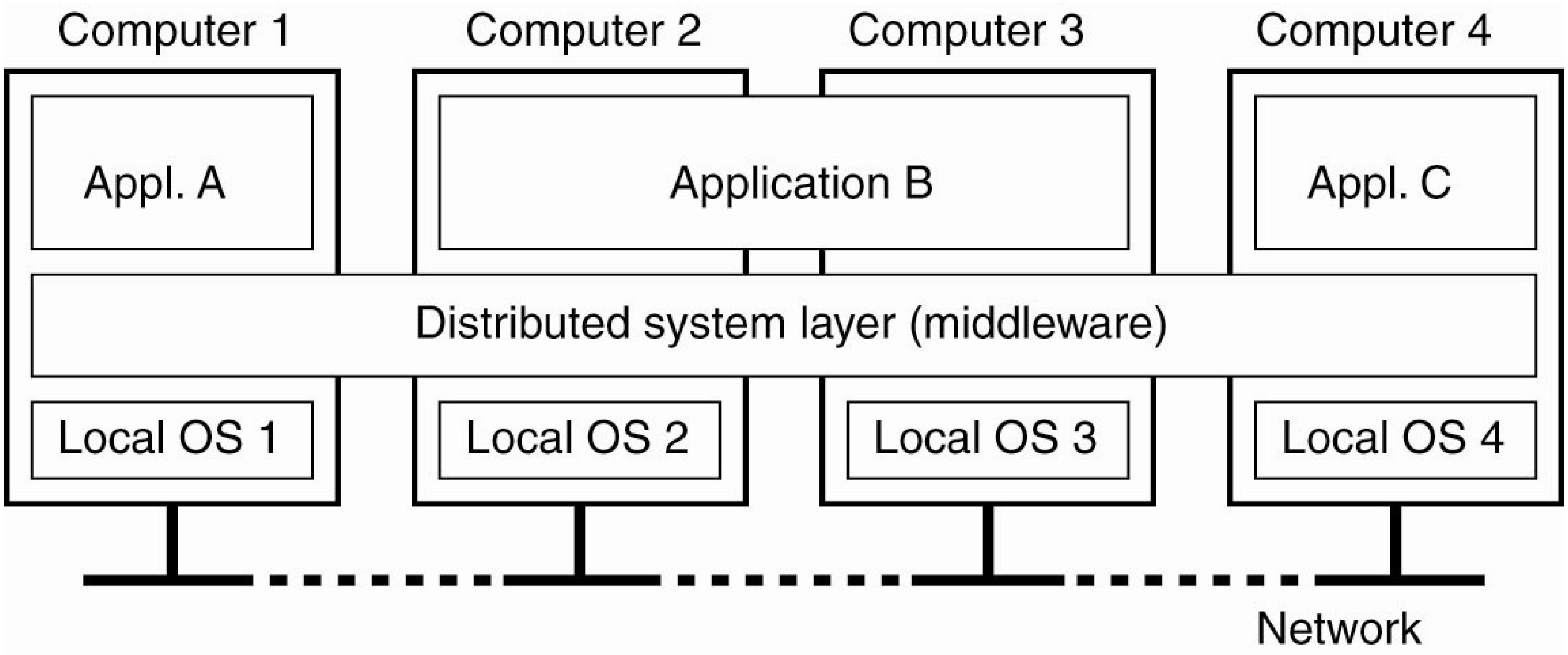

WHEN
WHO
WHAT
?
HOW
WHY
WHERE

What is this course about?



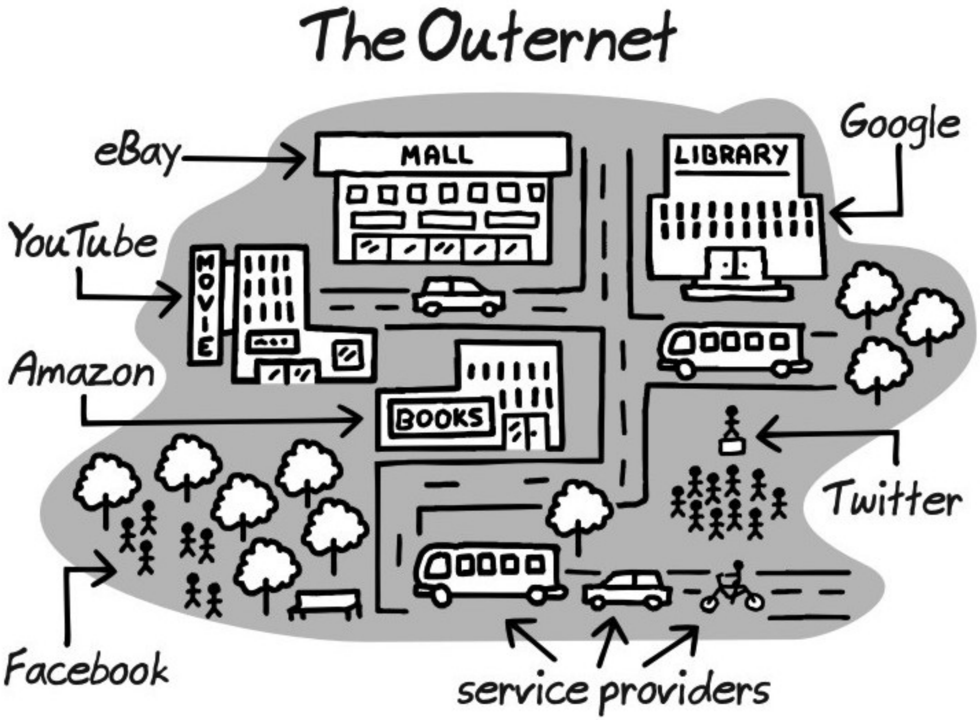
What?

Informal definition
A distributed system is a collection of **independent** computers that appears to its users as a **single coherent** system

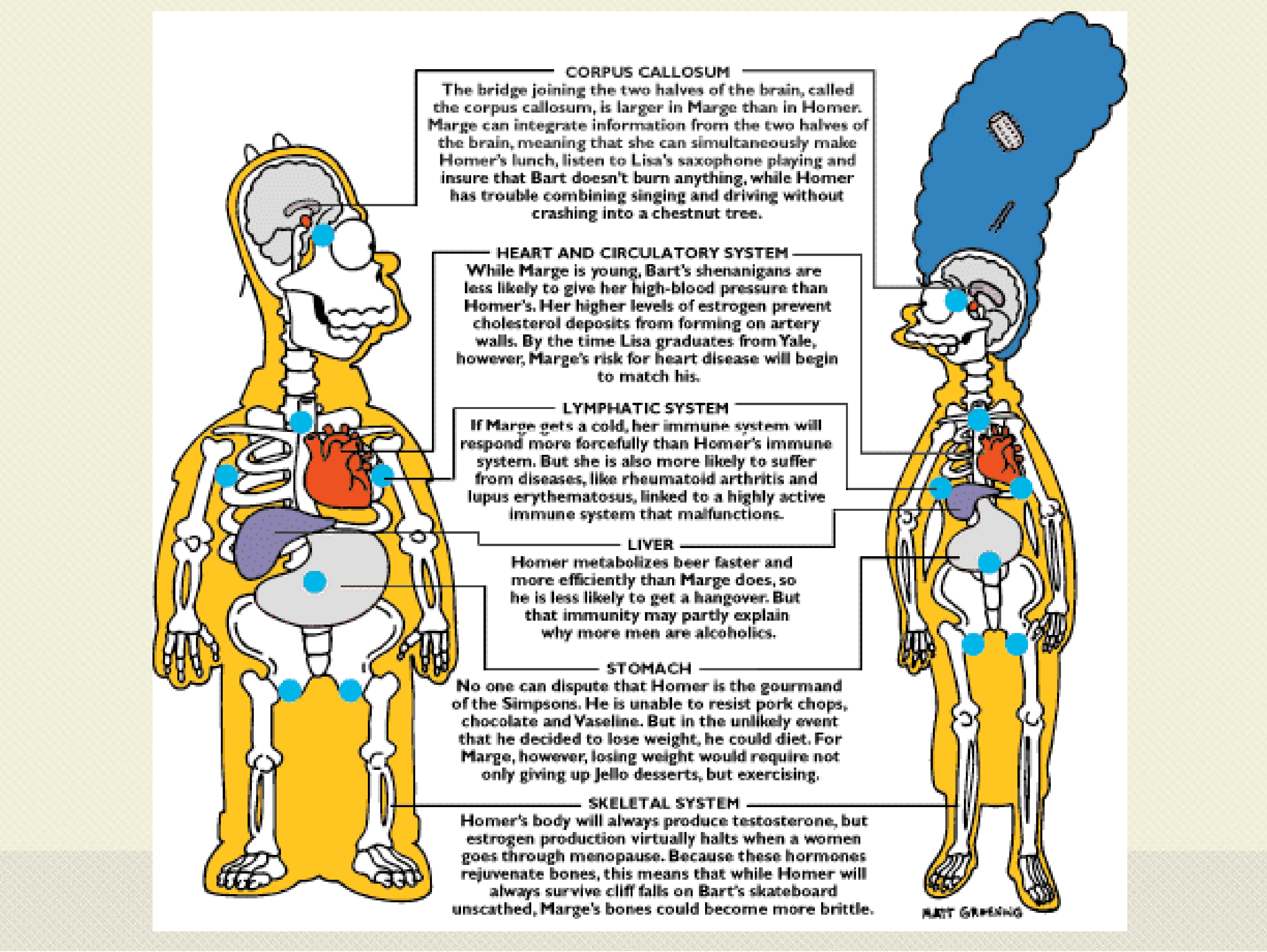


Build a powerful system out of many simpler systems
✉ The network is the computer

