Architectures

ECE 454 / 751: Distributed Computing

Instructor: Dr. Wojciech Golab

wgolab@uwaterloo.ca

Slides are derived from A. S. Tanenbaum and M. Van Steen,
Distributed Systems: Principles and Paradigms, 2nd Edition, Pearson-Prentice Hall, 2006
as well as
M. Van Steen and A. S. Tanenbaum, Distributed Systems, 3rd Edition, Pearson, 2017.

A few definitions

service

Component: a modular unit with well-defined interfaces.

Connector: mechanism that mediates communication, coordination, or cooperation among components.

more detail

Software architecture: organization of software components.

System architecture: instantiation of software architecture in which software components are placed on real machines.

Autonomic system: adapts to its environment by monitoring its own behavior and reacting accordingly.

Architectural Styles

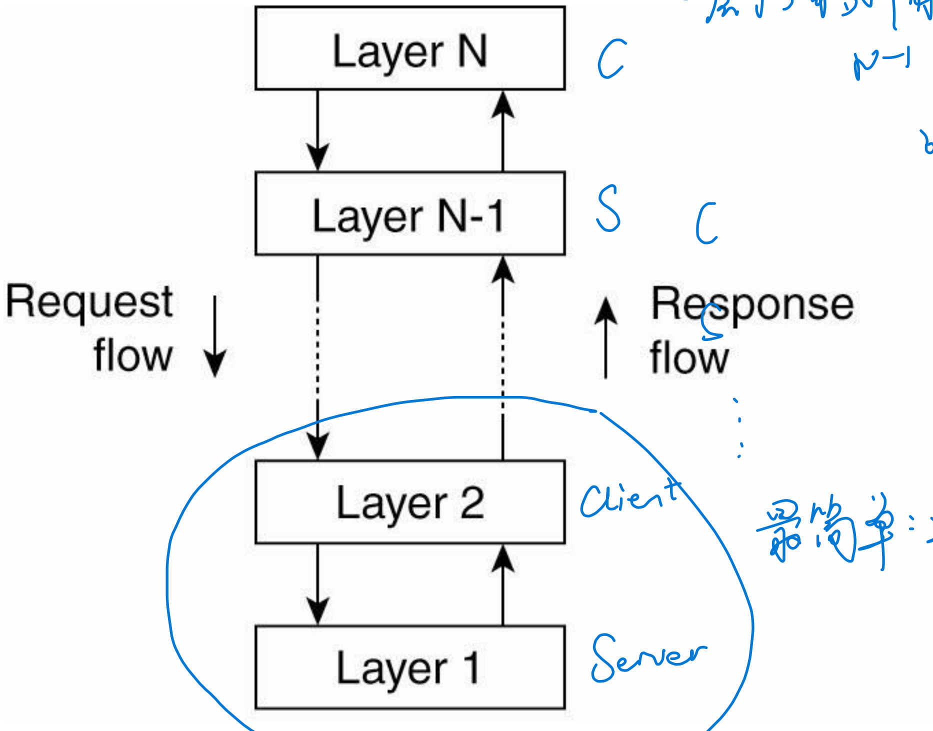
Many distributed software systems conform to one of the following architectural styles:

- layered
- object-based
- data-centered
- event-based

① Layered architecture

分层模式

In a layered architecture, control flows from layer to layer: requests flow down the hierarchy and responses flow upward.



