DISTRIBUTED SYSTEMS
Principles and Paradigms
Second Edition

ANDREW S. TANENBAUM
MAARTEN VAN STEEN

Chapter 2
ARCHITECTURES

Plan

- Software architecture
  - Architectural styles
- System architectures
  - Centralized architectures
  - Decentralized architectures
  - Hybrid architectures

Architecture

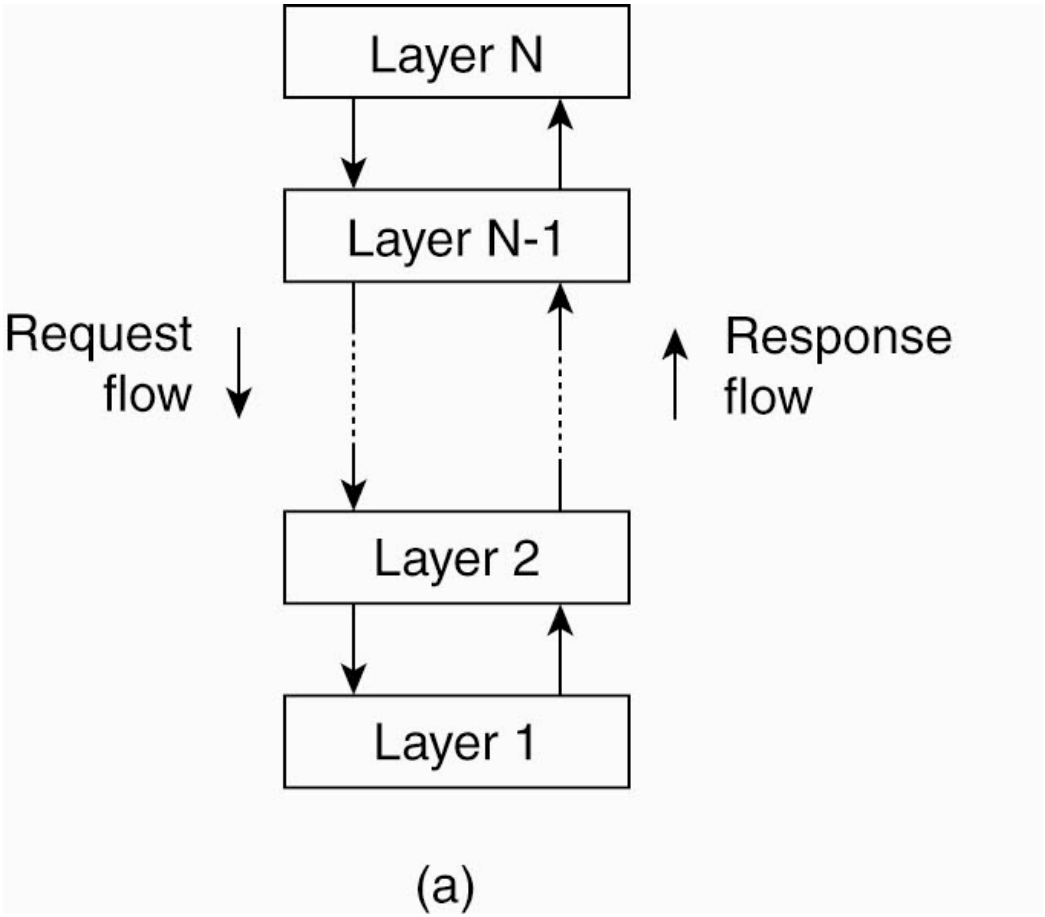
- Software Architecture
  - How our software components are organized
- System Architecture
  - How to realize our software architecture

Architectural Styles (1)

- Architectural style describe
  - Component/element types
  - Connections
  - Data exchange