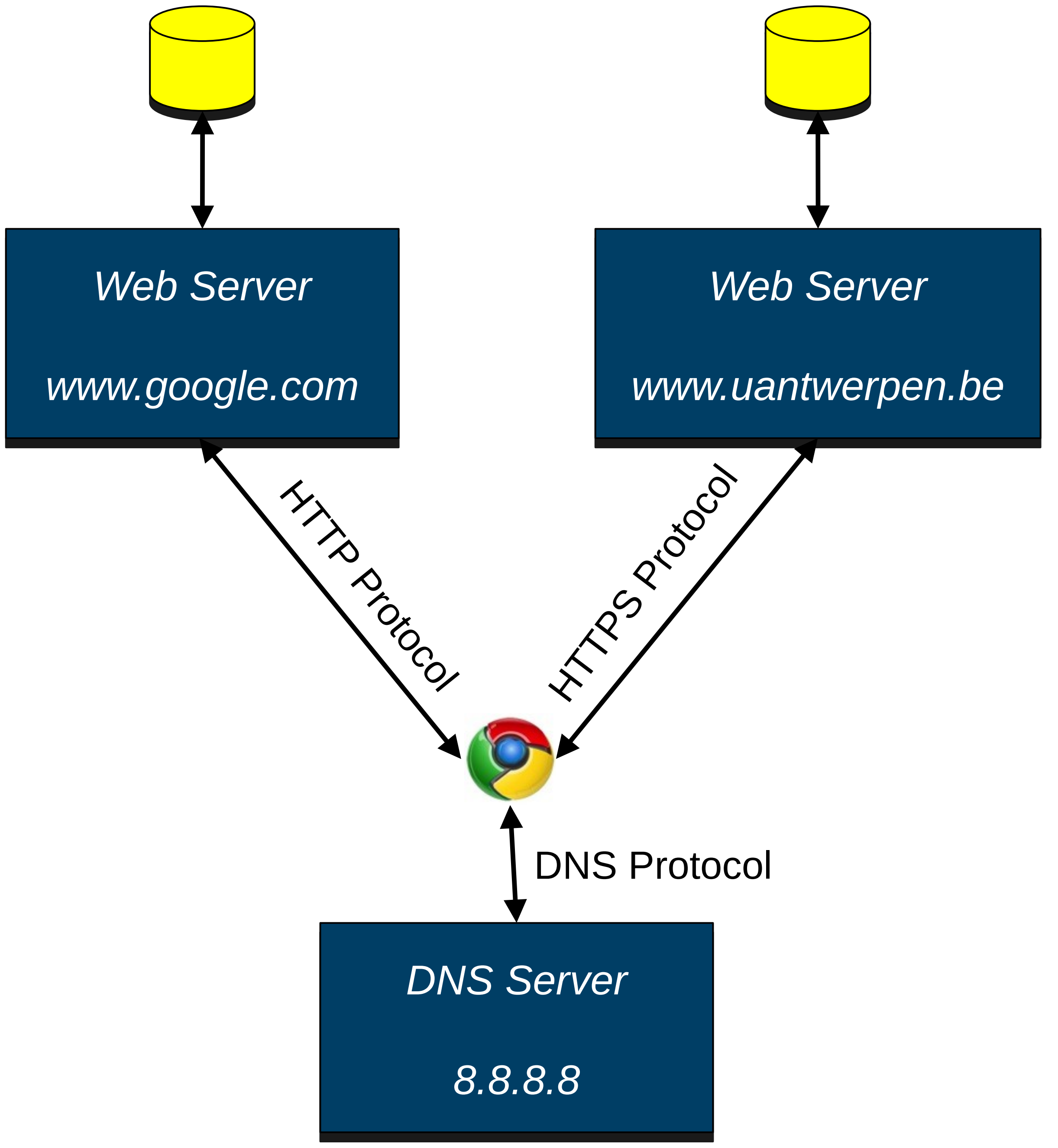
Examples and terminology



Examples of distributed systems



The World-Wide Web

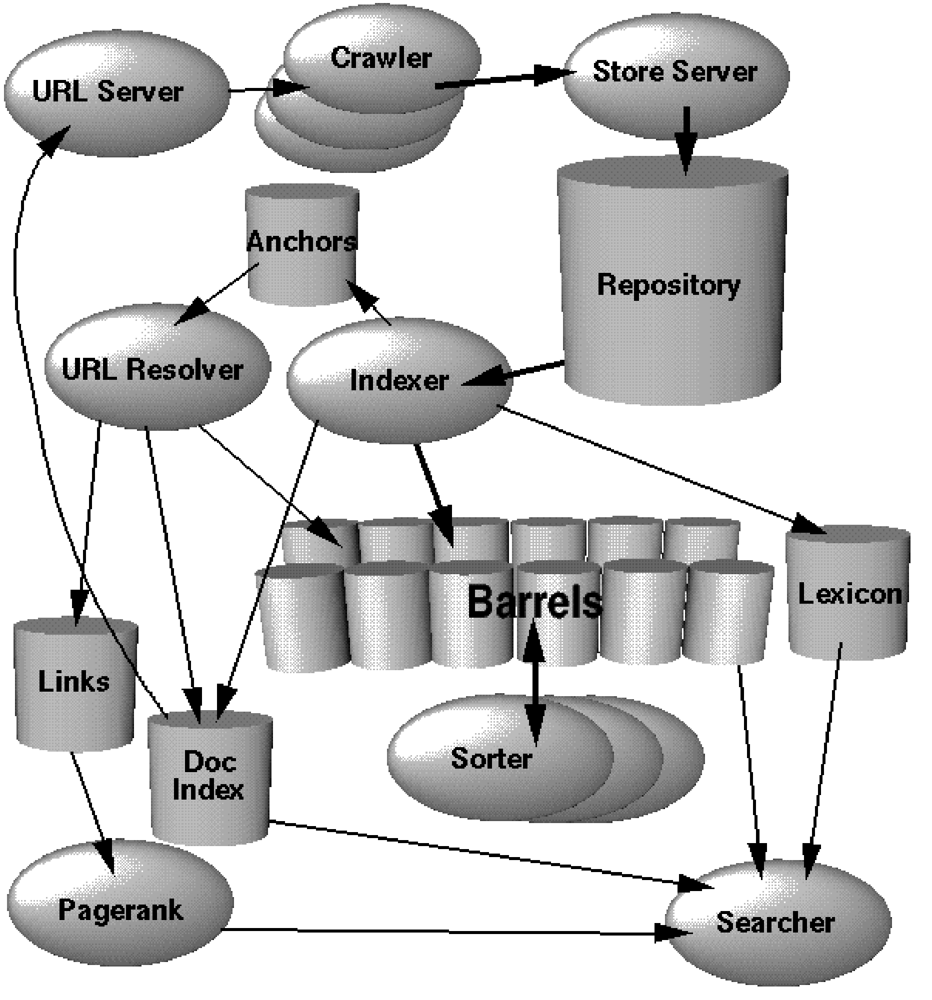


Google Search engine

Start of Google (1998)

