Distributed Systems
Types and Architectures

Dr. Michael J. May Kinneret College

Topics for Today

- Types of Distributed Systems
  - Distributed Computing Systems
  - Distributed Information Systems
  - Pervasive Systems
- Architectural Styles
- System Architectures

Types of Distributed Systems

High performance distributed computing systems

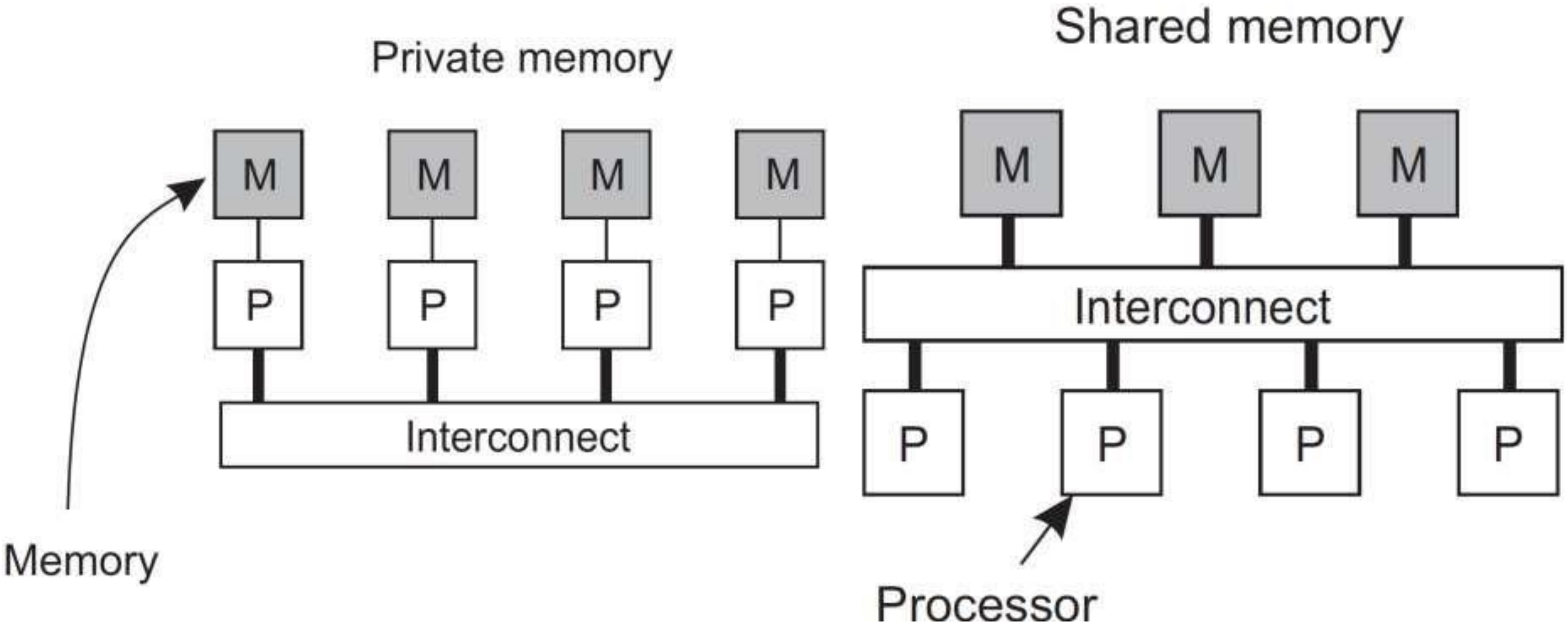
Distributed Information Systems

Distributed Pervasive Systems

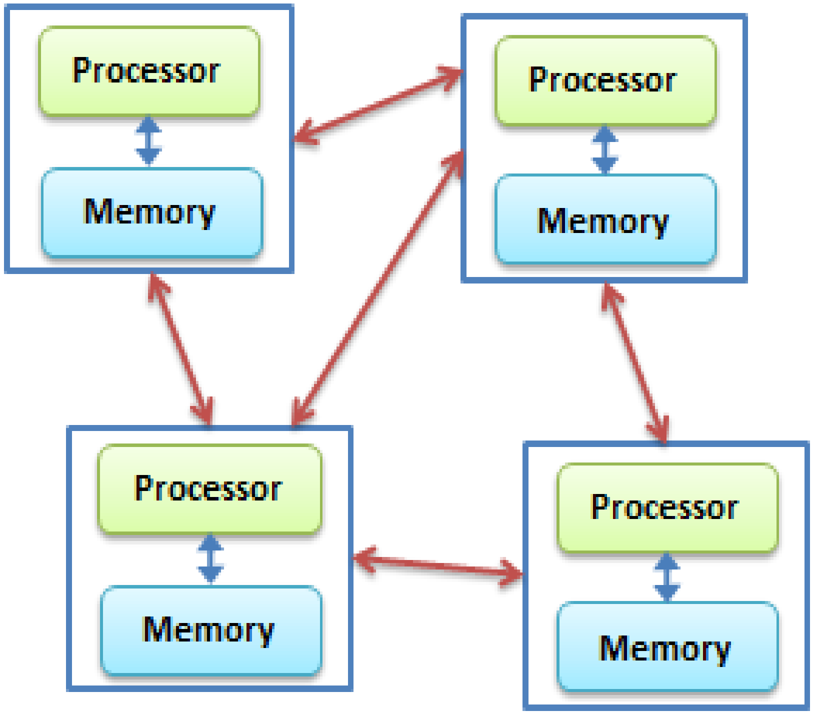
Parallel Computing

- High-performance distributed computing started with parallel computing

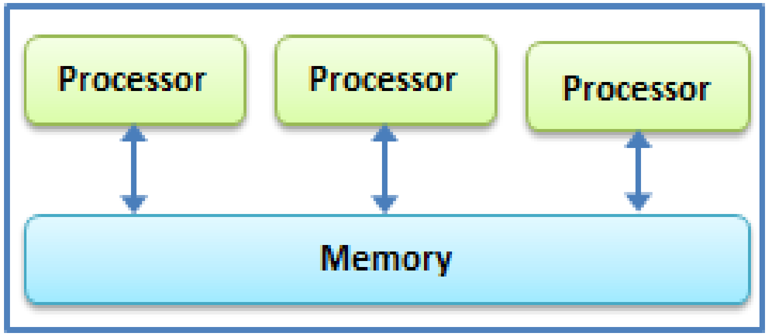
- Multiprocessor and Multicore versus Multicomputer



Distributed Computing



Parallel Computing



Distributed shared memory systems

Observation: Multiprocessors easier to program than multicomputers, but have problems when increasing the number of processors (or cores).

Solution: Implement a shared-memory model on top of a multicomputer

Example: Distributed Shared Memory (DSM)

1. Map all main-memory pages (from different processors) into one single virtual address space.
2. If process at processor A addresses page P located at processor B, the OS at A traps and fetches P from B, just as it would if P had been located on local disk.

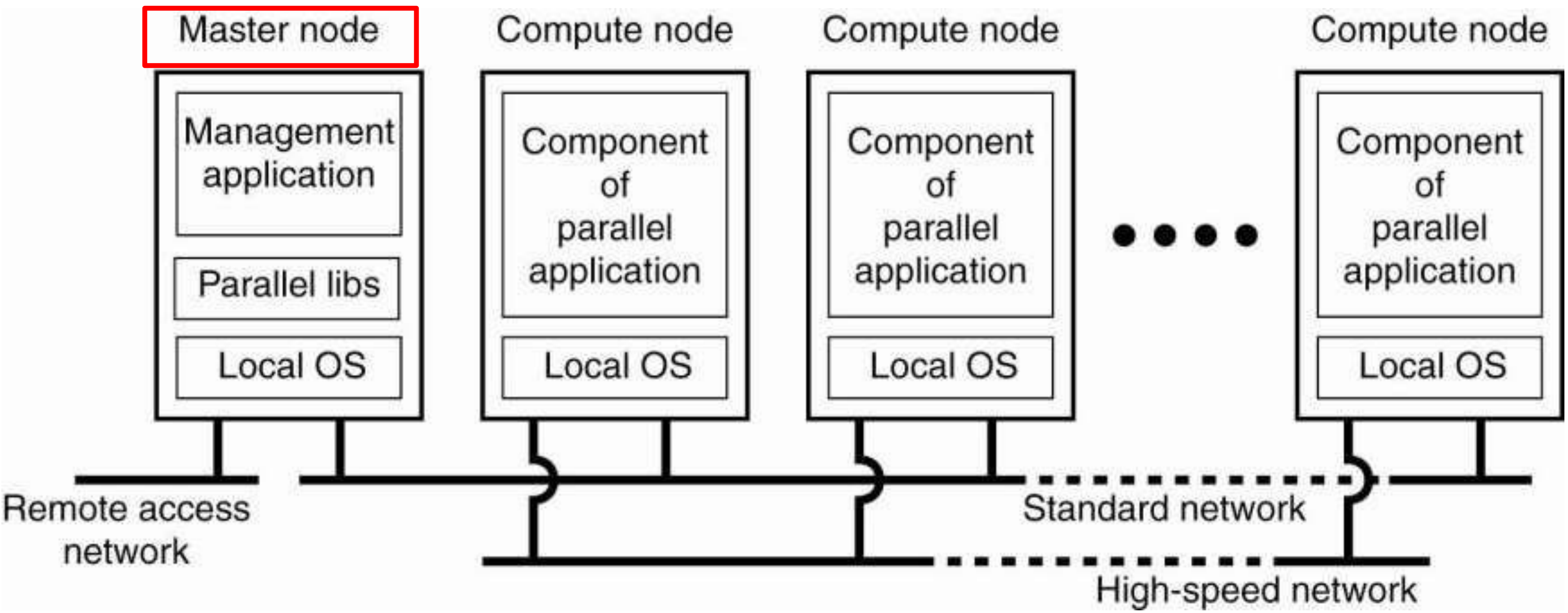
Problem:

- Performance of DSM could never compete with that of multiprocessors
- Failed to meet the expectations of programmers.
- Has been widely abandoned by now.