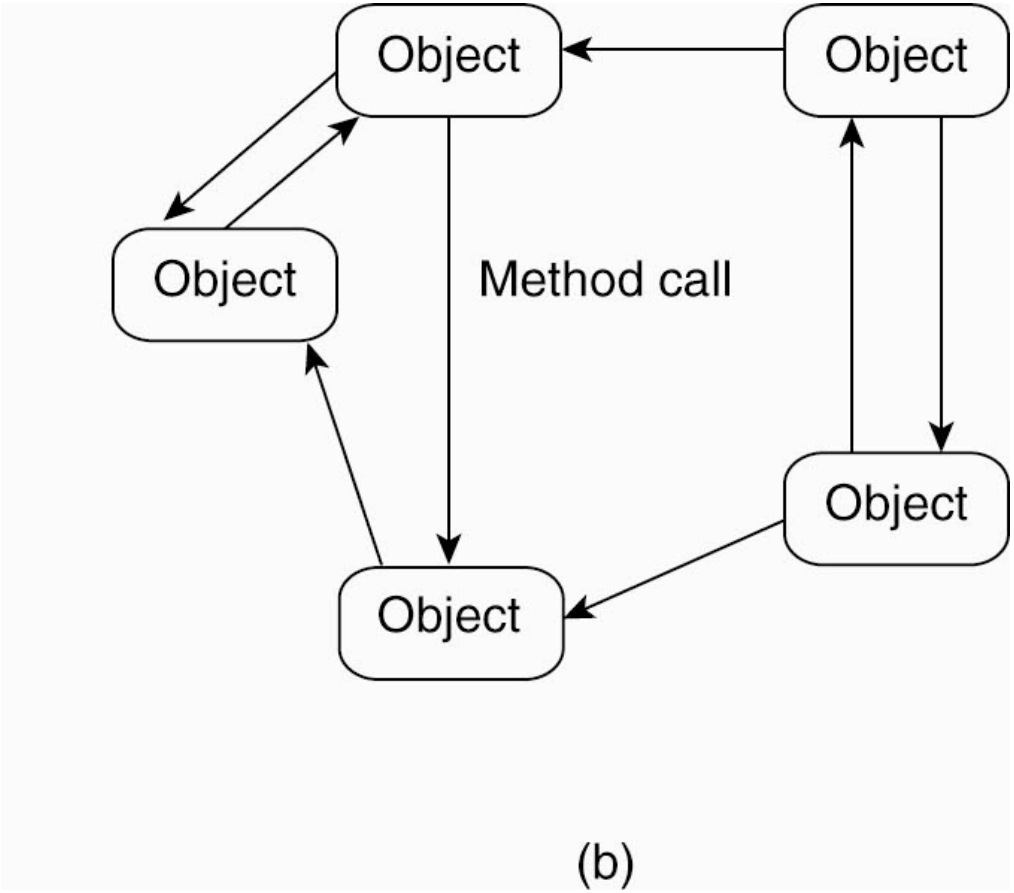
- Important styles of architecture for distributed systems
  - Layered architectures
  - Object-based architectures
  - Data-centered architectures
  - Event-based architectures

Architectural Styles (2)



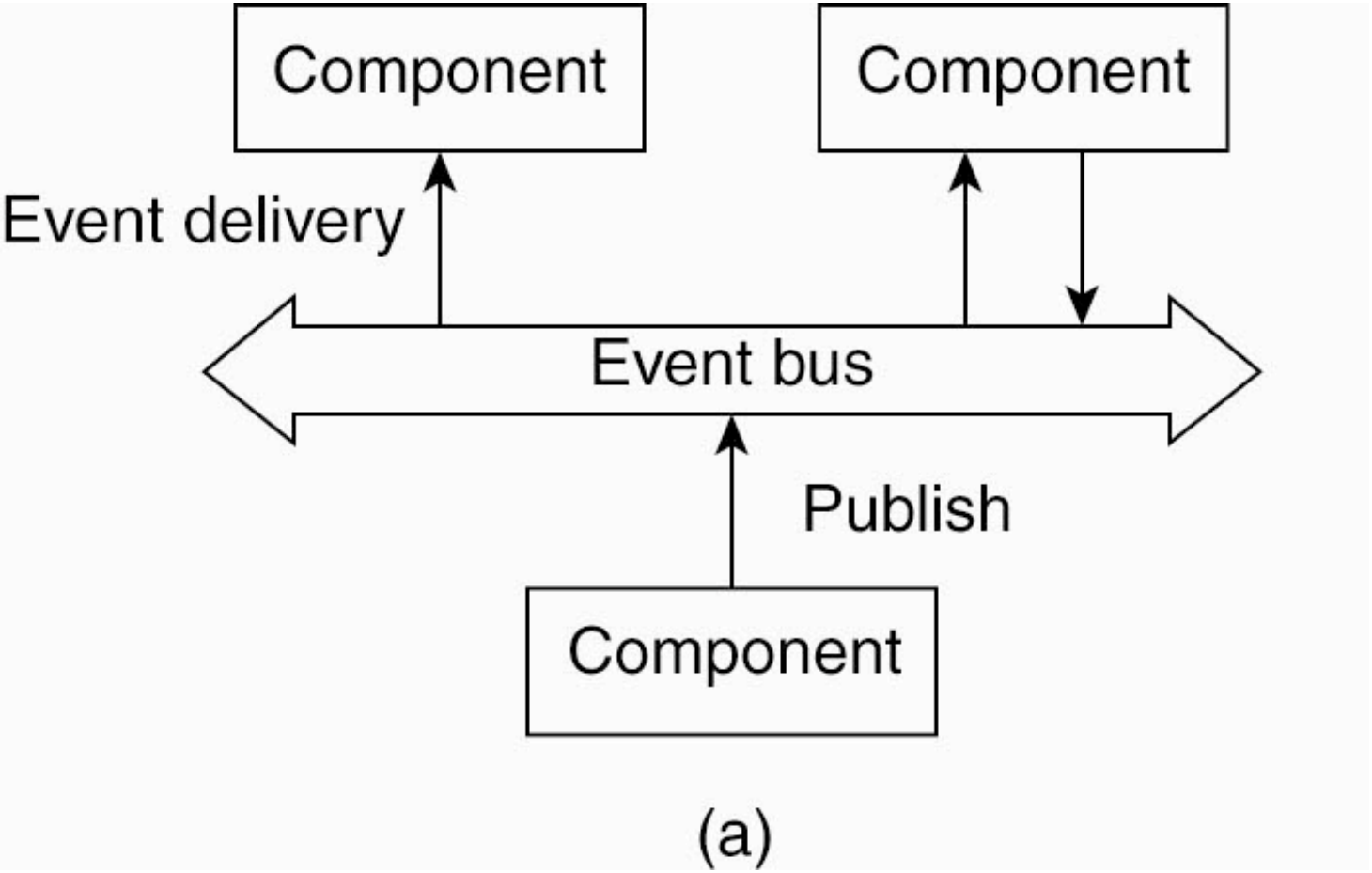
- Figure 2-1. The (a) layered architectural style and …