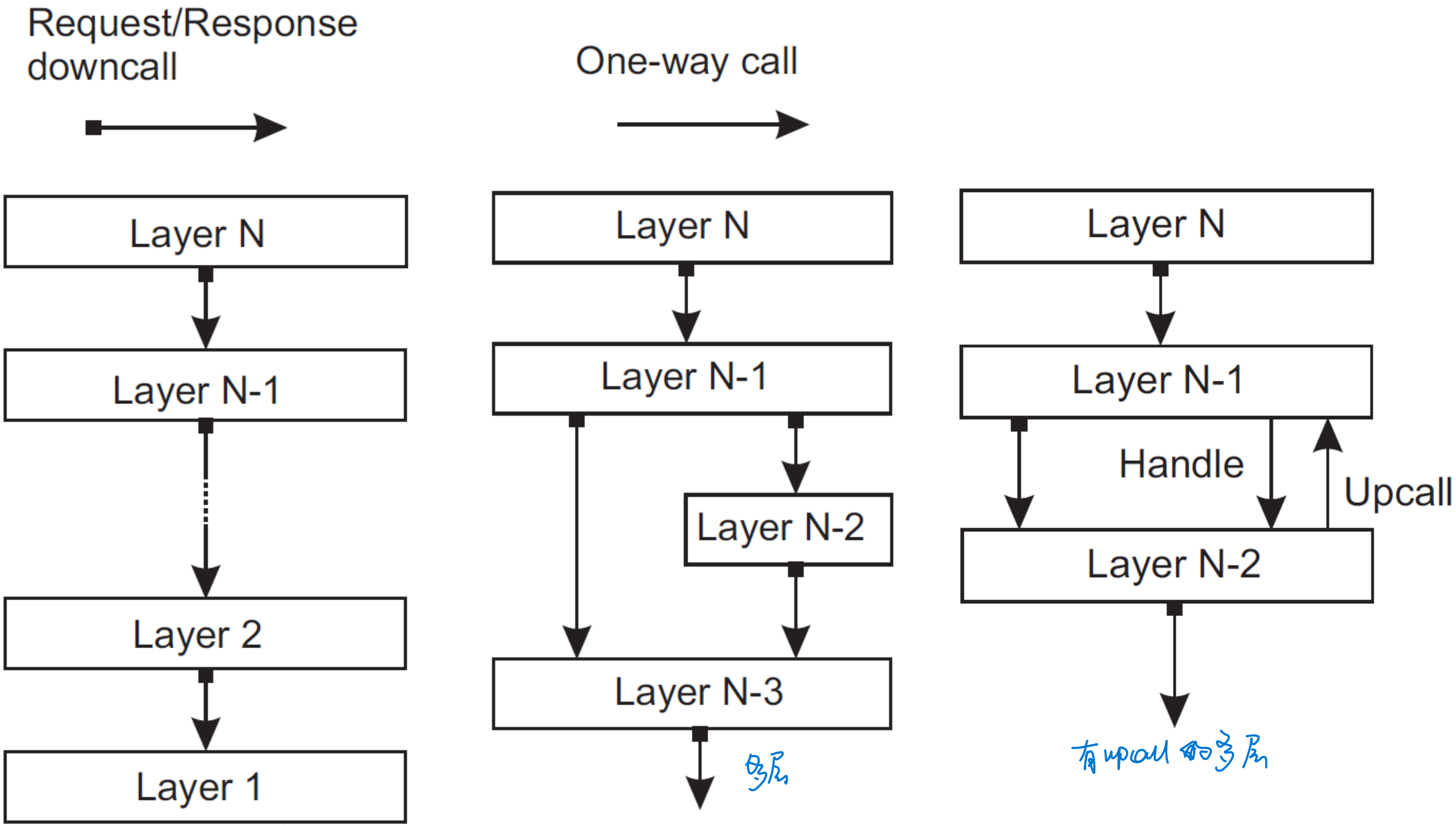
一层可以有多种角色

N-1 是N的Server

but 是 N-2的 client.

最简单：2层

Variations on layers



单层

多层

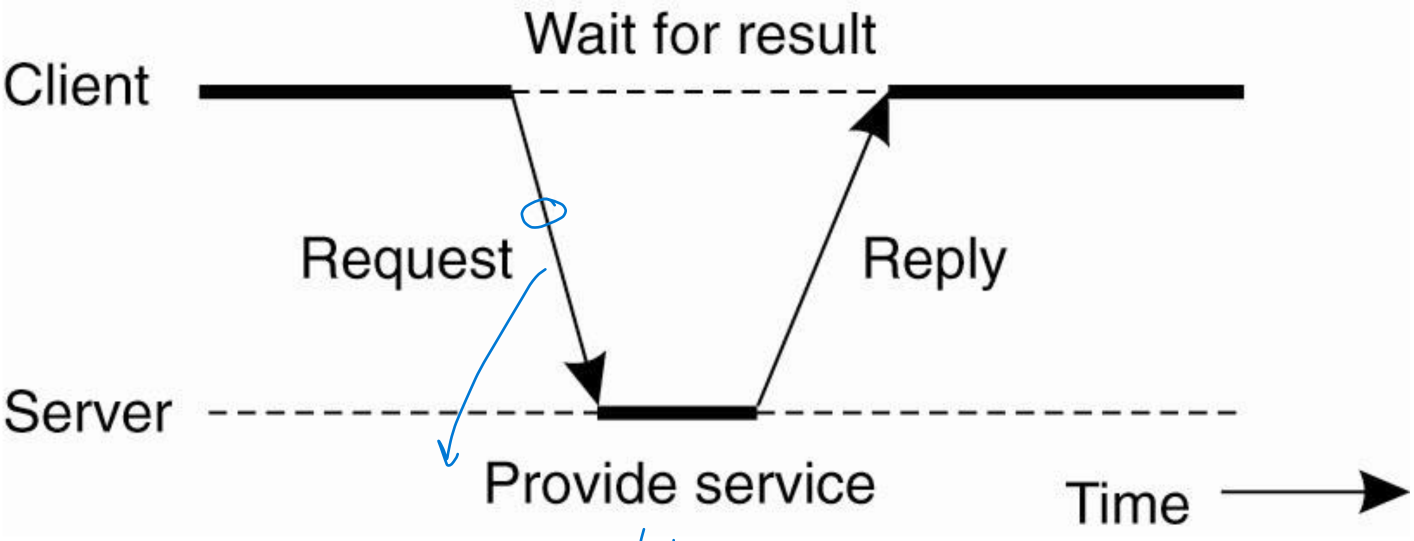
有upcall的多层

Figure 2.1: (a) Pure layered organization. (b) Mixed layered organization. (c) Layered organization with upcalls (adopted from [Krakowiak, 2009]).