Cluster Computing

Essentially a group of high-end systems connected through a LAN

- Homogeneous: same OS, near-identical hardware
- Single managing node



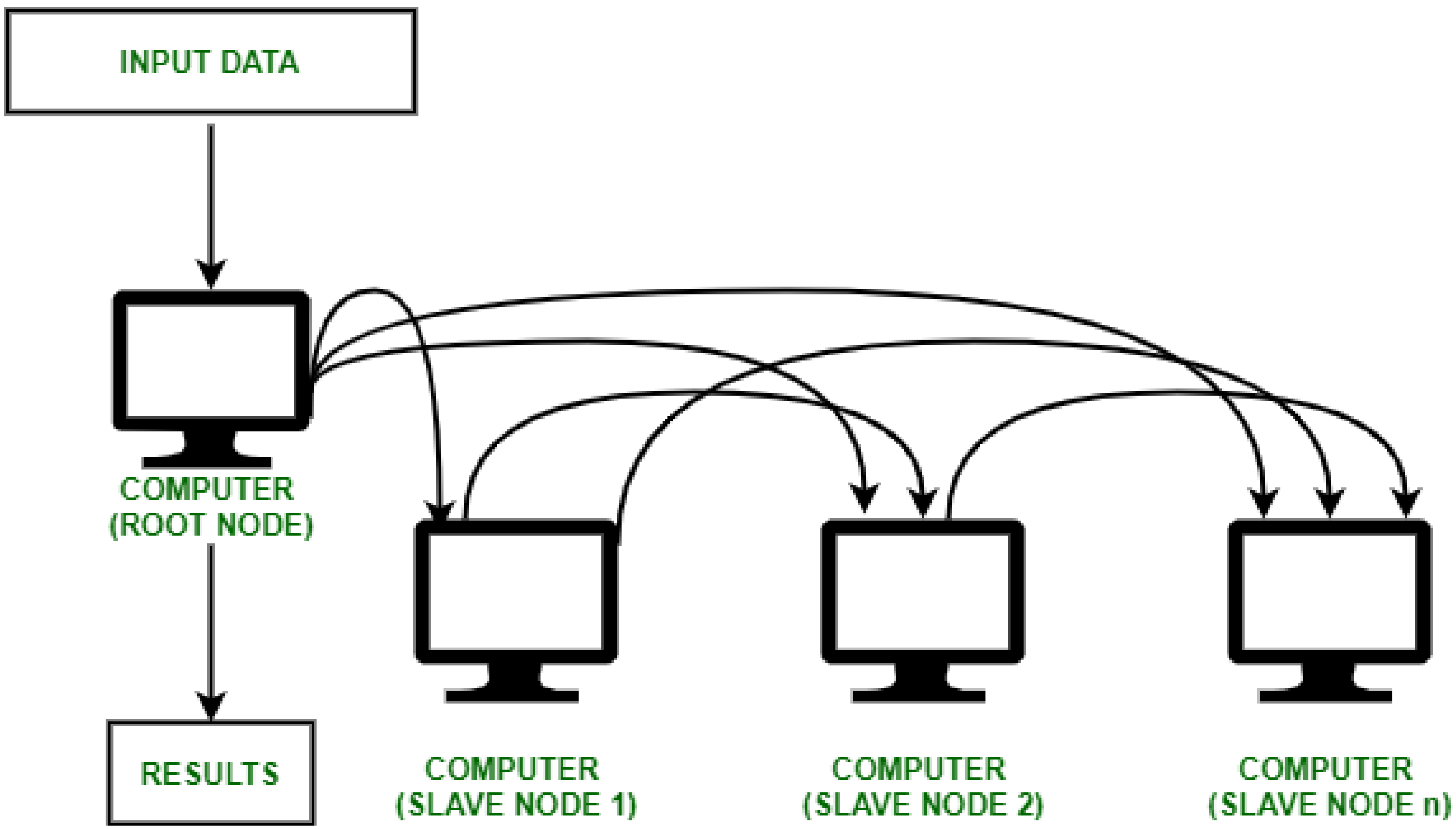


Photo by Massimo Botturi on Unsplash

Photo by imgix on Unsplash

Grid Computing

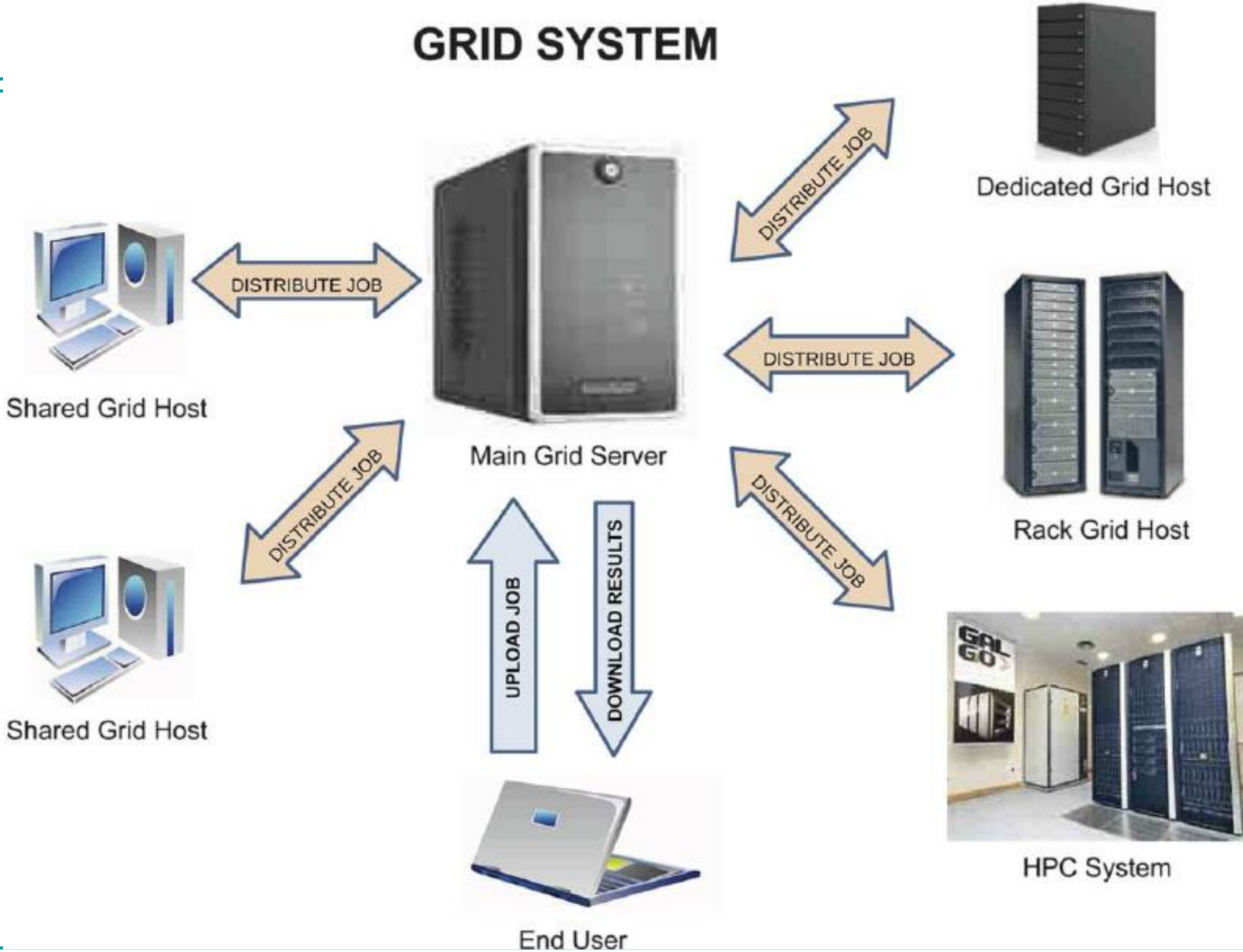
The next step: lots of nodes from everywhere:

- Heterogeneous
- Dispersed across several organizations
- Can easily span a wide-area network

Note

To allow for collaborations, grids generally use virtual organizations. In principle, this is a grouping of users (or better: their IDs) that will allow for authorization on resource allocation.