Architectural Styles (3)



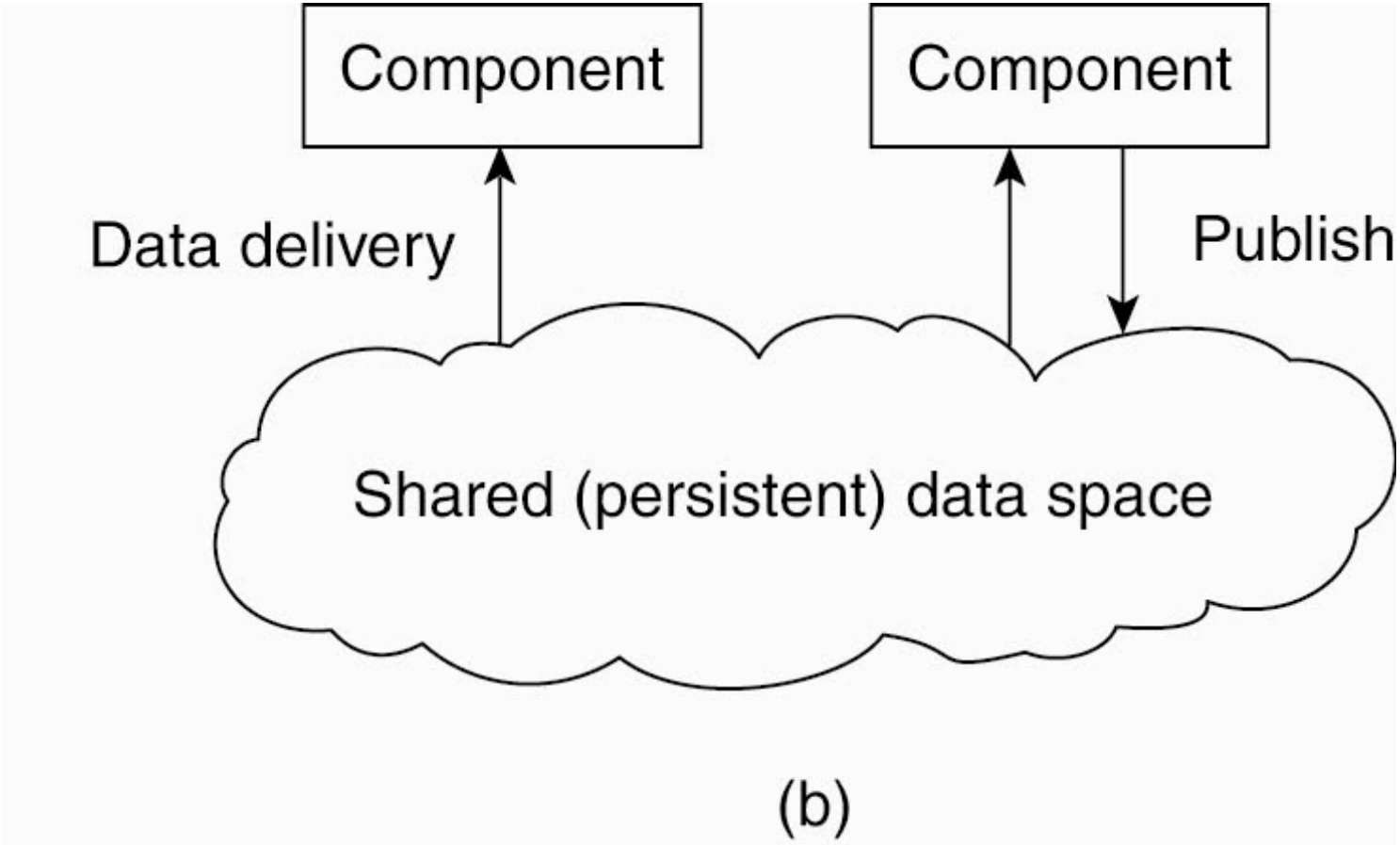
- Figure 2-1. (b) The object-based architectural style.