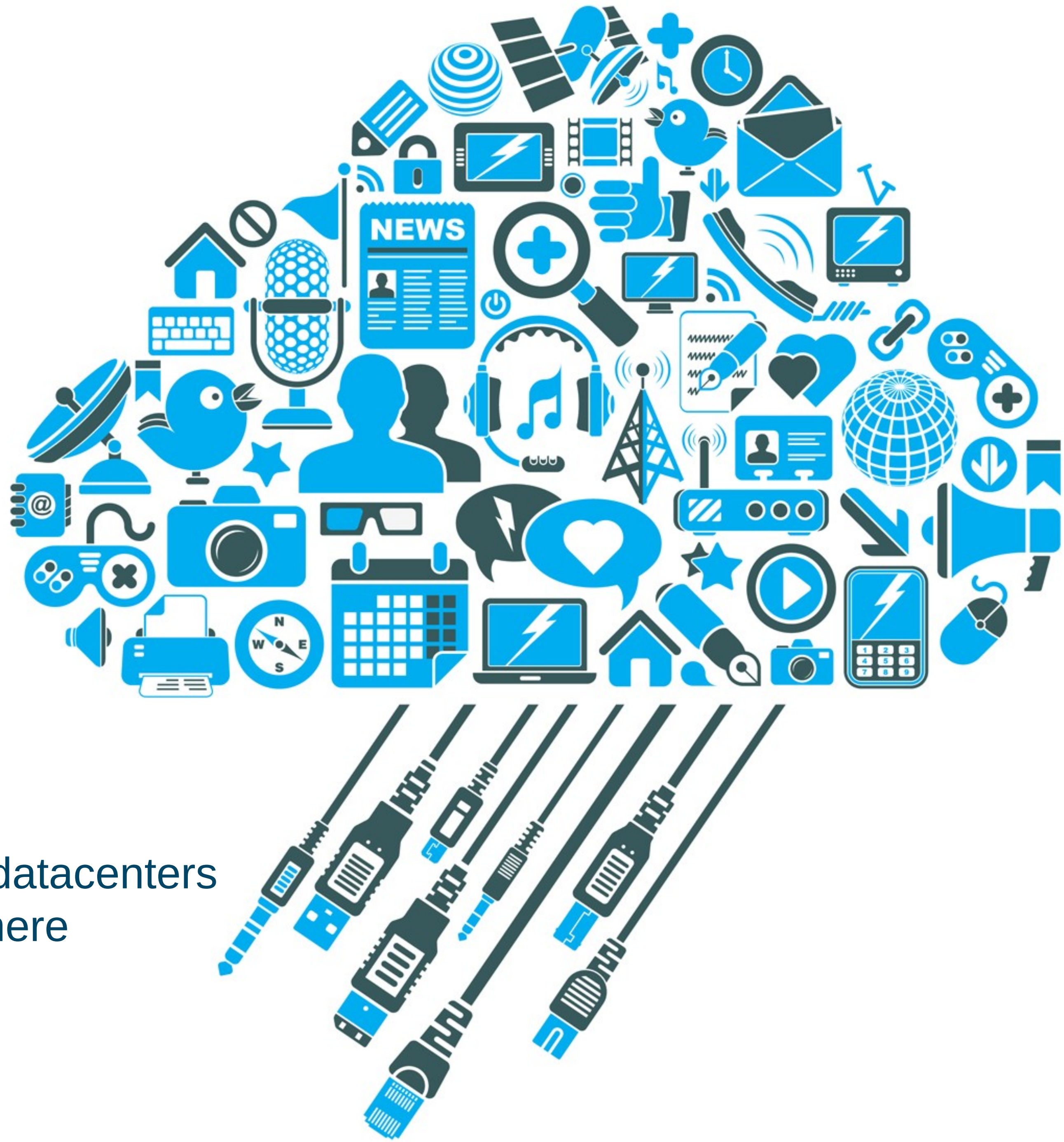
Sergey Brin

Lawrence Page



Paper: The Anatomy of a Large-Scale Hypertextual Web Search Engine

Google datacenters around the world

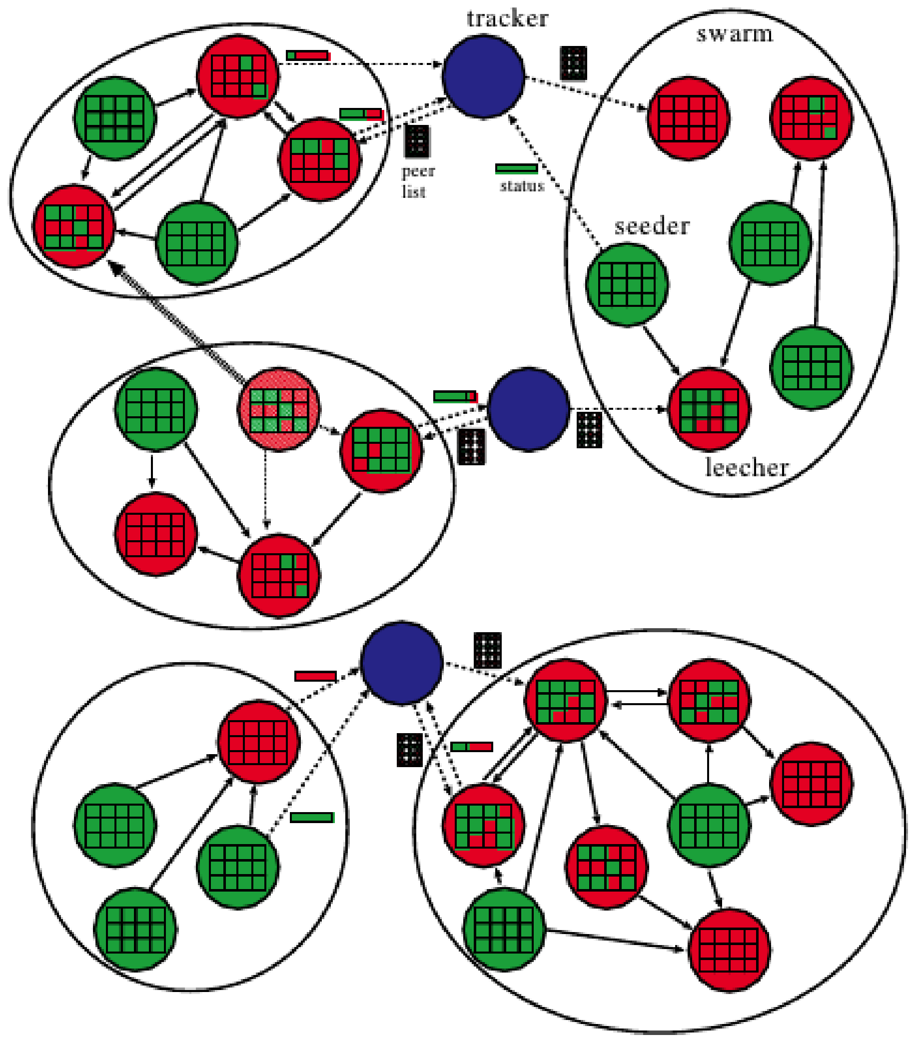


Cloud computing

- Computing is done on datacenters
- User does not know where

Peer 2 Peer applications