GRID SYSTEM


Dedicated Grid Host
DISTRIBUTE JOB
Shared Grid Host
DISTRIBUTE JOB
DISTRIBUTE JOB
Rack Grid Host
Main Grid Server
DISTRIBUTE JOB
DISTRIBUTE JOB
UPLOAD JOB
DOWNLOAD RESULTS
Shared Grid Host
HPC System
End User

Architecture for Grid Computing

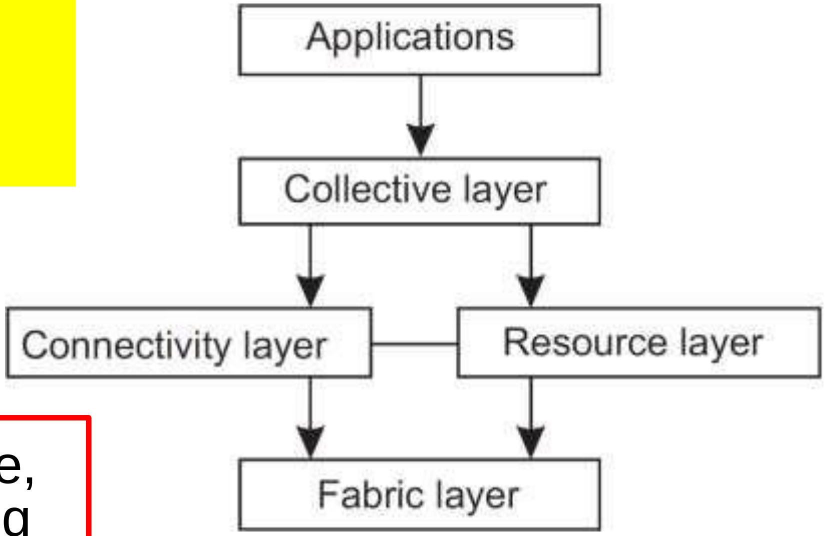
Application: Contains actual grid applications in a single organization.

Collective: Handles access to multiple resources: discovery, scheduling, replication.

Resource: Manages a single resource, such as creating processes or reading data.

Connectivity: Communication/transaction protocols, e.g., for moving data between resources. Also various authentication protocols.

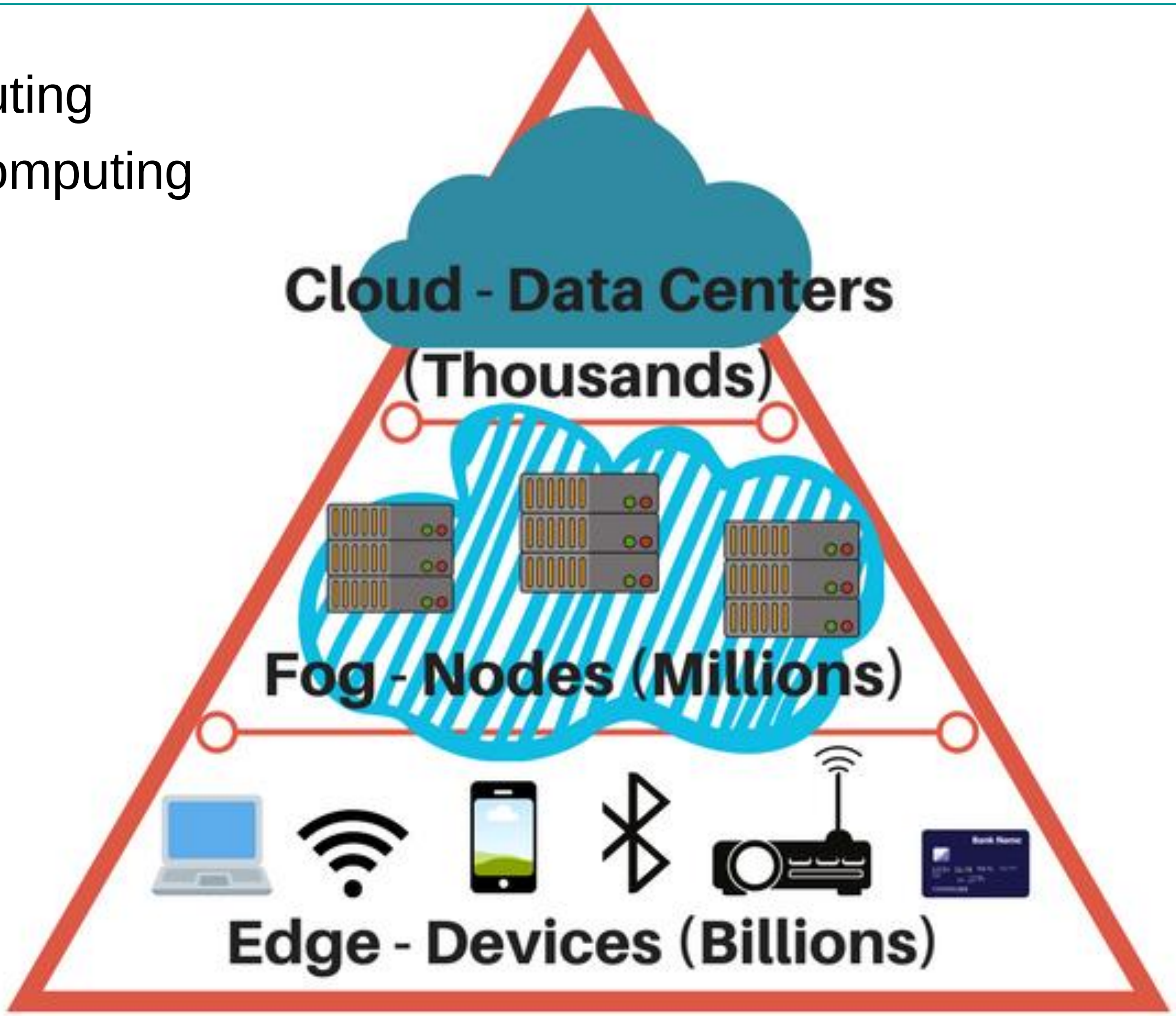
Fabric: Provides interfaces to local resources (for querying state and capabilities, locking, etc.)



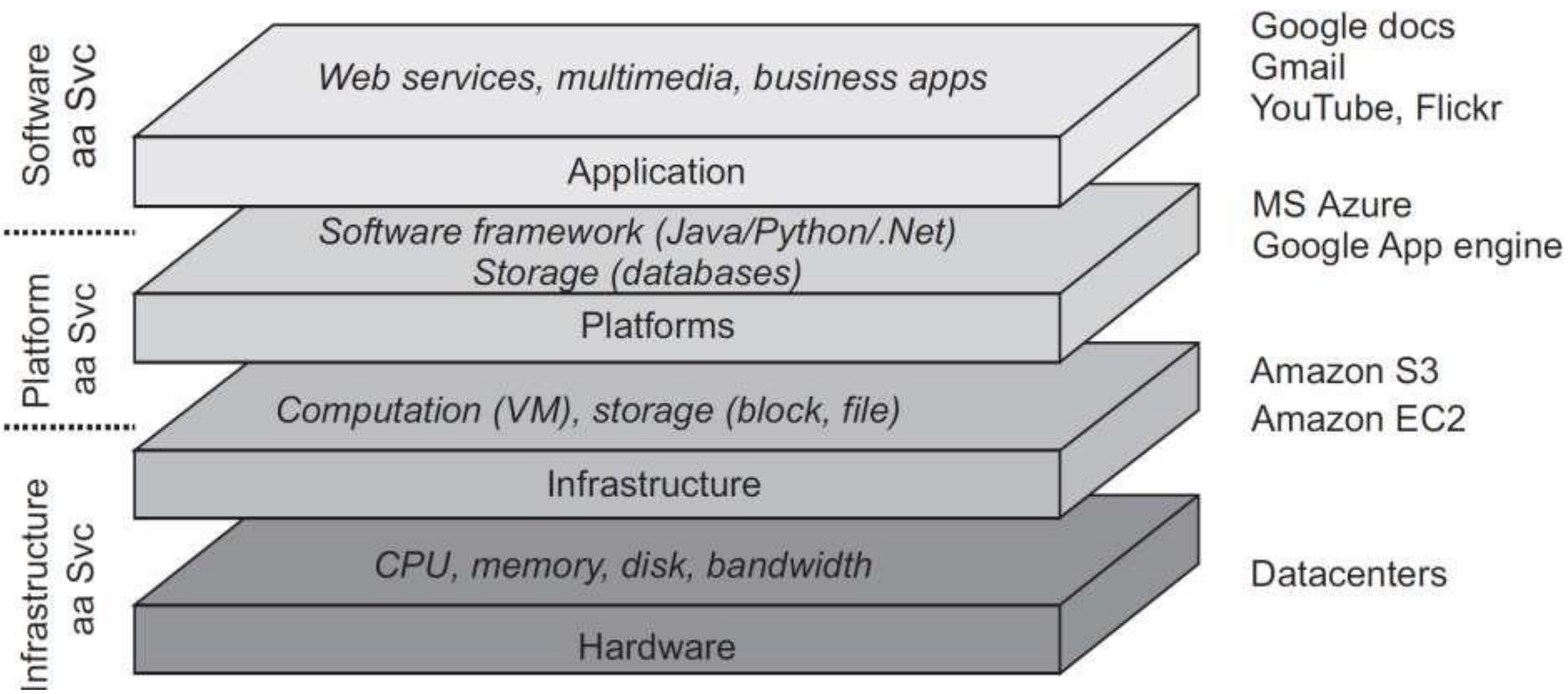
Grid's History

- Was a hit about 10 years ago
- Divided into
  - Cloud computing
  - Edge/Fog Computing