Architectural Styles (4)



- Figure 2-2. (a) The event-based architectural style and …

Architectural Styles (5)

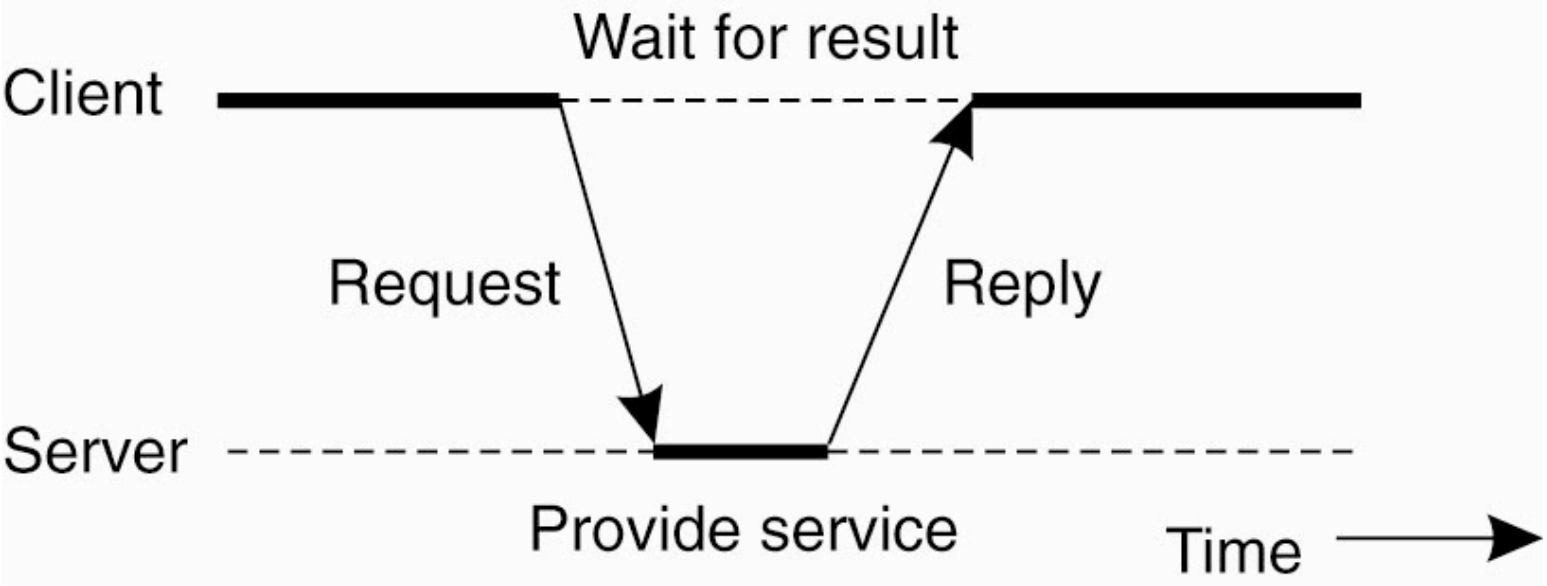


- Figure 2-2. (b) The shared data-space architectural style.