Client-server interactions

客户端一般为这模式

Interactions among components often follow a client-server pattern in which one component (client) requests a service from another component (server) and waits for a response.

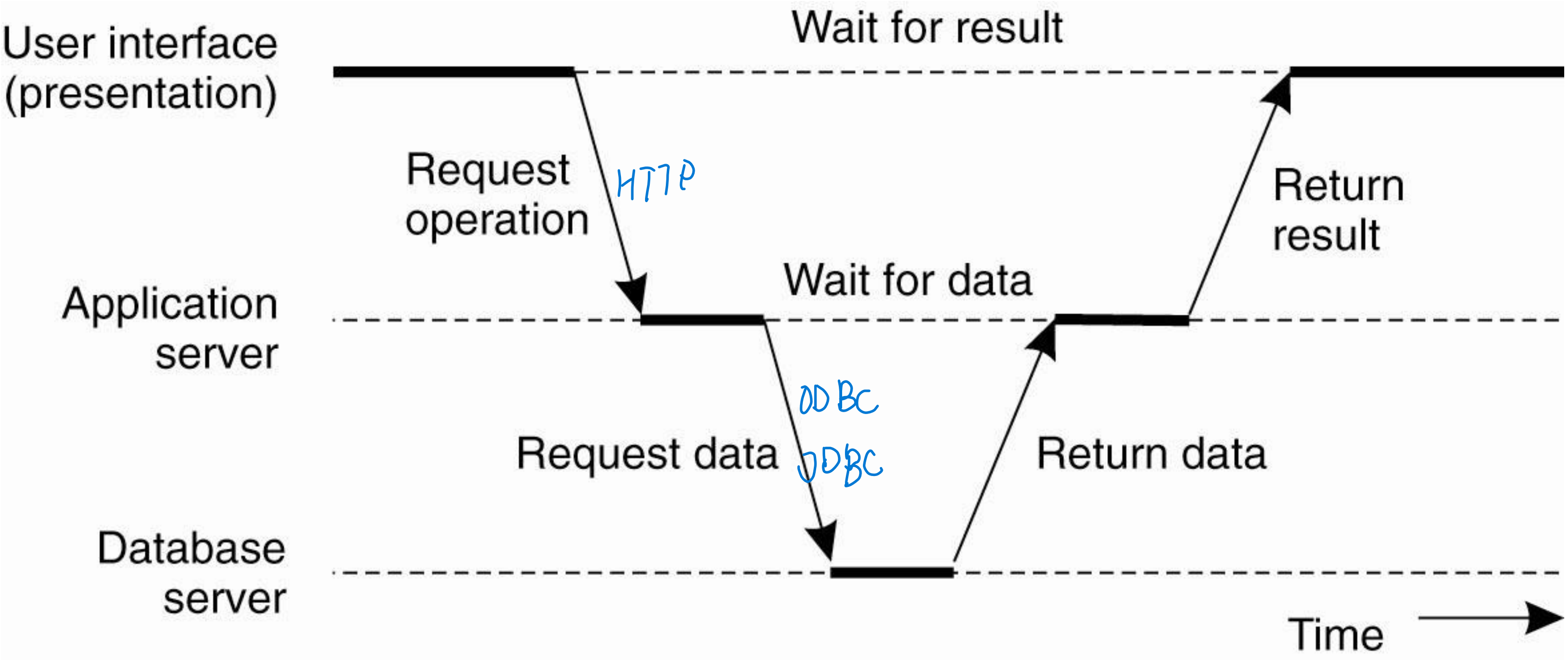


there is some latency.