Example: BitTorrent

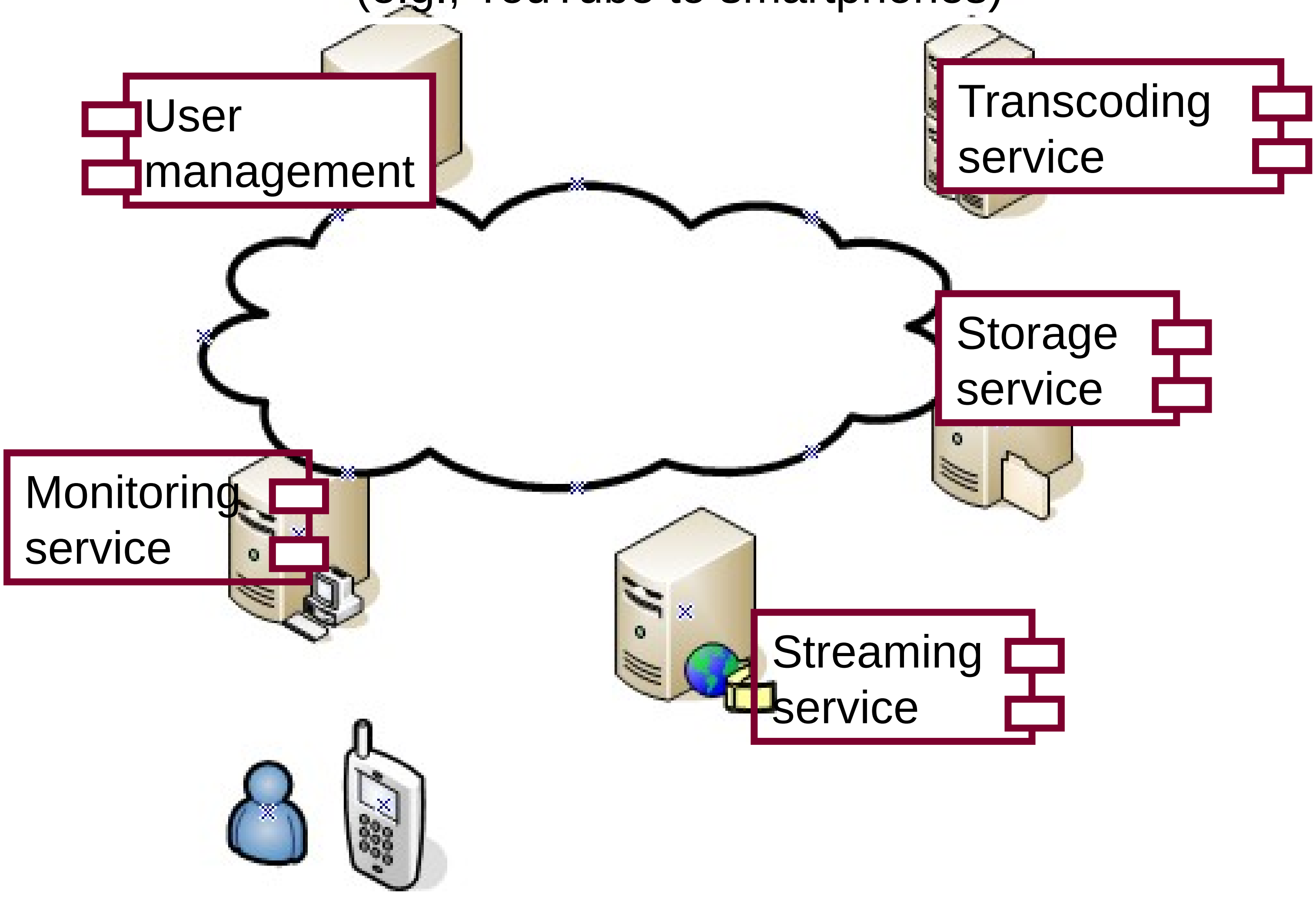


DEFINING A DISTRIBUTED SYSTEM

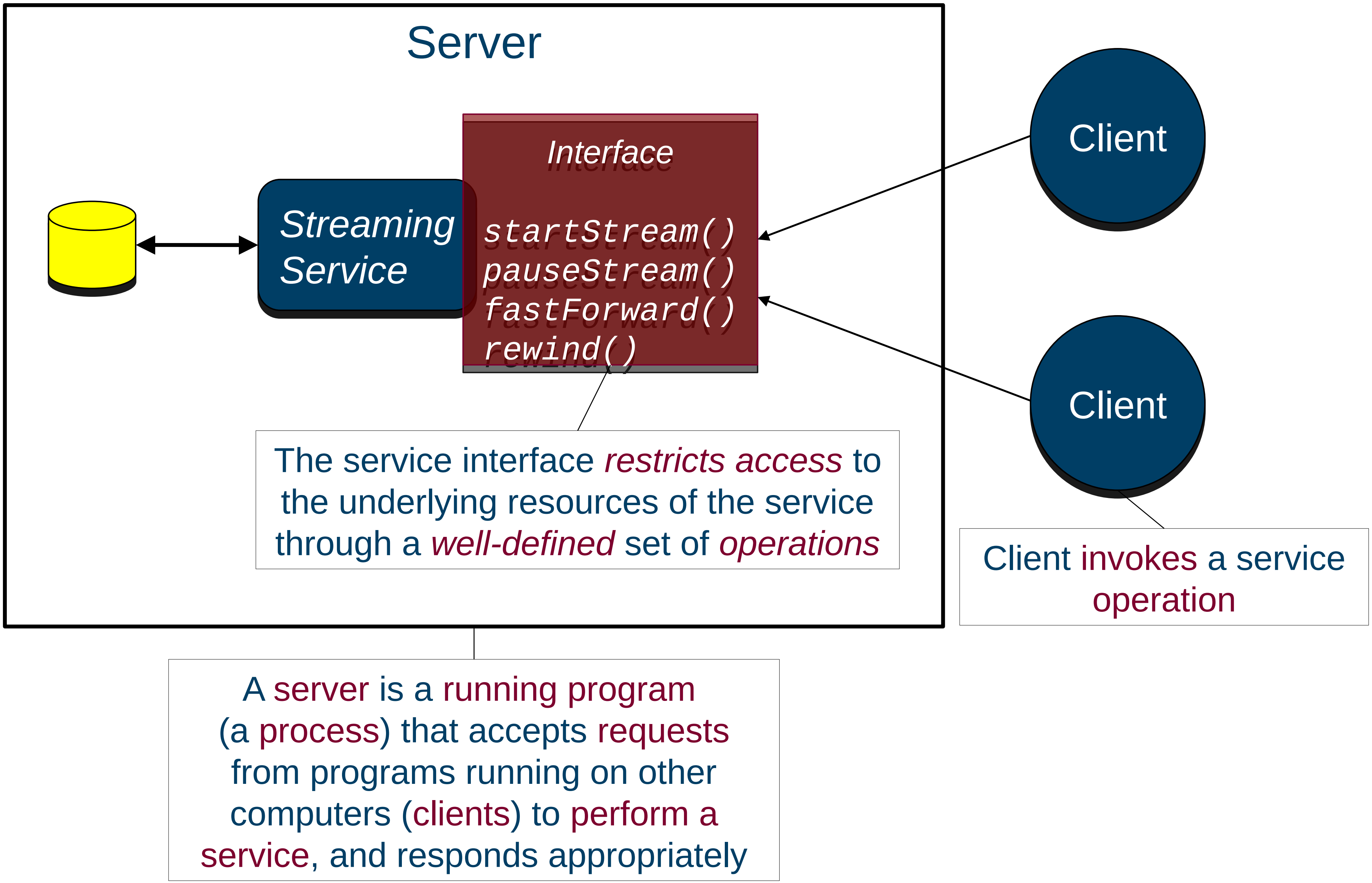
Service

= part of a computer system managing a collection of related resources, presenting their functionality to users and applications

Streaming content to mobile clients
(e.g., YouTube to smartphones)