- Cloud computing
- Edge/Fog Computing



Cloud Computing



Parameter	Grid Computing	Cloud computing
Goal	Collaborative sharing of resources	Use of service (eliminates the detail)
Computational focuses	Computationally intensive	Operations Standard and high-level instances
Level of abstraction	Low (more details)	High (eliminate details)
Degree of scalability	Normal	High
Multitask	Yes	Yes
Transparency	Low	High
Time to run	Not real-time	Real-time services

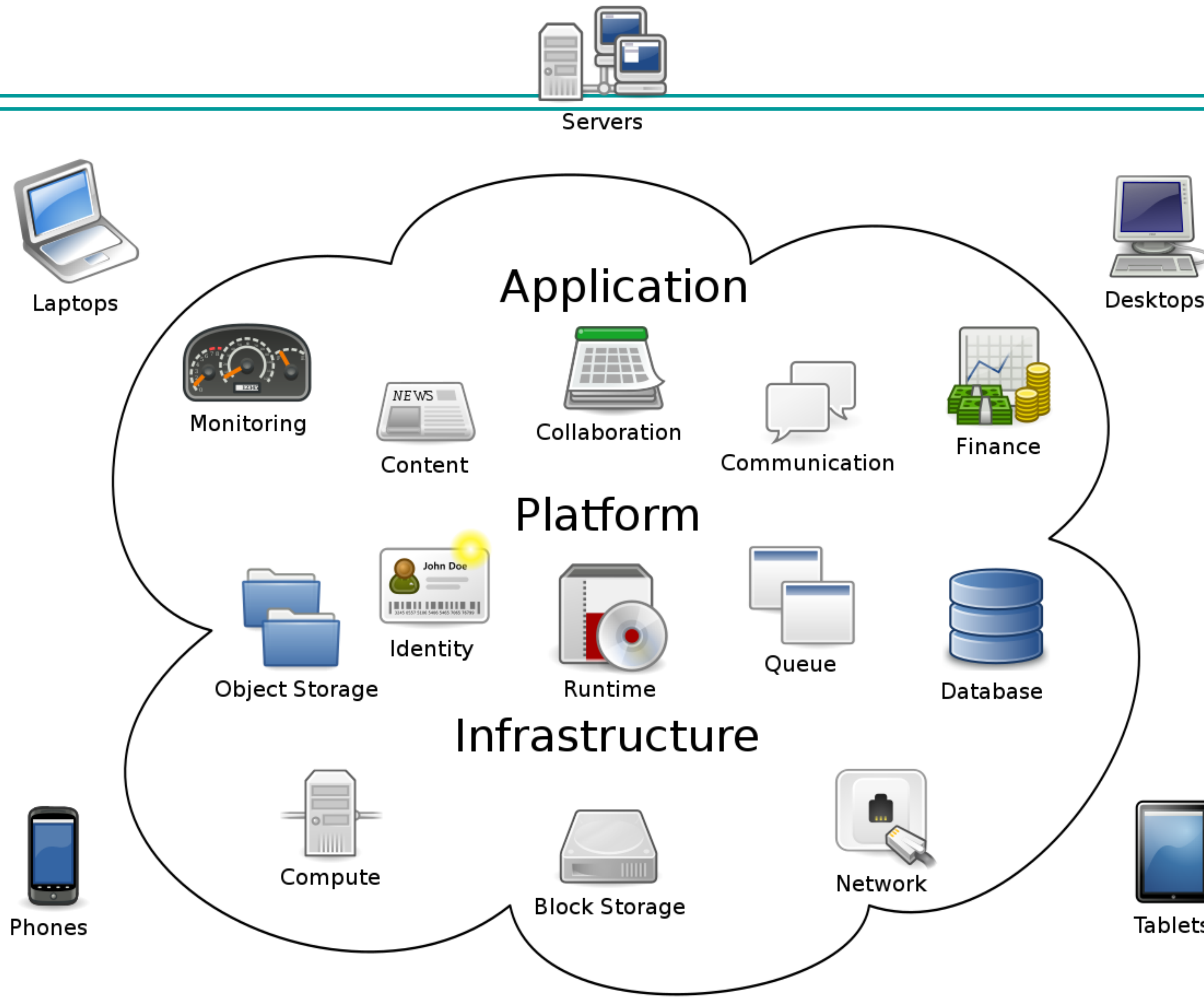
Clouds: Four Layers

1. Hardware: Processors, routers, power and cooling systems. Customers normally never get to see these.

2. Infrastructure: Deploys virtualization techniques. Evolves around allocating and managing virtual storage devices and virtual servers.

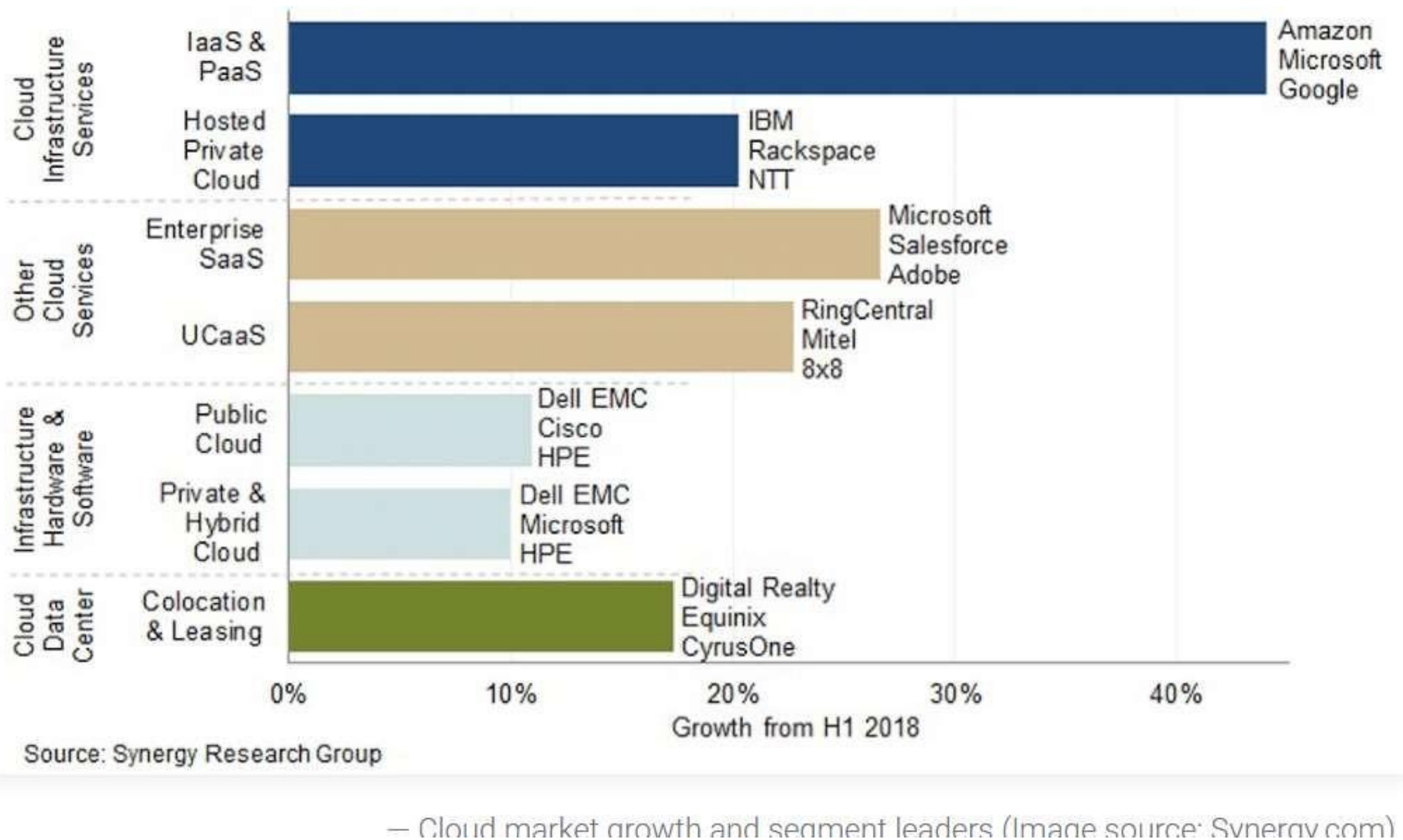
3. Platform: Provides higher-level abstractions for storage and such.
   - Example: *Amazon S3* storage system offers an API for (locally created) files to be organized and stored in "buckets".

4. Application: Actual applications, such as office suites (text processors, spreadsheet applications, presentation applications).
   - Comparable to the suite of apps shipped with OSes.