不是垂直的

Application layering

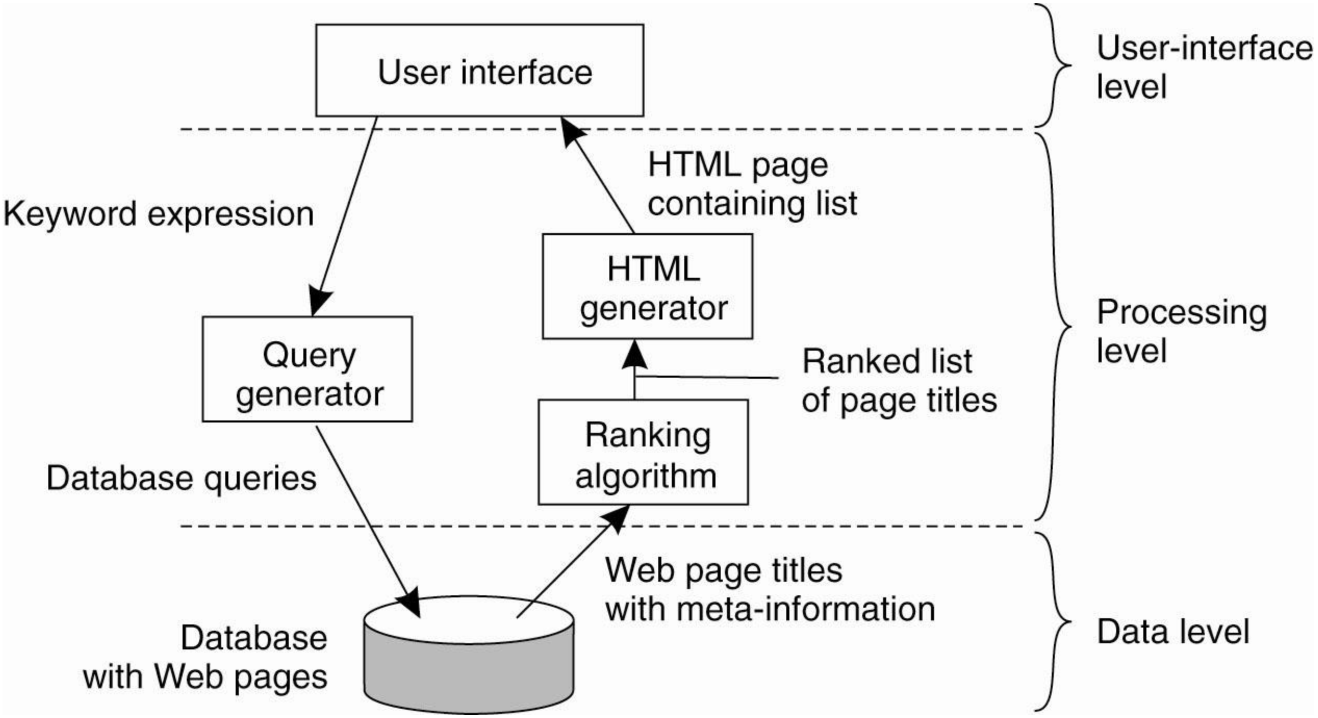
Many enterprise systems are organized into three layers: user interface, application server, and database. The middle layer acts as both a client and a server to the others.