System Architectures

- Centralized
- Decentralized
- Hybrid

System Architecture: Centralized Architectures

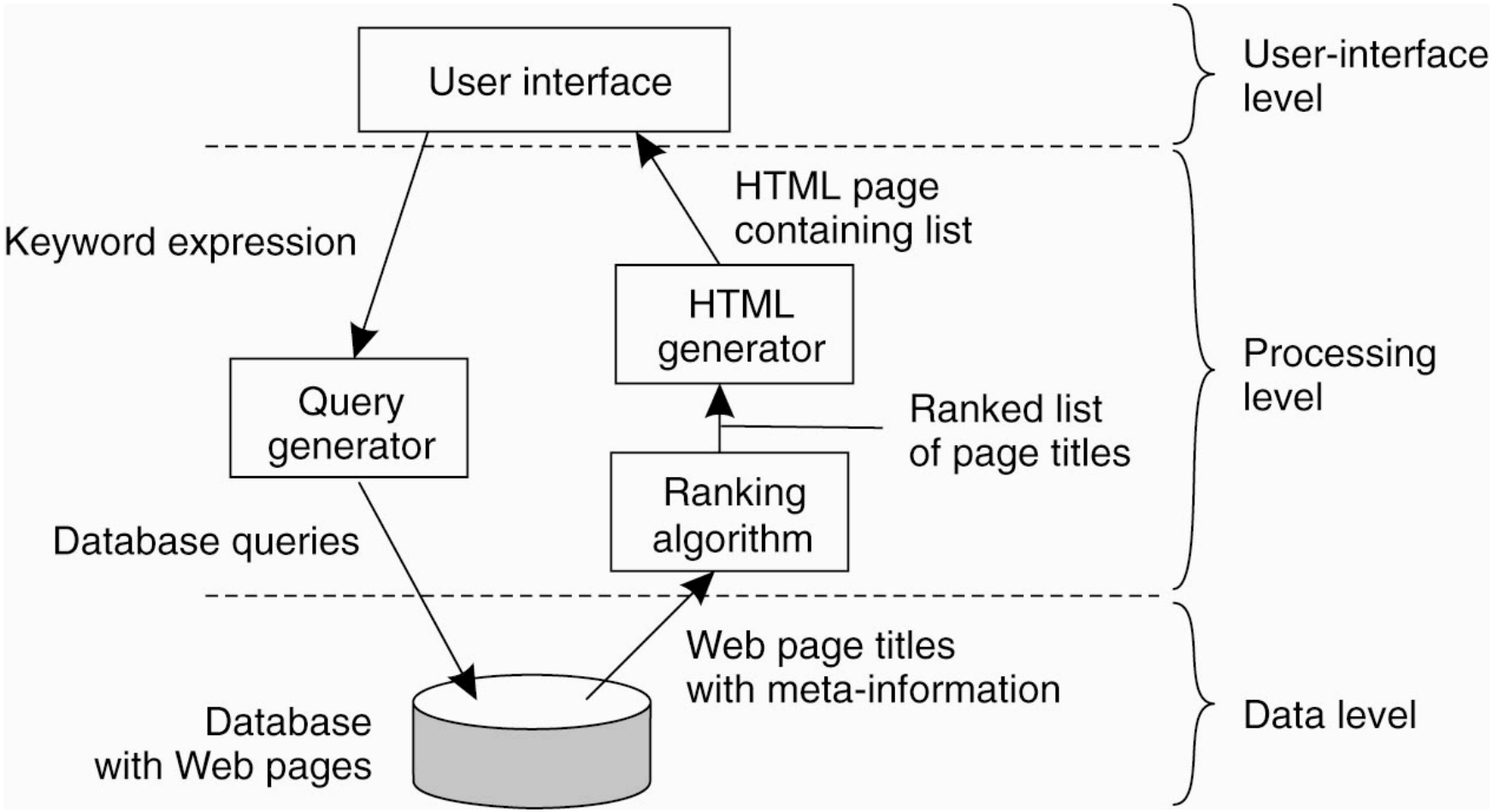


- Figure 2-3. General interaction between a client and a server.

Application Layering (1)

- Recall previously mentioned layers of architectural style
  - The user-interface level
  - The processing level
  - The data level

Application Layering (2)