Servers
Laptops
Desktops
Application
Monitoring
Content
Collaboration
Communication
Finance
Platform
Identity
Object Storage
Runtime
Queue
Database
Infrastructure
Compute
Block Storage
Network
Phones
Tablets

It’s a big market

Image source: https://kinsta.com/blog/cloud-market-share/



Integrating applications

Situation: Organizations have many networked applications, but achieving inter-operability is painful.

Basic approach:

- A networked application is one that runs on a server making its services available to remote clients.
- Simple integration
  - Clients combine requests for (different) applications
  - Send them
  - Collect responses
  - Present a clear result to the user.

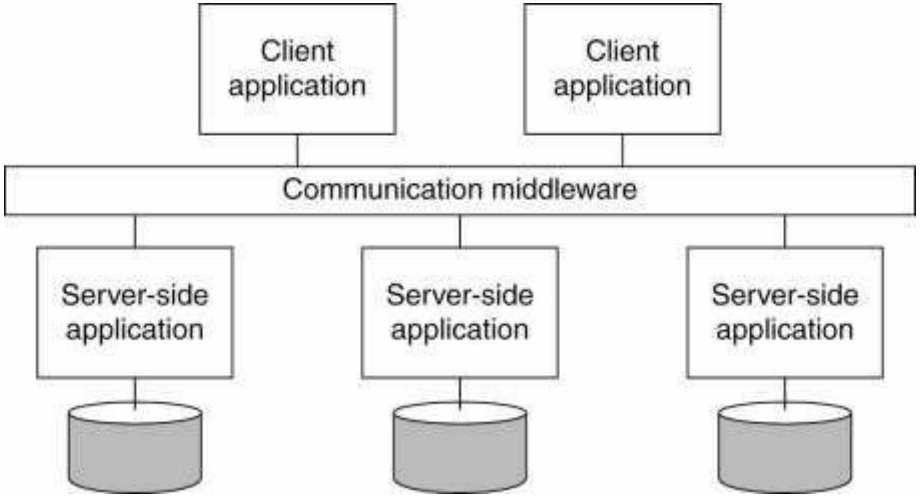
Next step

- Allow direct application-to-application communication
- ❼ Enterprise Application Integration

Middleware and EAI

Problem

A Transaction Processing monitor doesn't separate apps from their databases. Also needed are facilities for direct communication between apps.



- Remote Procedure Call (RPC)
- Message Oriented Middleware (MOM)

- Remote Procedure Call (RPC)

a software communication protocol that one program can use to request a service from a program located in another computer on a network without having to understand the network's details

- Message Oriented Middleware (MOM)

software or hardware infrastructure supporting sending and receiving messages between distributed systems

How to integrate applications

- File transfer: Technically simple, but not flexible:
  - Figure out file format and layout
  - Figure out file management
  - Update propagation, and update notifications.

- Shared database: Much more flexible, but still requires common data scheme next to risk of bottleneck (each application locks others out of the data (cause Deadlock))

- Remote procedure call: Effective when execution of a series of actions is needed.

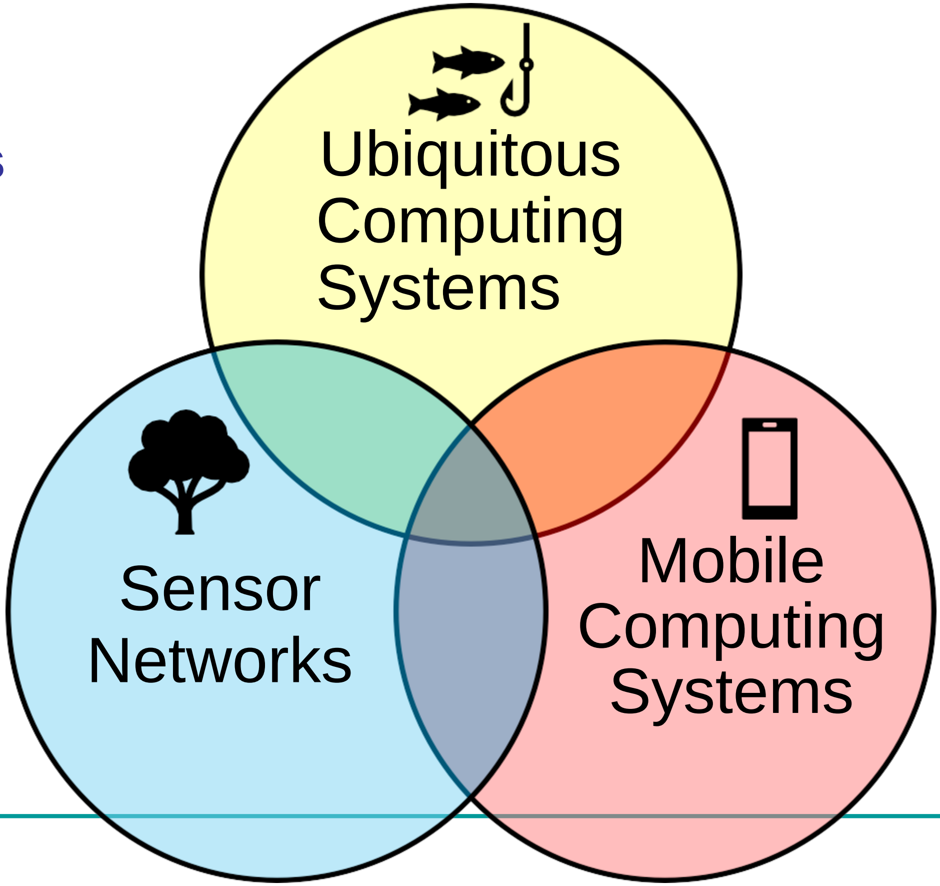
- Messaging: RPC uses the client-server model. The requesting program is a client, and the service-providing program is the server.

Distributed Pervasive Systems

Observation

Emerging next-generation of distributed systems in which nodes are small, mobile, and often embedded in a larger system characterized by the fact that the system naturally merge into the user's environment.

Three (overlapping) subtypes



Ubiquitous Characteristics

Distribution

- Devices are networked, distributed, and accessible in a transparent manner

Interaction

- Interaction between users and devices is highly self-wipe-out

Context awareness

- The system is aware of a user's context in order to optimize interaction

Autonomy

- Devices operate autonomously without human intervention, and are thus highly self- managed

Intelligence

- The system as a whole can handle a wide range of dynamic actions and interactions

Mobile Computing Systems

Distinctive features

Different numerous mobile devices ❼ smartphones, tablets, GPS devices, remote controls, active badges

A device's location changes over time ❼ change of local services, reachability, etc.

Keyword: discovery.

Communication may become more difficult: no stable route, also no guaranteed connectivity ❼ disruption tolerant networking (lack of connectivity, resulting in a lack of instantaneous end-to-end path)

Keyword: Offline support

Sensor Nets Characteristics

The nodes to which sensors are attached are:

Many

- 10s-1000s

Simple

- Small memory, compute, communication capacity

Often battery-powered

- Or even battery-less
- Solar powered

Airflow Sensors

Contain advanced microstructure technology to provide a sensitive & fast response to flow, amount/direction of air or other gases.

Current Sensors

Accurate & fast response for power management. Series includes adjustable linear, null balance, digital, & linear current sensors.

Carbon Dioxide (CO_2) Sensors

Non-dispersive infrared (NDIR) CO_2 sensors for use in potential HVAC, indoor air quality measurement, & purification system applications.

Force Sensors

PCB sensors measure the addition or backup of force, with proportional output.

Humidity, Thermal & Flexible Heater Products

A wide variety of humidity sensors & wall mount transducers, thermal sensing elements, thermostats/thermal switches, & flexible heaters.

Inertial Measurement Units

Provide motion, position, & navigational sensing from a durable single device over six degrees of freedom via MEMS technology.