Application layering

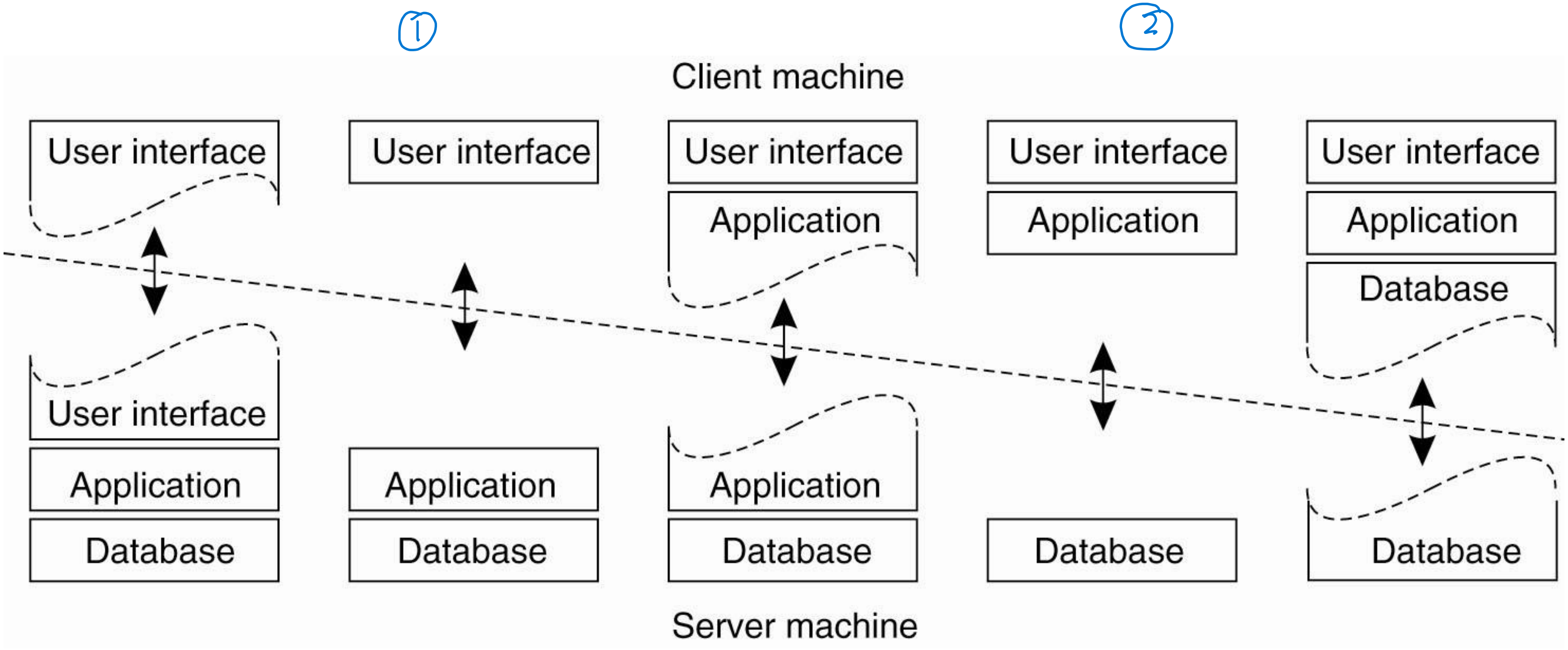
old form

A (grossly simplified) Internet search engine, illustrated below, can also be modelled as a three-layer system.



Multi-tiered architectures

Logical software layers must be mapped onto physical **tiers**. A two-tiered architecture comprises client machines and server machines only, leading to several alternative mappings.



Horizontal vs. vertical distribution

A0 **Vertical distribution:** when the logical layers of a system are organized as separate physical tiers. Example: use separate machines for the application server and database.

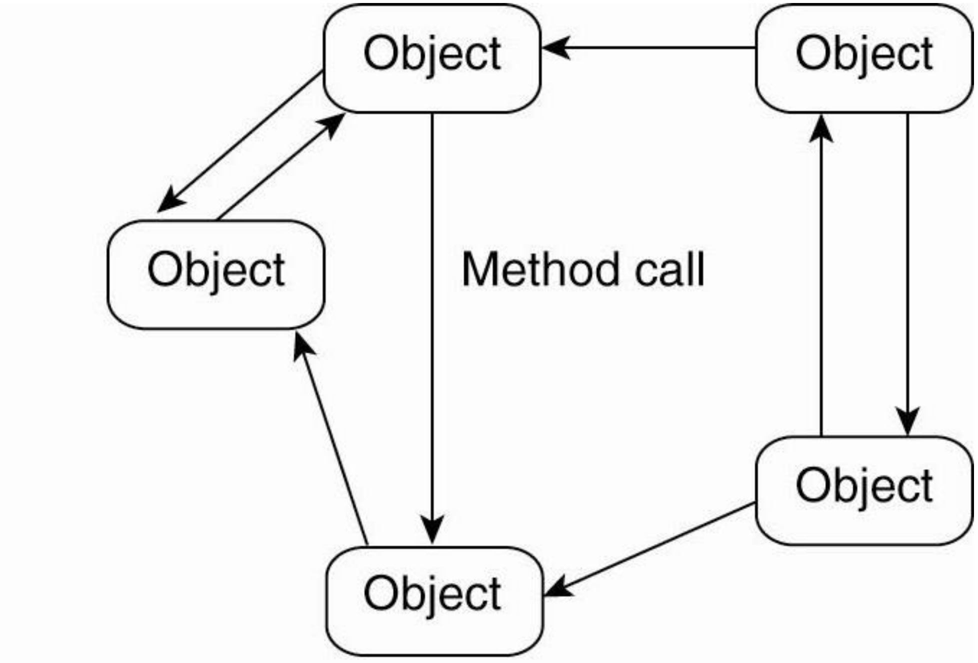
A1 **Horizontal distribution:** when one logical layer is split across multiple machines. Example: a data set is hash-partitioned across multiple independent database instances running on separate machines (also known as **sharding**).

同一层附

Food for thought: how do these two types of distribution affect performance, scalability, and dependability?

Object-based architecture

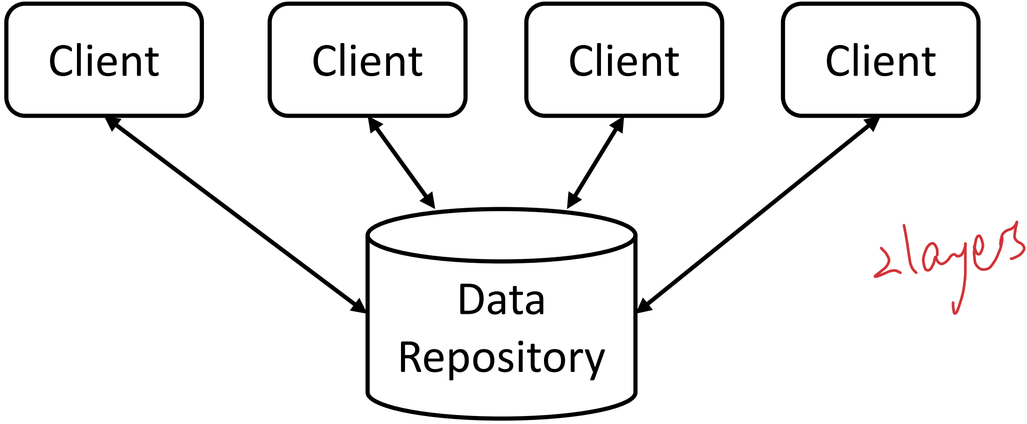
Components are more loosely organized in an object-based architecture. APIs such as Java remote method invocation (RMI) allow remote object references and method calls.