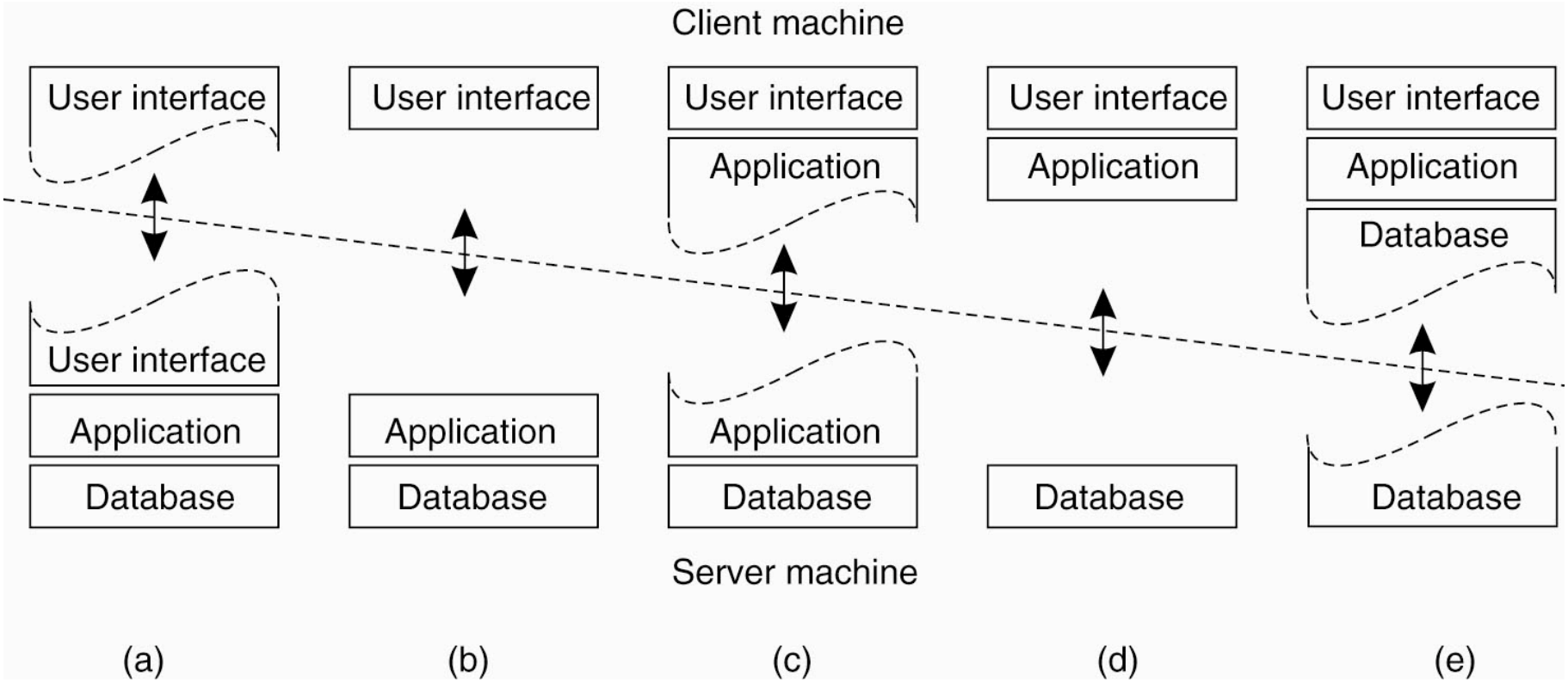


- Figure 2-4. The simplified organization of an Internet search engine into three different layers.

Multitiered Architectures (1)

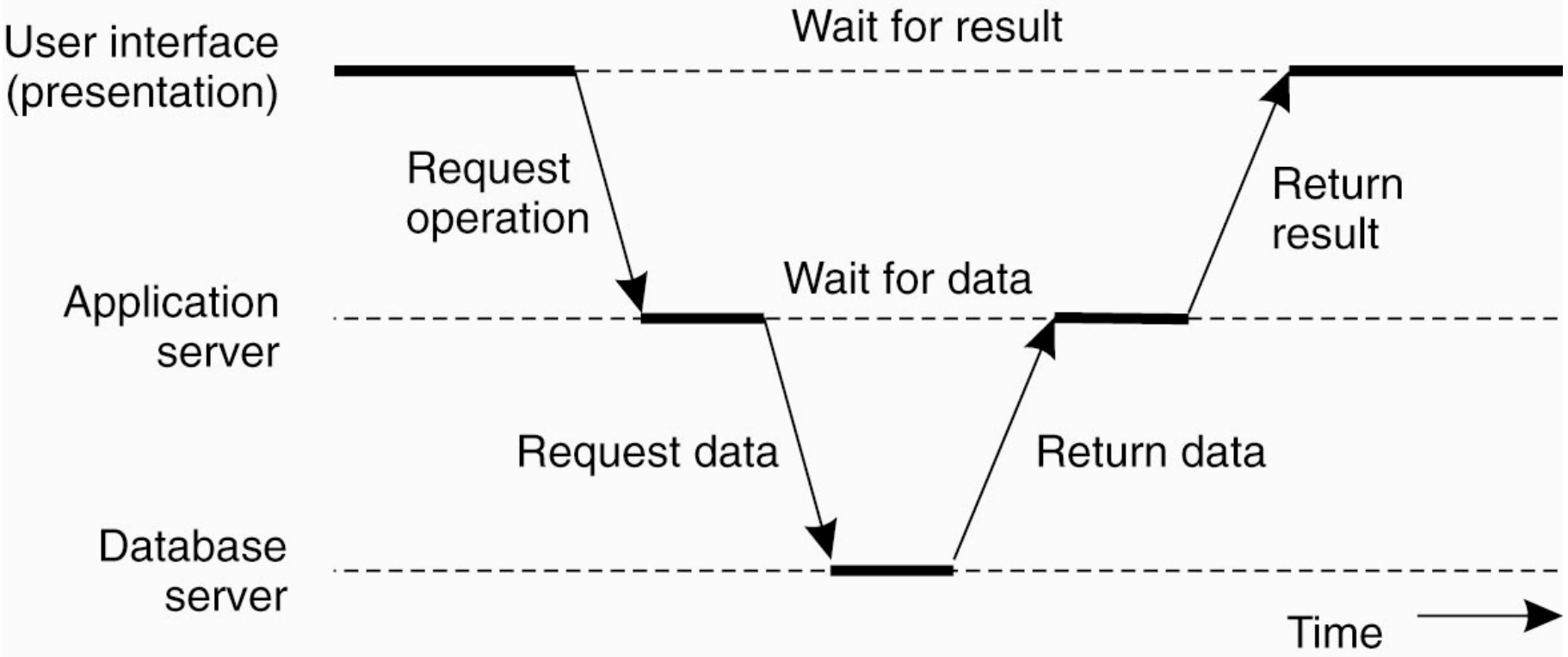
- The simplest organization is to have only two types of machines:
  - A client machine containing only the programs implementing (part of) the user-interface level
  - A server machine containing the rest,
    - the programs implementing the processing and data level