Service, client and server

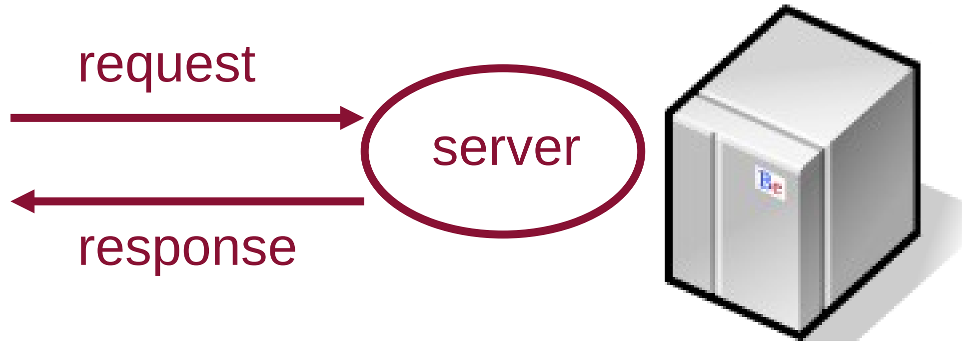


Client and Server

A server

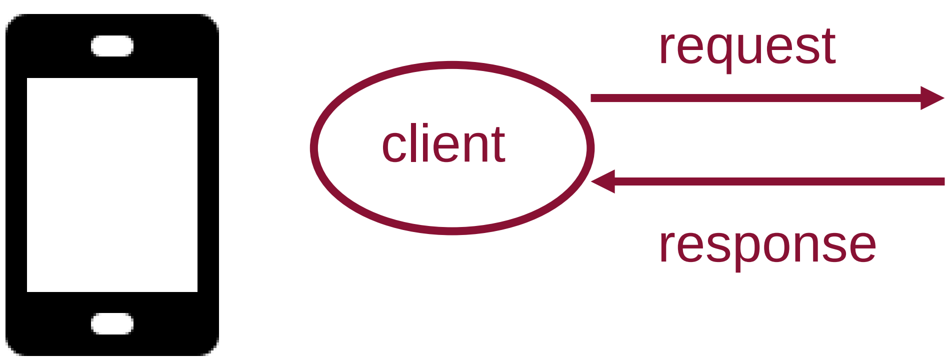
= running process on networked computer

- accepting requests to perform a service
- responding appropriately



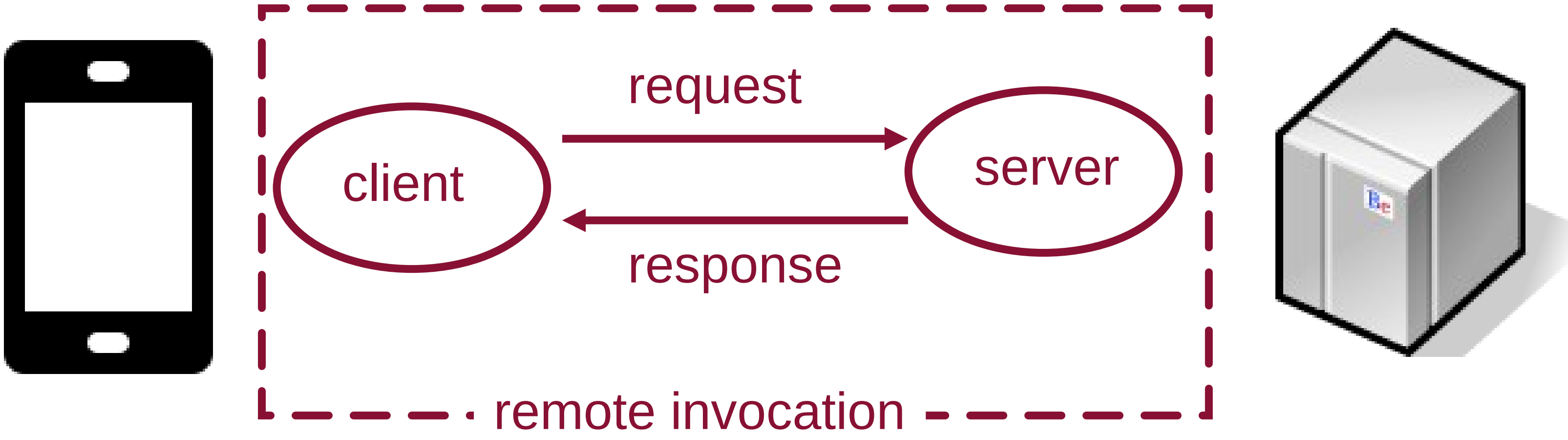
A client

= running process on networked computer sending service requests to servers



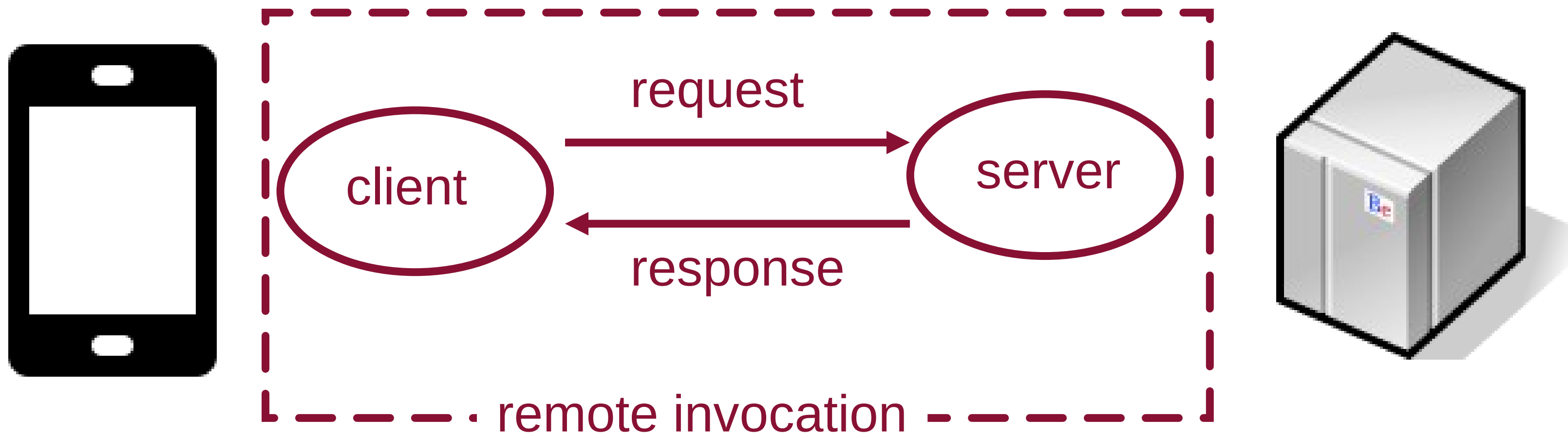
Remote invocation

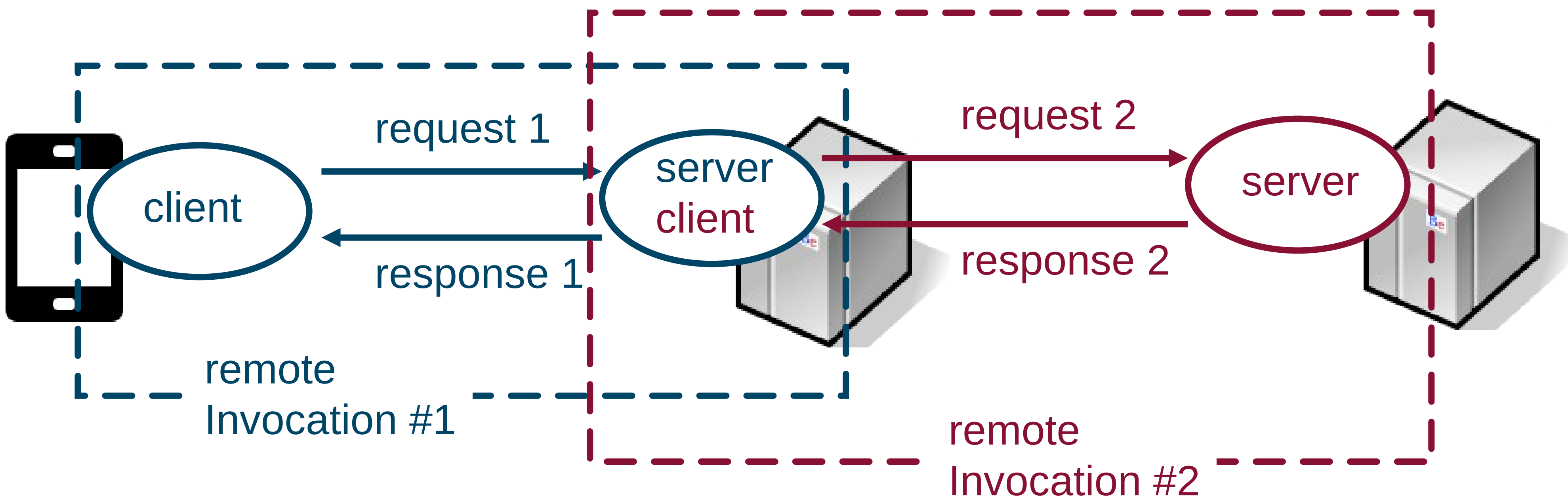
A remote invocation = complete interaction between client and server to process single request



Remote invocation

A remote invocation = complete interaction between client and server to process single request





Why distributed systems?

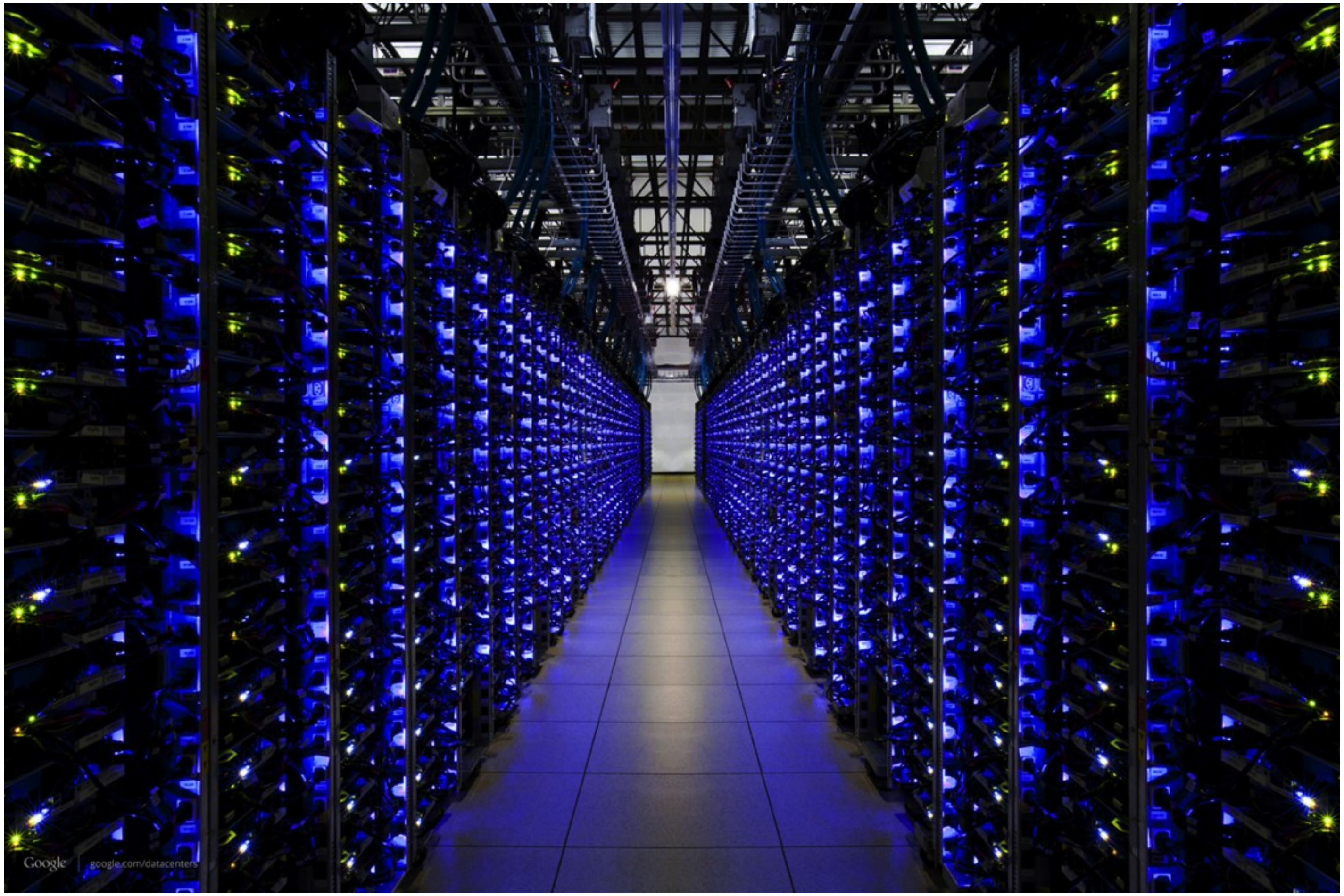
- **Cost** Networked commodity systems can render the best performance/$
- **Capability** Many computational problems are too large for any single system because of memory, data storage, computational requirements
- **Concurrency** Many 'large' problems have inherent options for parallelism
  Era of horizontal versus vertical scaling
- **Reliability** Distributing redundant components minimizes the probability faults impact the user
- **Integration** For organizations to interact, their systems need to interact
  Highly specialized infrastructures need to be integrated and shared (e.g. radio telescopes, mass storage facilities, experimental facilities)
- **Distribution** E-mail, WWW, … are inherently distributed as users are geographically spread