Data-centered architecture

In a data-centered architecture, components communicate by accessing a shared data repository such as a database, storage system, or file system. Web applications often incorporate this pattern.

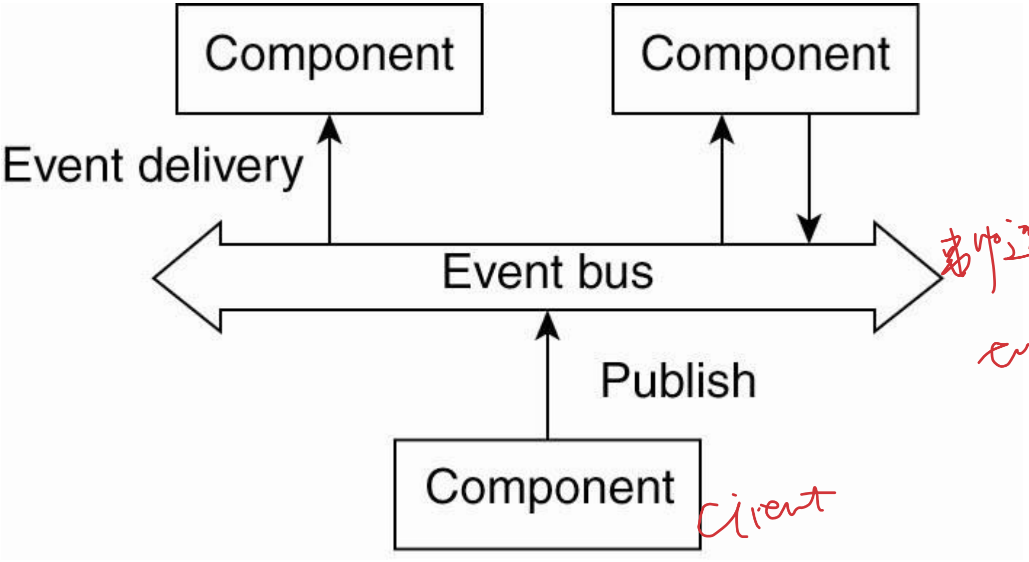
reading & writing data



2 layers.

Event-based architecture

In event-based architectures, components communicate by propagating events. **Publish/subscribe systems** can be used for sharing news, balancing workloads, refreshing distributed caches, event logging, and asynchronous workflows.



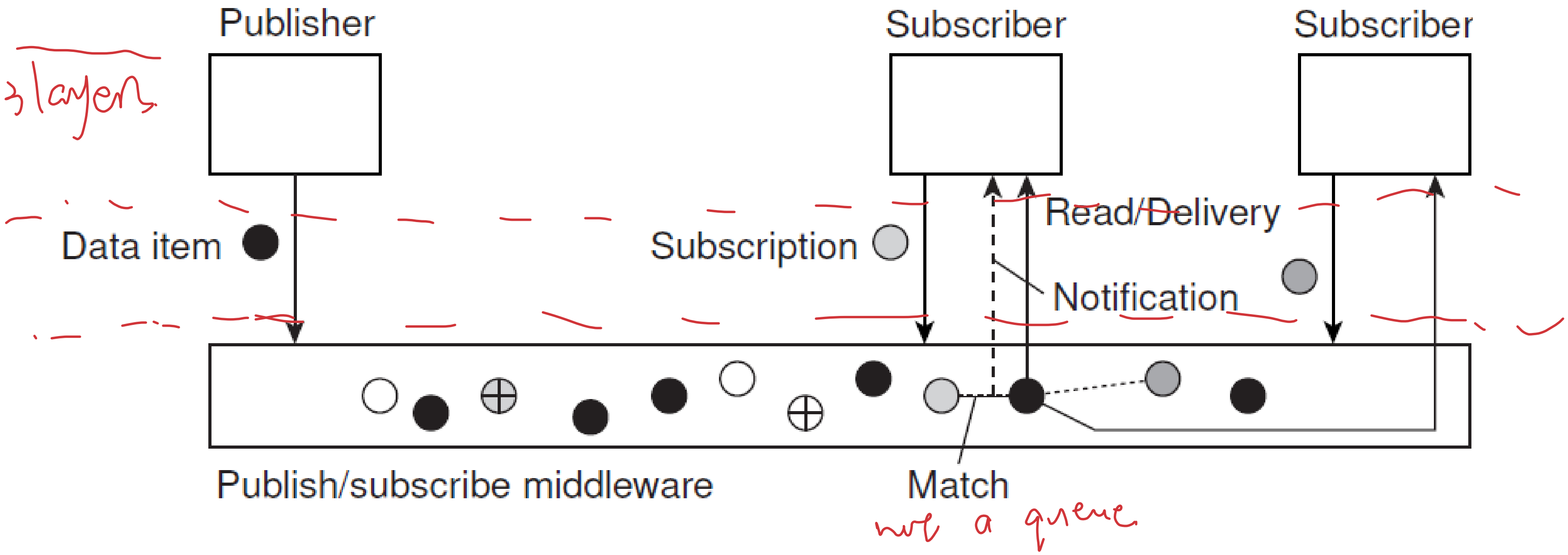
Client

都运行 动作很安全

twitter.