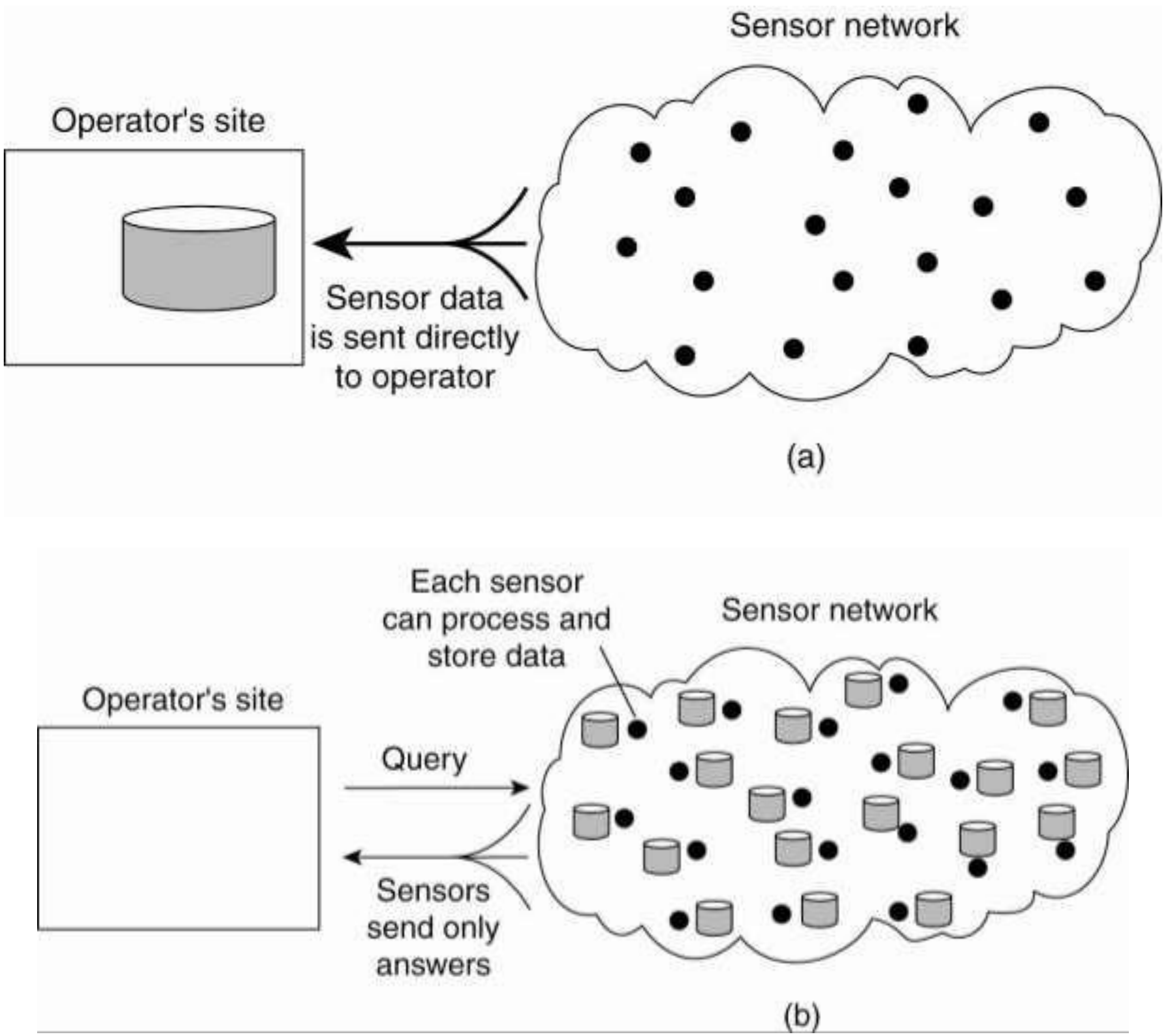
Magnetic Sensor ICs & Value-added Packages

Hall-effect or anisotropic magnetoresistive (AMR) sensor ICs in digital or linear outputs for angle, position & speed sensing; value-added packages.

Motion & Position Sensors

Encoders, angular/rotary potentiometers, non-contact Hall-effect rotary position sensors, resolvers, torque, & accelerometers.

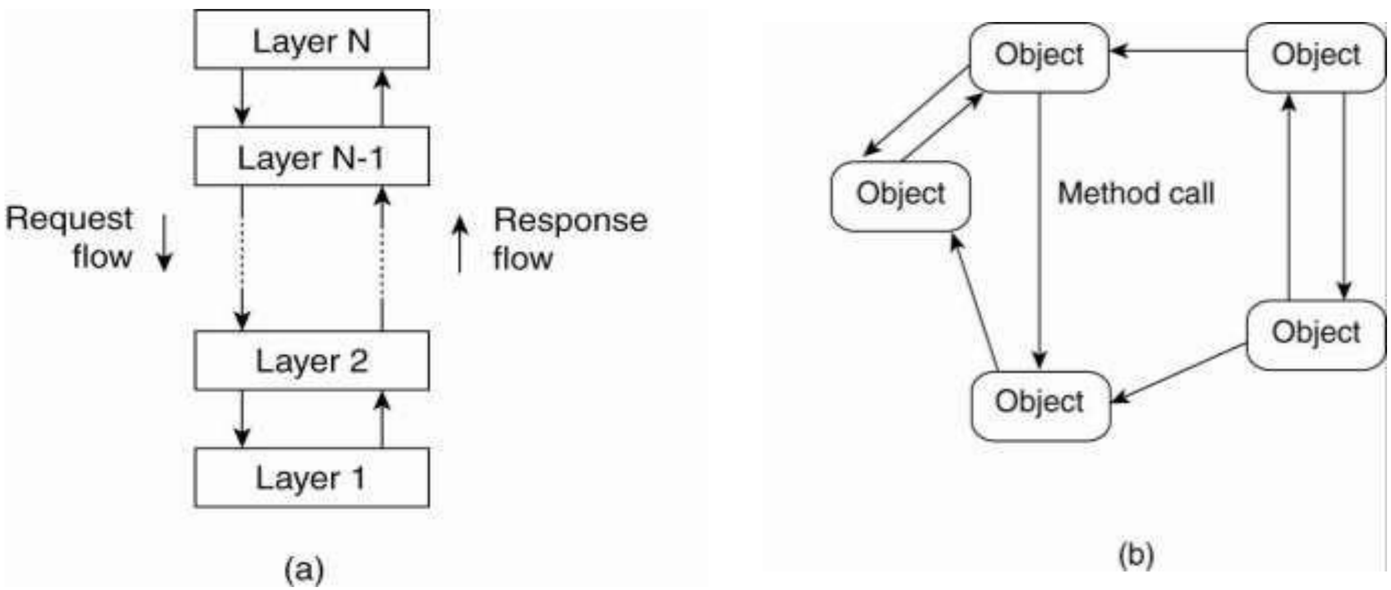
Sensor networks as distributed databases



Architectural Styles

Basic Idea

Organize into logically different components and distribute those components over the various machines



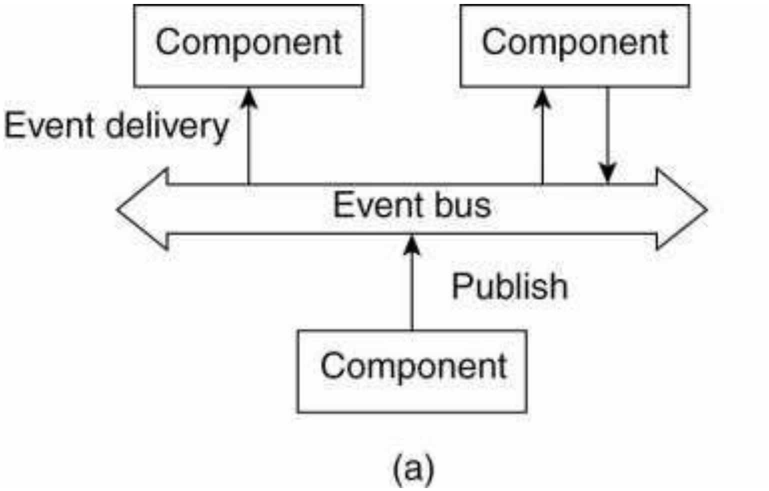
(a) Layered style is used for client-server system

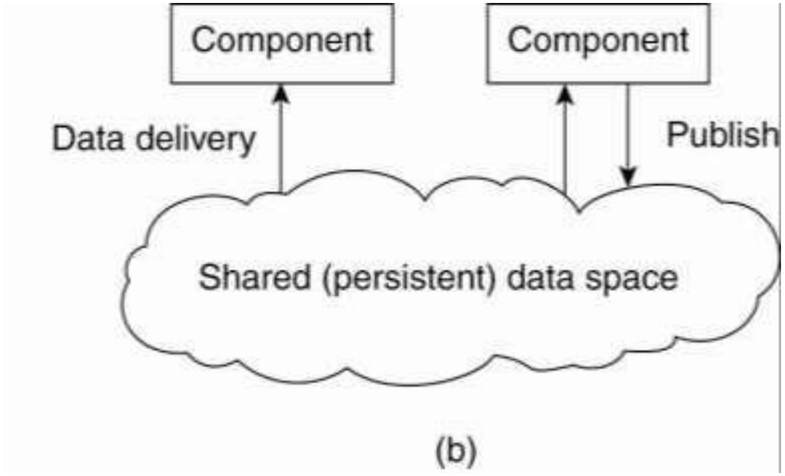
(b) Object-based style for distributed object systems

Architectural Styles

Observation

Decoupling processes (“anonymous”) and removing time constraints (“asynchronous”) led to alternative styles.