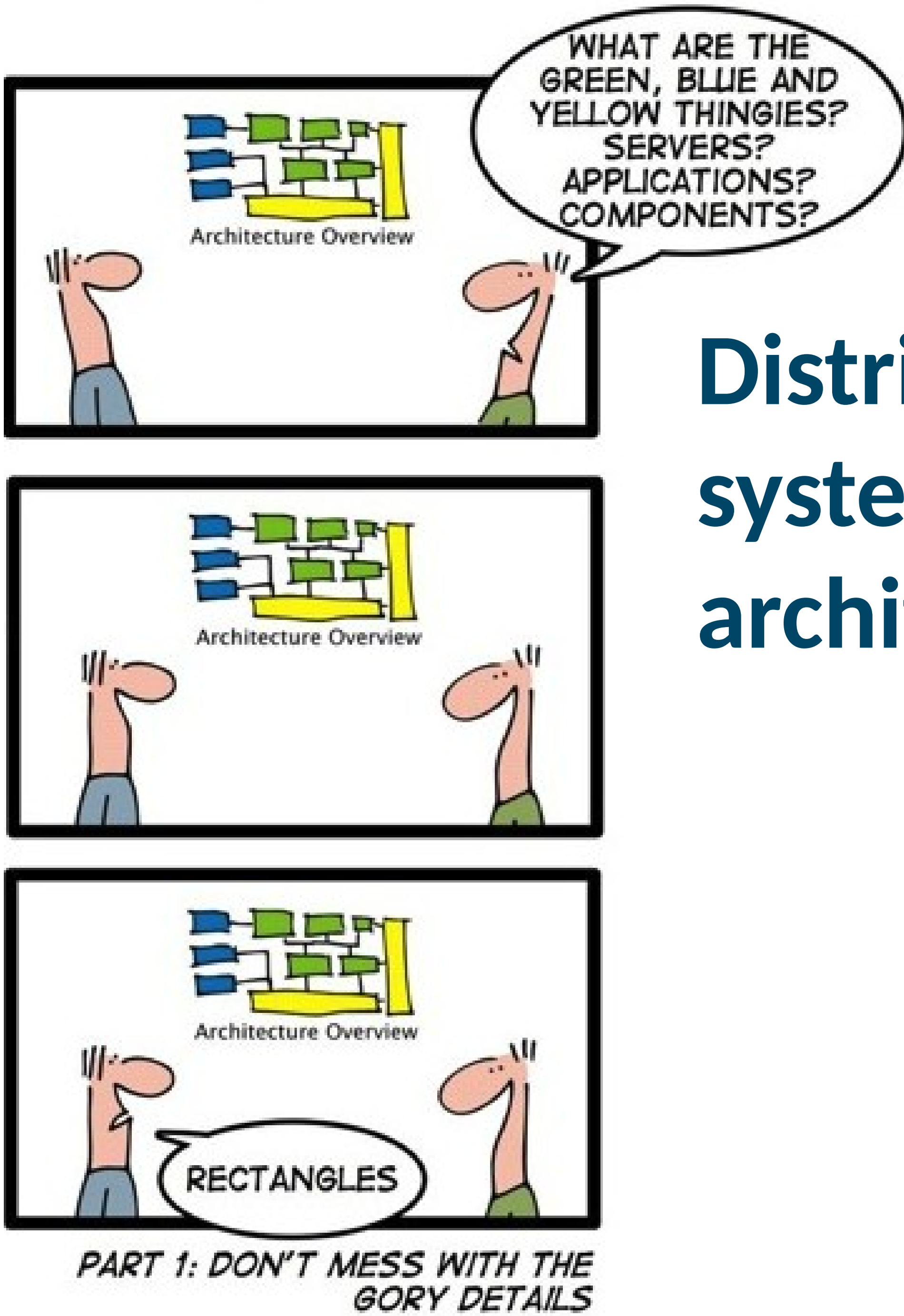
Why distributed systems?

Many problems / challenges

- no limit on spatial extent, difficult to manage
- no global time notion
- almost always concurrent execution
- (partial) failures likely to happen

Will be discussed in this course





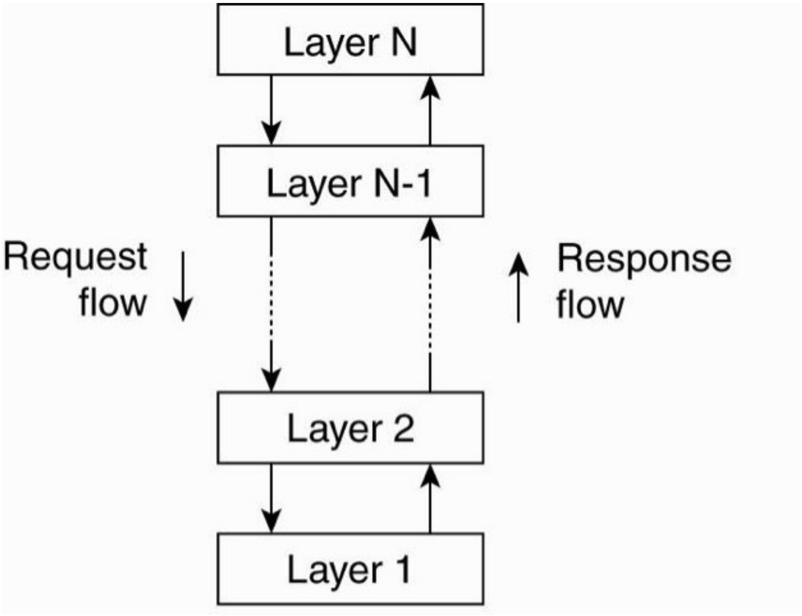
Distributed system architectures

Logical architecture styles: coupled

Coupled architectures:

- Components are tightly linked with each other
- Removing/adding a component is non-trivial

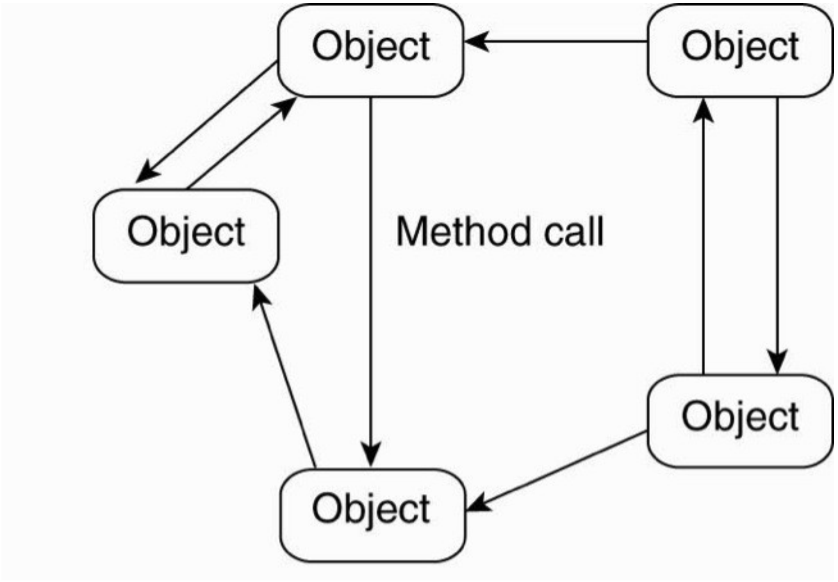
Layered



layer only interacts with neighbour

+ reduced number of interfaces, dependencies
+ easy replacement of a layer
- possible duplication of functionality