Multitiered Architectures (2)



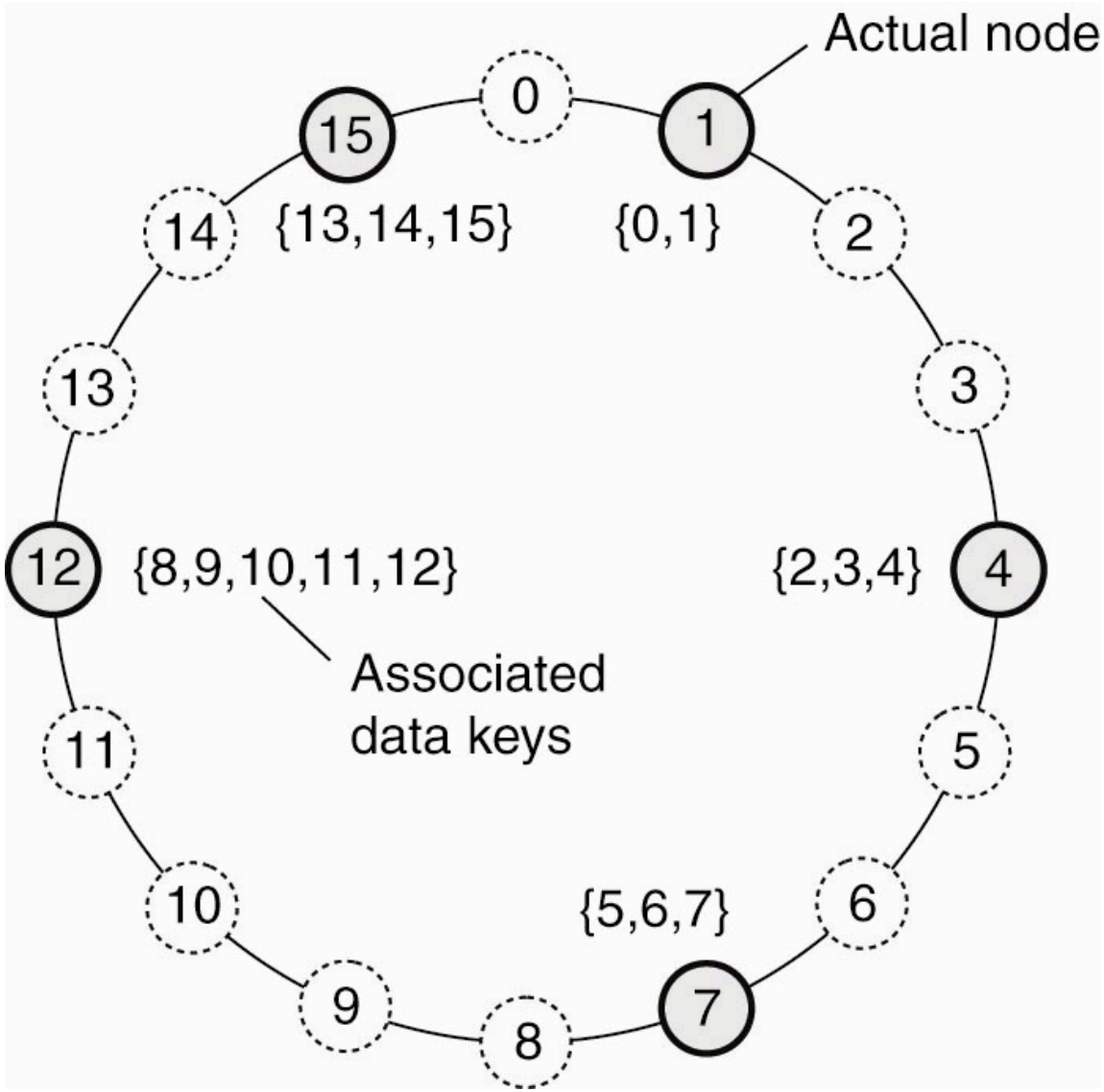
- Figure 2-5. Alternative client-server organizations (a)–(e).