Event-based architecture

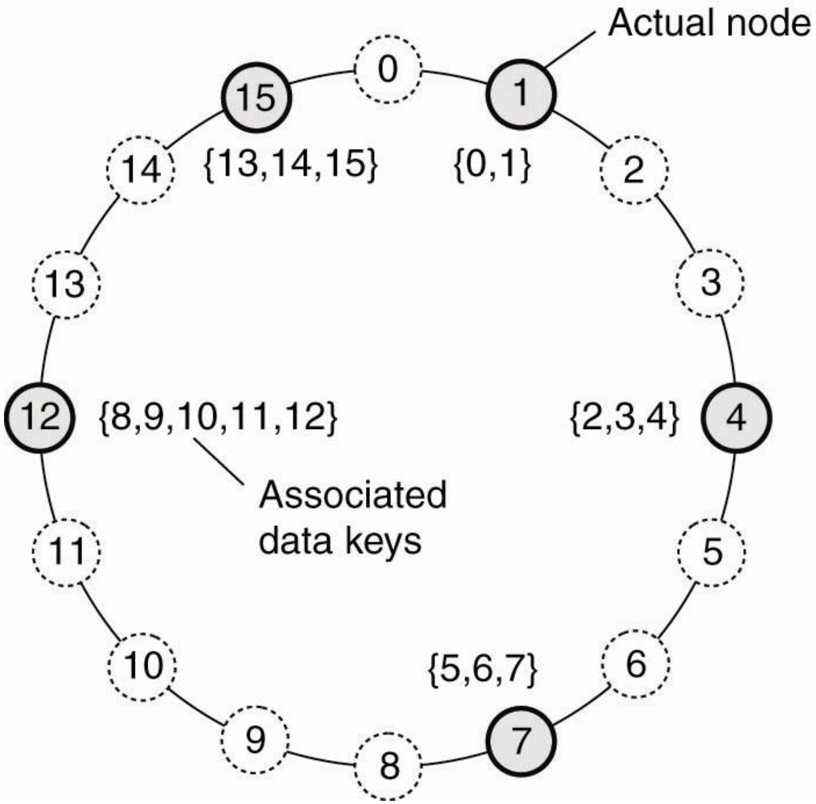
Data exchange between publishers and subscribers.



Peer-to-peer systems

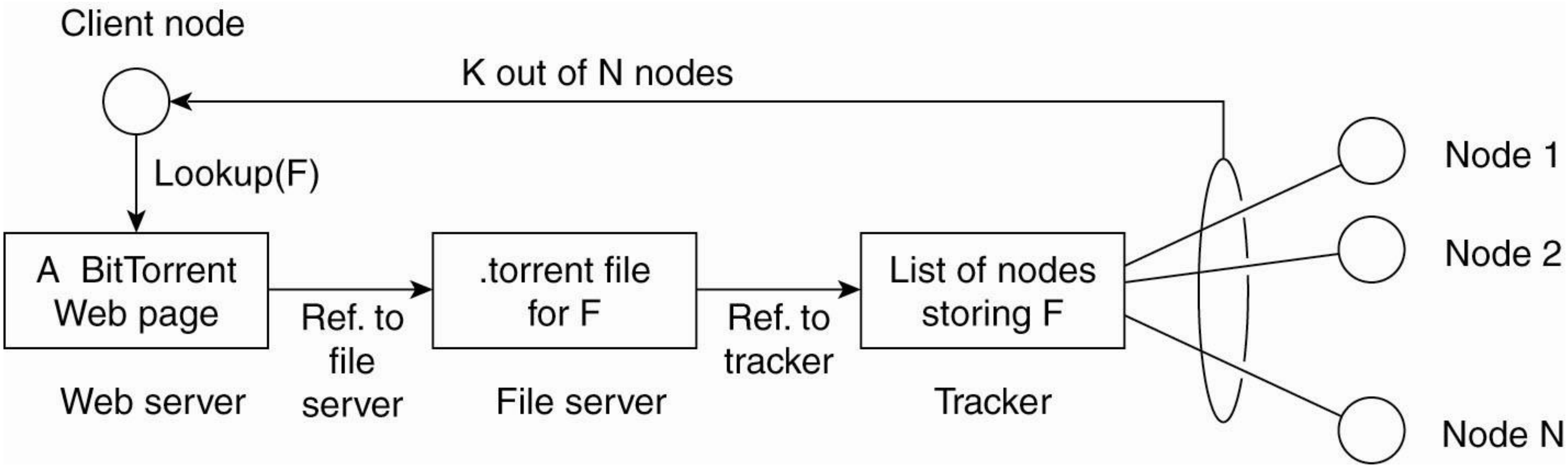
Peer-to-peer (P2P) systems rely on horizontal distribution and are designed to deal with churn (machines joining and leaving). They organize processes in an **overlay network** that defines a set of communication channels.