Object-based



Interacting objects

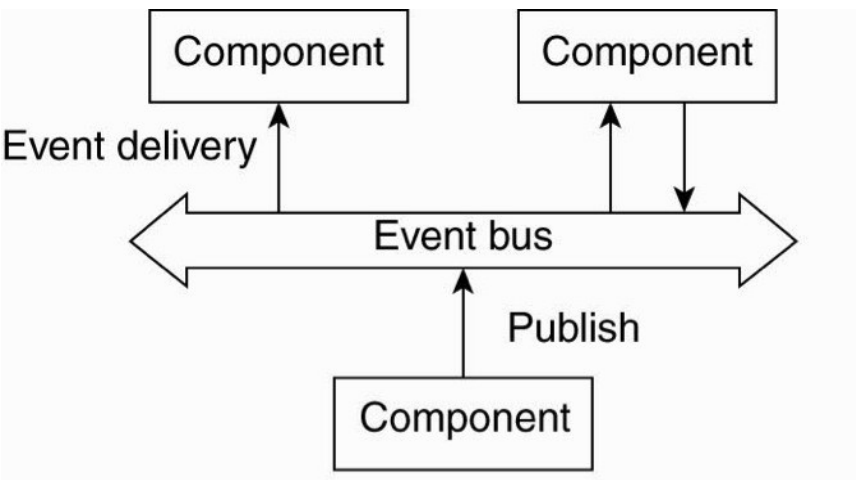
no predefined interaction pattern
+ highly flexible
- complex to manage and maintain

Logical architecture styles: decoupled

De-coupled architectures:

- Components are loosely linked with each other
- Removing/adding a component is easier and can happen frequently

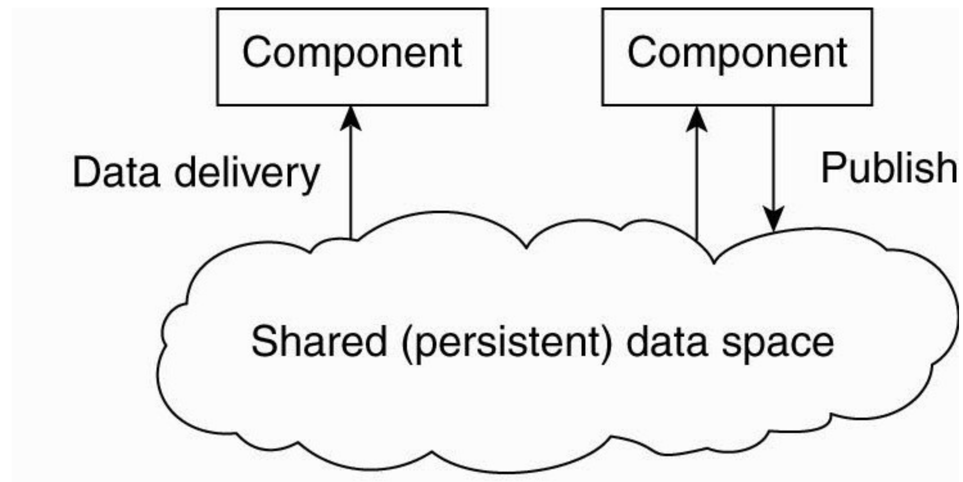
Event-based



Event based interaction

"publish-subscribe" style
+ loose coupling of components
related: message based interaction
(also decoupling in time)
often used to integrate legacy systems

Data-centric



Data centric architecture

only interaction through shared data base
+ loose coupling of components
- possibly slow (central bottleneck,
  locking, ...)

System architecture

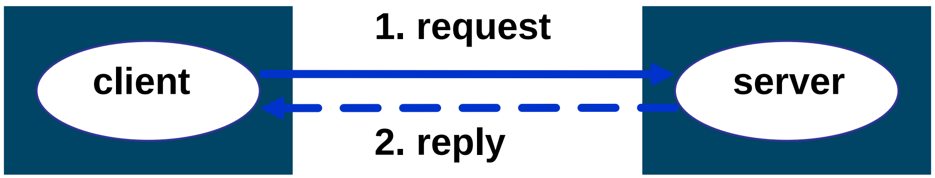
How is the software architecture instantiated using hardware components?

How are the hardware components organized?

How to map logical components to actually deployed components (replicated ?, P2P, pure client server, …)

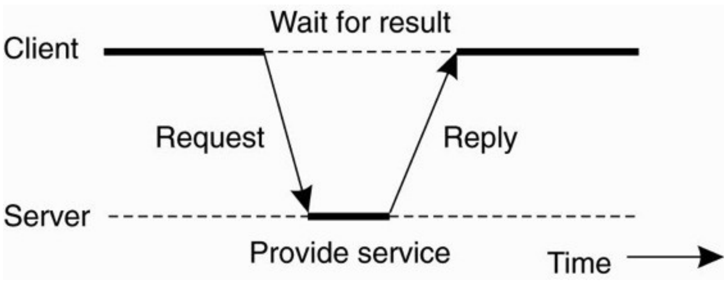
Client server architectures

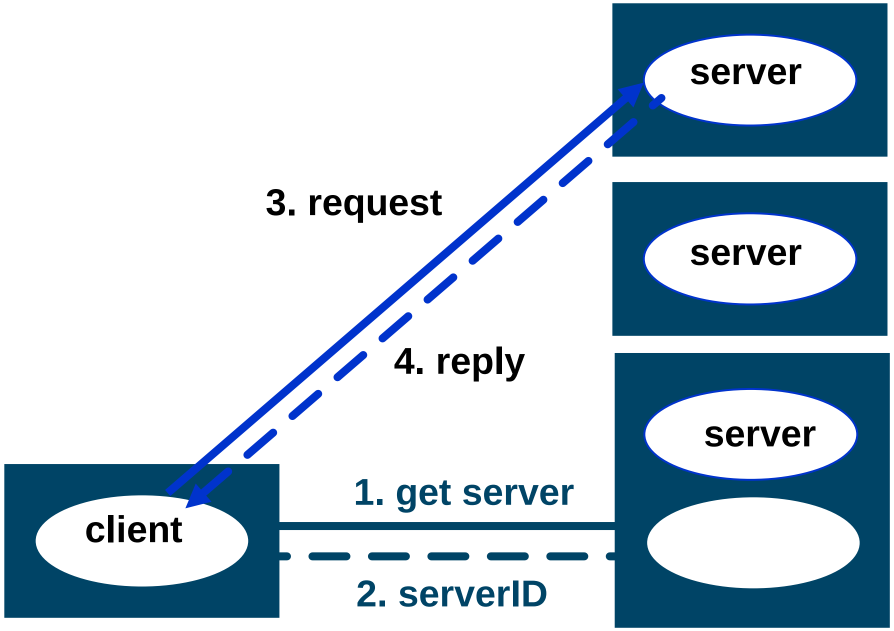
"simple" client-server



client- multiserver
(explicit server lookup)