(a) Publish/Subscribe [anonymous]

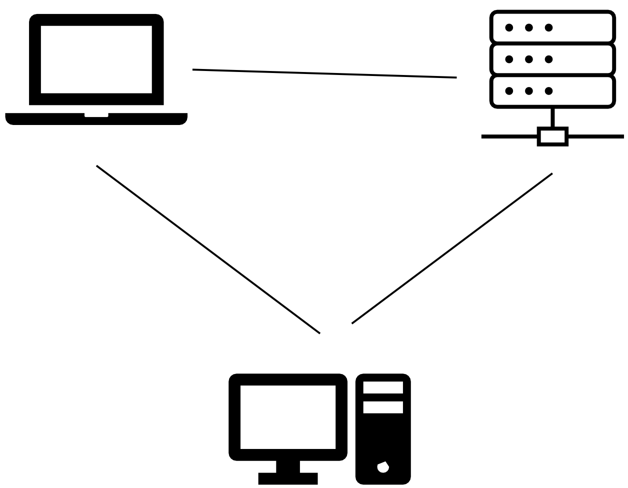
(b) Shared dataspace [anonymous and asynchronous]

Memory mapping: map processes to anonymous memory regions that may be shared by cooperating processes

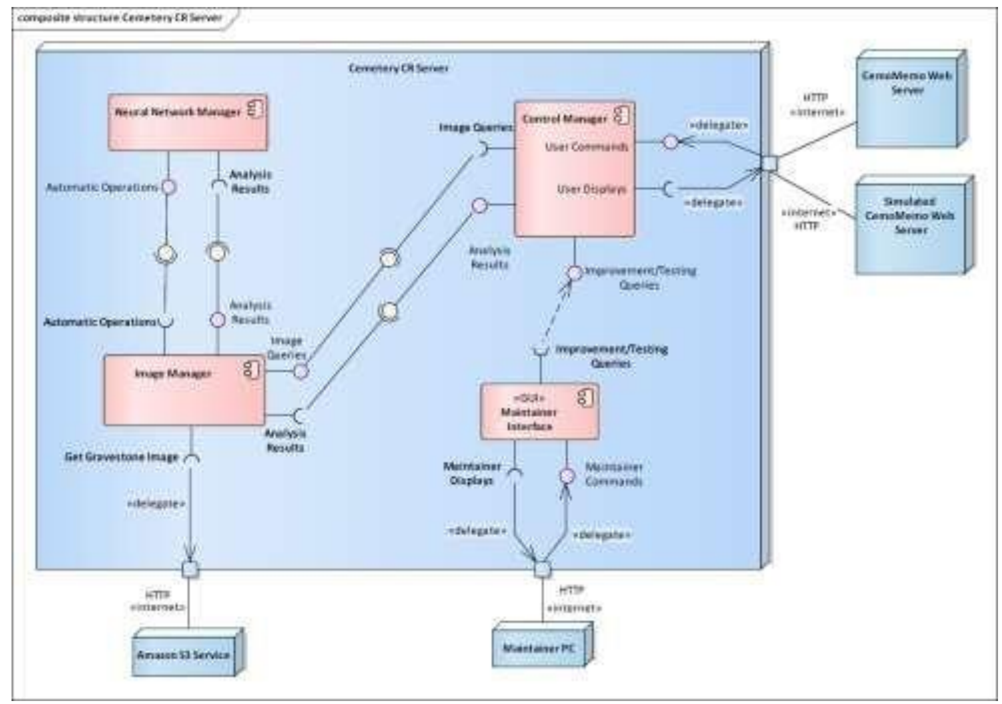
Discern or Recognize

Physical architecture

- How many computers?
- Where they are located?



Logical architecture & Deployment

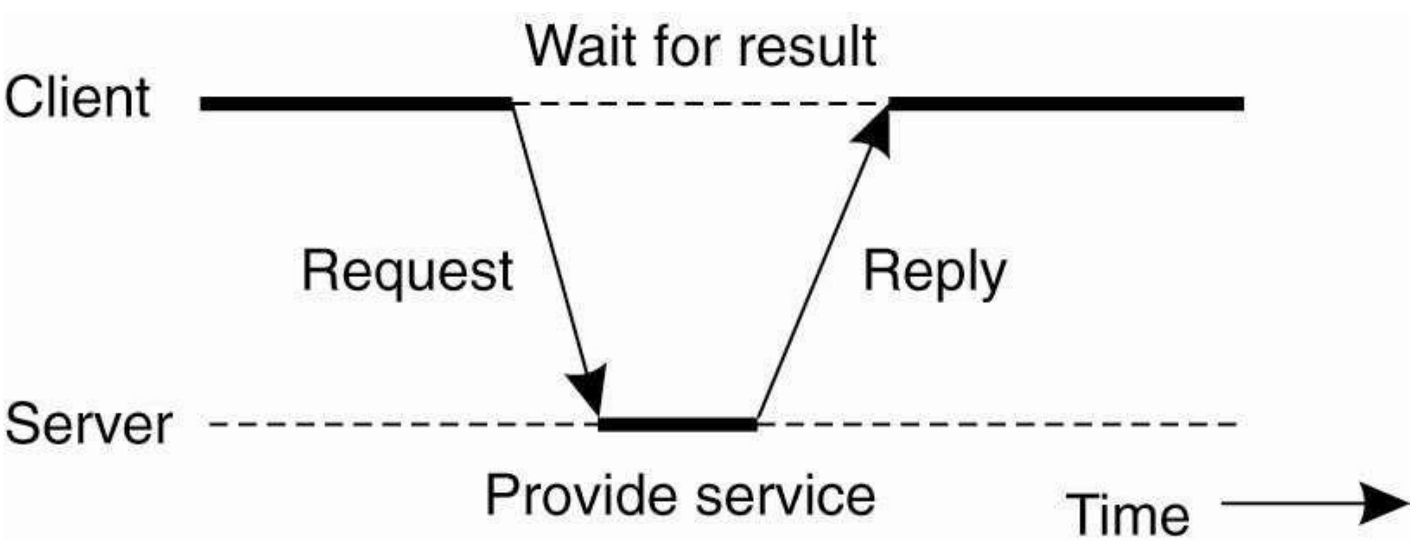


Physical: Centralized Architectures

Basic Client-Server Model

Characteristics:

- There are processes offering services (servers)
- There are processes that use services (clients)
- Clients and servers can be on different machines
- Clients follow request/reply model with respect to using services



Logical: Application Layering

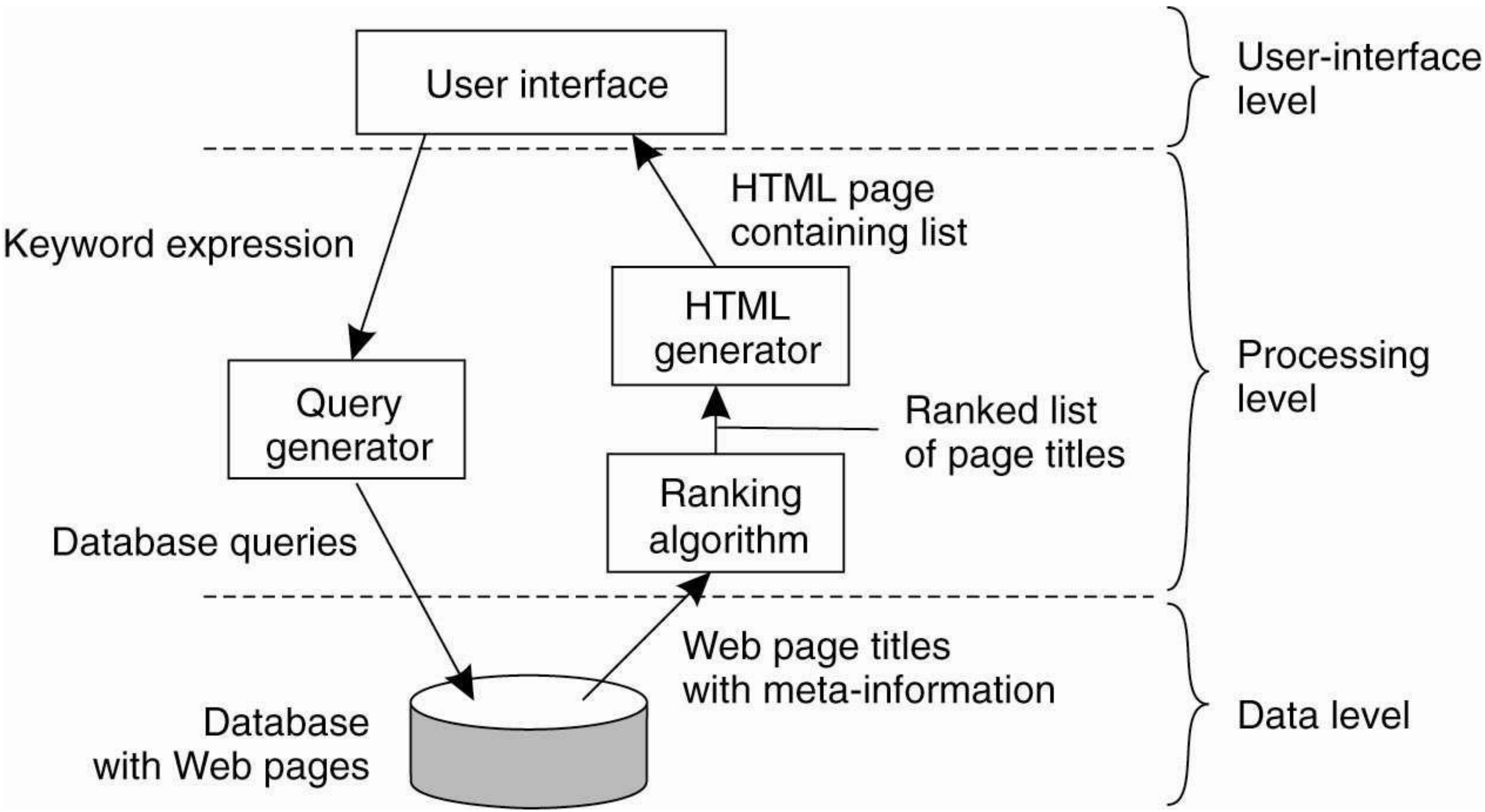
Traditional three-layered view

- User-interface layer contains units for an application's user interface
- Processing layer contains the functions of an application, i.e. without specific data
- Data layer contains the data that a client wants to manipulate through the application components