Example: Chord is a P2P **distributed hash table (DHT)** that uses a ring overlay (+ shortcuts, not shown).



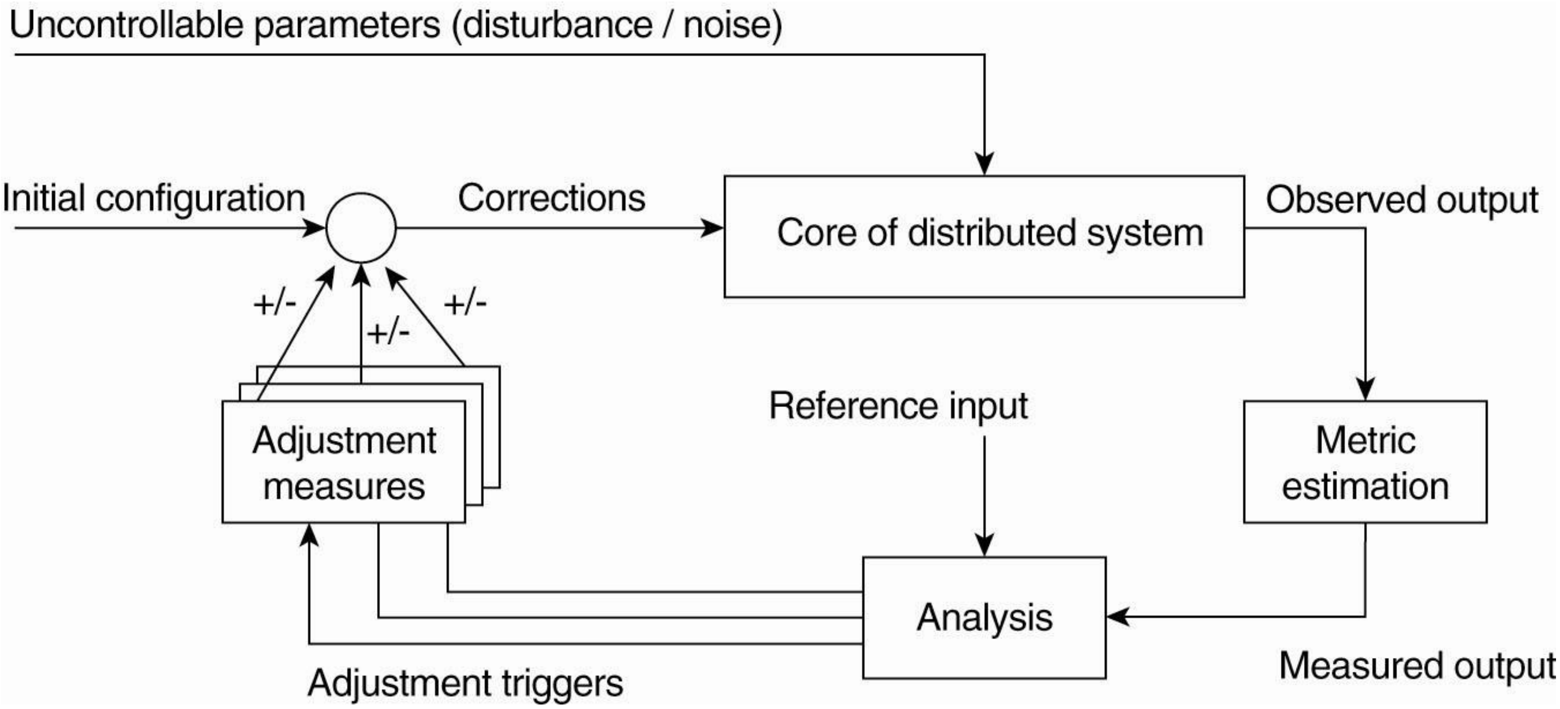
Hybrid architectures

BitTorrent combines client-server and P2P architectures. Client nodes obtain tracker information from a server and then exchange data with peer nodes.



Self-management

Self-managing systems can be constructed using a **feedback control loop** that monitors system behaviors and adjusts the system's internal operation (e.g., data placement, scheduling).


Uncontrollable parameters (disturbance / noise)
Initial configuration
Corrections
Core of distributed system
Observed output
+/-
+/-
+/-
Adjustment measures
Reference input
Metric estimation
Analysis
Measured output
Adjustment triggers