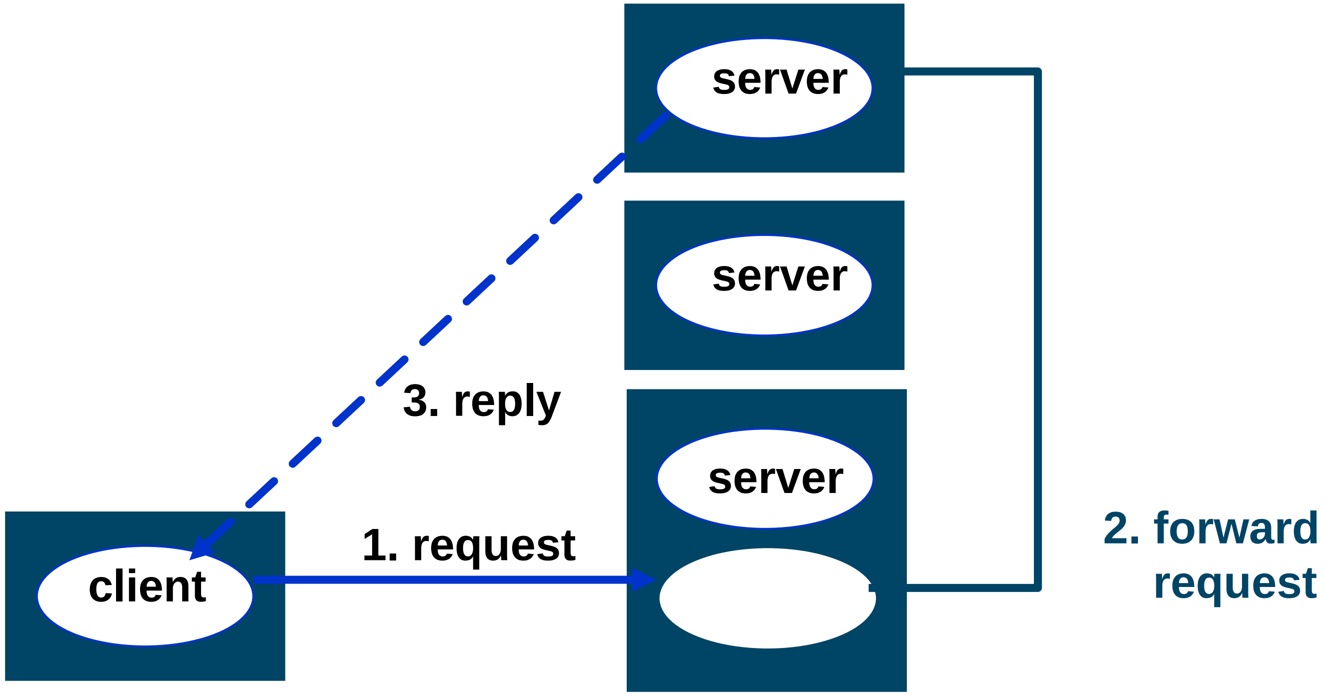
e.g. DNS based load balancing





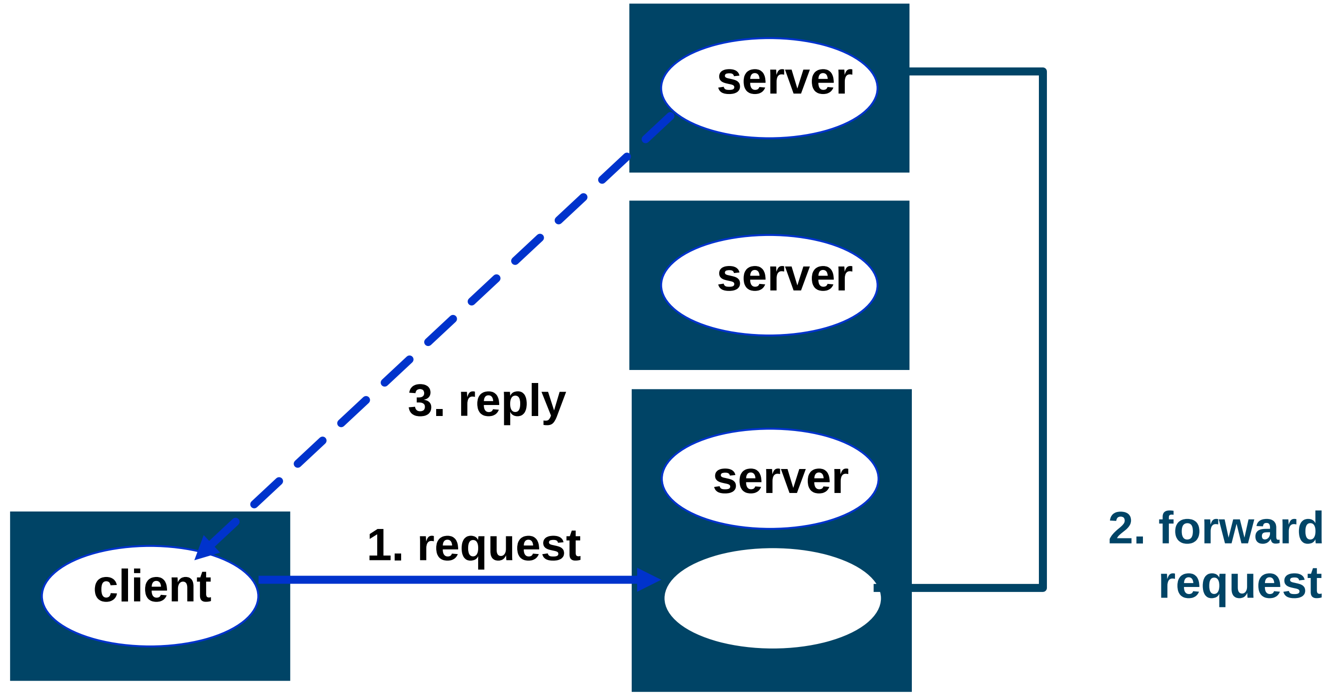
System architecture: client-server

client- multiserver
(implicit server lookup)

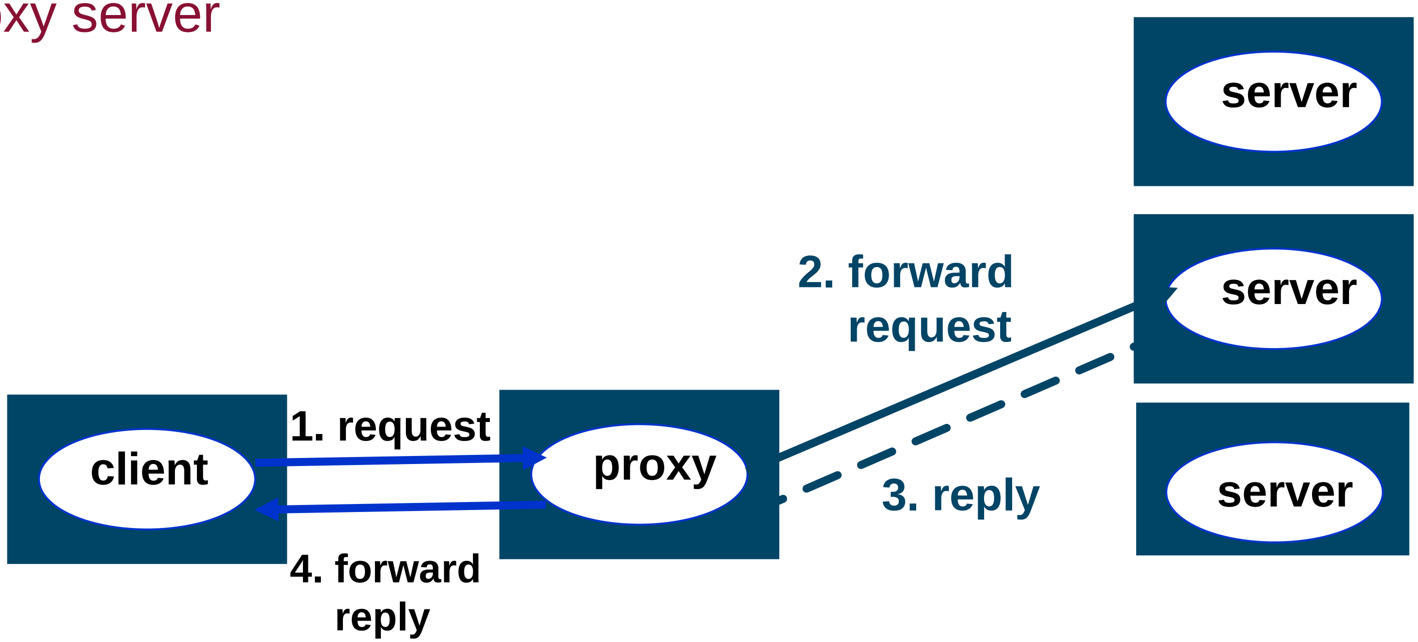


System architecture: client-server

client- multiserver
(implicit server lookup)

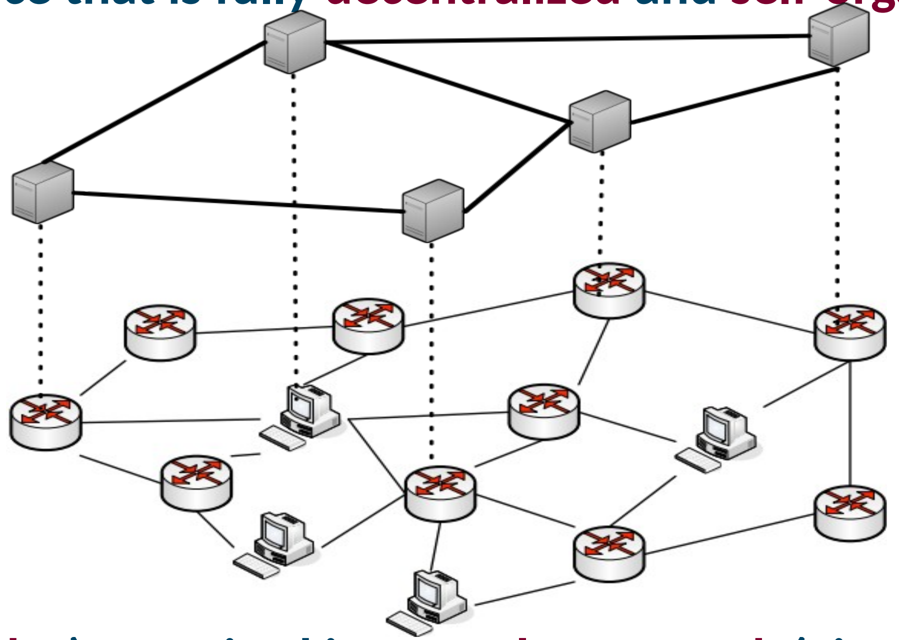


proxy server



Peer-to-peer architectures

Deliver a service that is fully decentralized and self-organizing



P2P network

physical topology

Processes (nodes) organized in a overlay network (virtual network)

- Each node fulfills both a client and a server role (servant)
- Nodes and data item keys are assigned Globally Unique Identifiers (GUIDs)
- Nodes have no or limited direct knowledge on other nodes
- Application-level message routing
- Nodes are volatile
- Structured or unstructured

To Conclude

Pay Attention to…

- **Distributed Systems:**
  - **Definition + Relevant Components**
  - **Motivation & Challenges**
  - **Logical and System Architectures**
    sub-categories, comparison

Questions?

Distributed Systems

José Oramas