Observation

This layering is found in many distributed information systems, using traditional database technology and accompanying applications

Application Layering



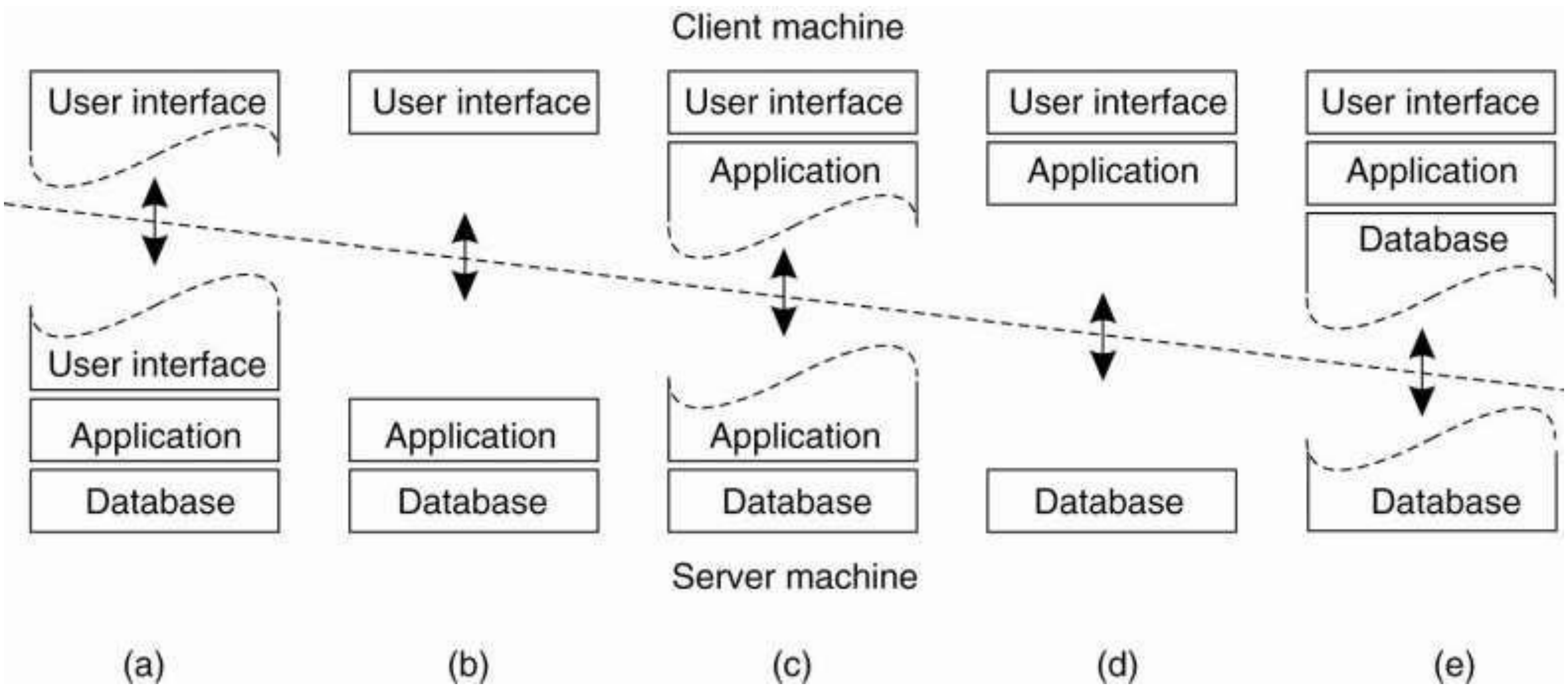
Multi-Tiered Architectures

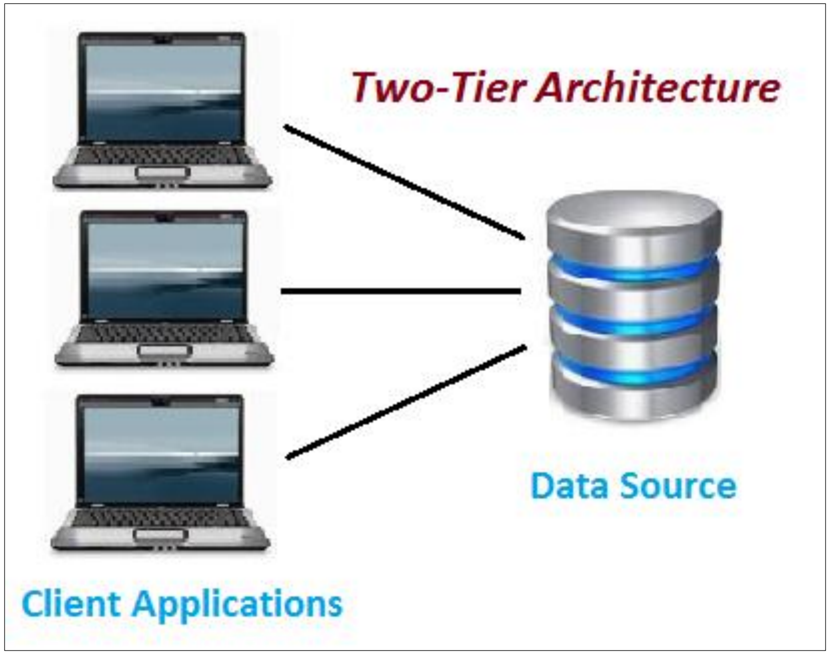
Single-tiered: dumb terminal/mainframe configuration

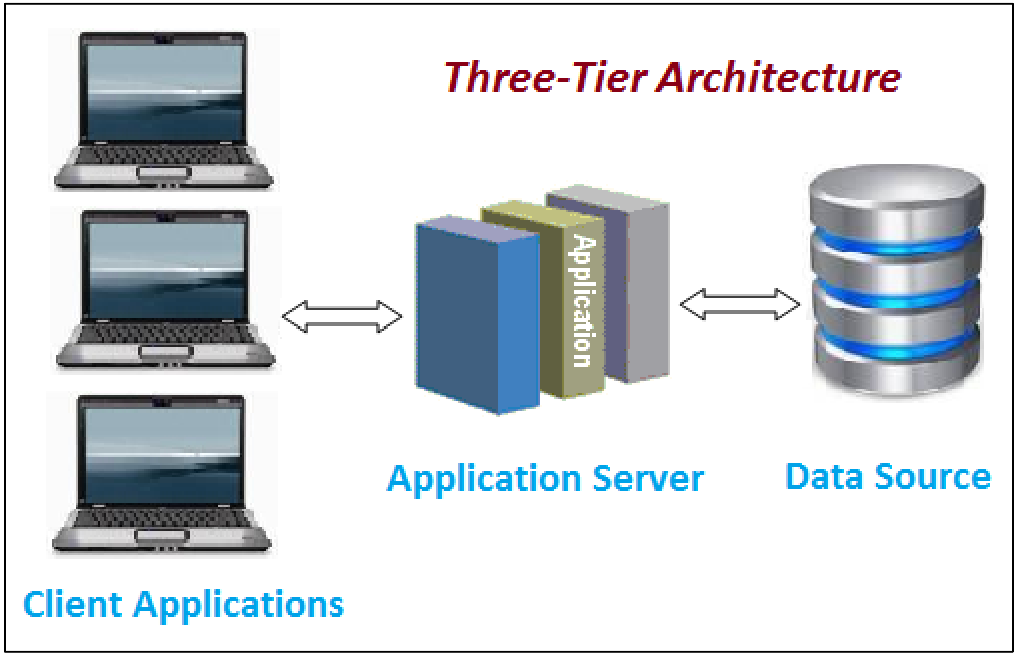
Two-tiered: client/single server configuration

Three-tiered: each layer on a separate machine

Traditional two-tiered configurations:




Two-Tier Architecture
Data Source
Client Applications


Three-Tier Architecture
Application
Application Server
Data Source
Client Applications

Physical: Decentralized Architecture

Observation

In the last couple of years we have been seeing a tremendous growth in peer-to-peer systems

- Structured P2P: nodes are organized following a specific distributed data structure
- Unstructured P2P: nodes have randomly selected neighbors
- Hybrid P2P: some nodes are appointed special functions in a well-organized fashion

---

Note

In virtually all cases, we are dealing with overlay networks: data is routed over connections setup between the nodes (via application level multicasting)

END

- Types of Distributed Systems
  - Distributed Computing Systems
  - Distributed Information Systems
  - Pervasive Systems
- Architectural Styles
- System Architectures