Multitiered Architectures (3)



- Figure 2-6. An example of a server acting as client.
- Why do this?

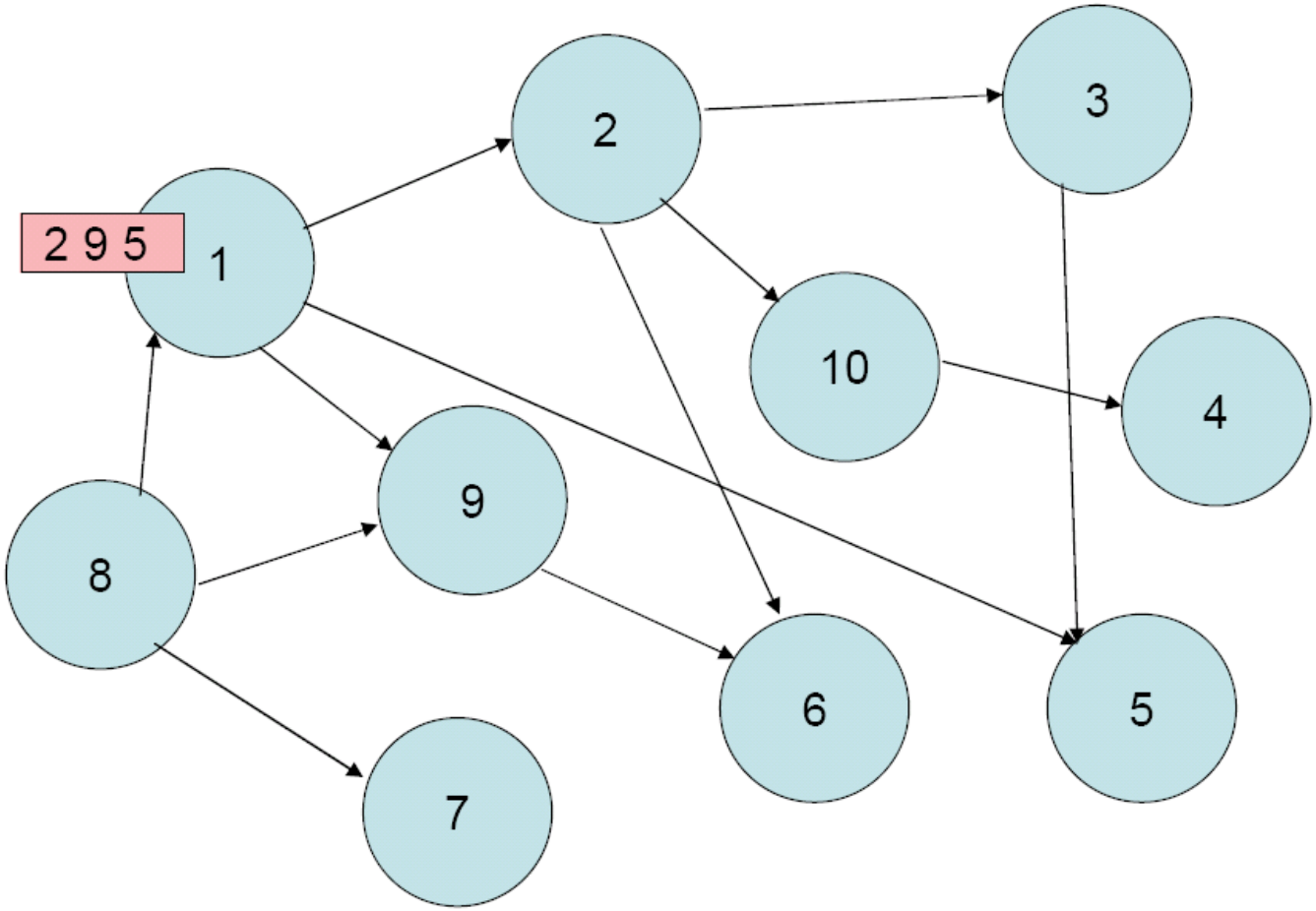
Structured Peer-to-Peer Architectures (1)

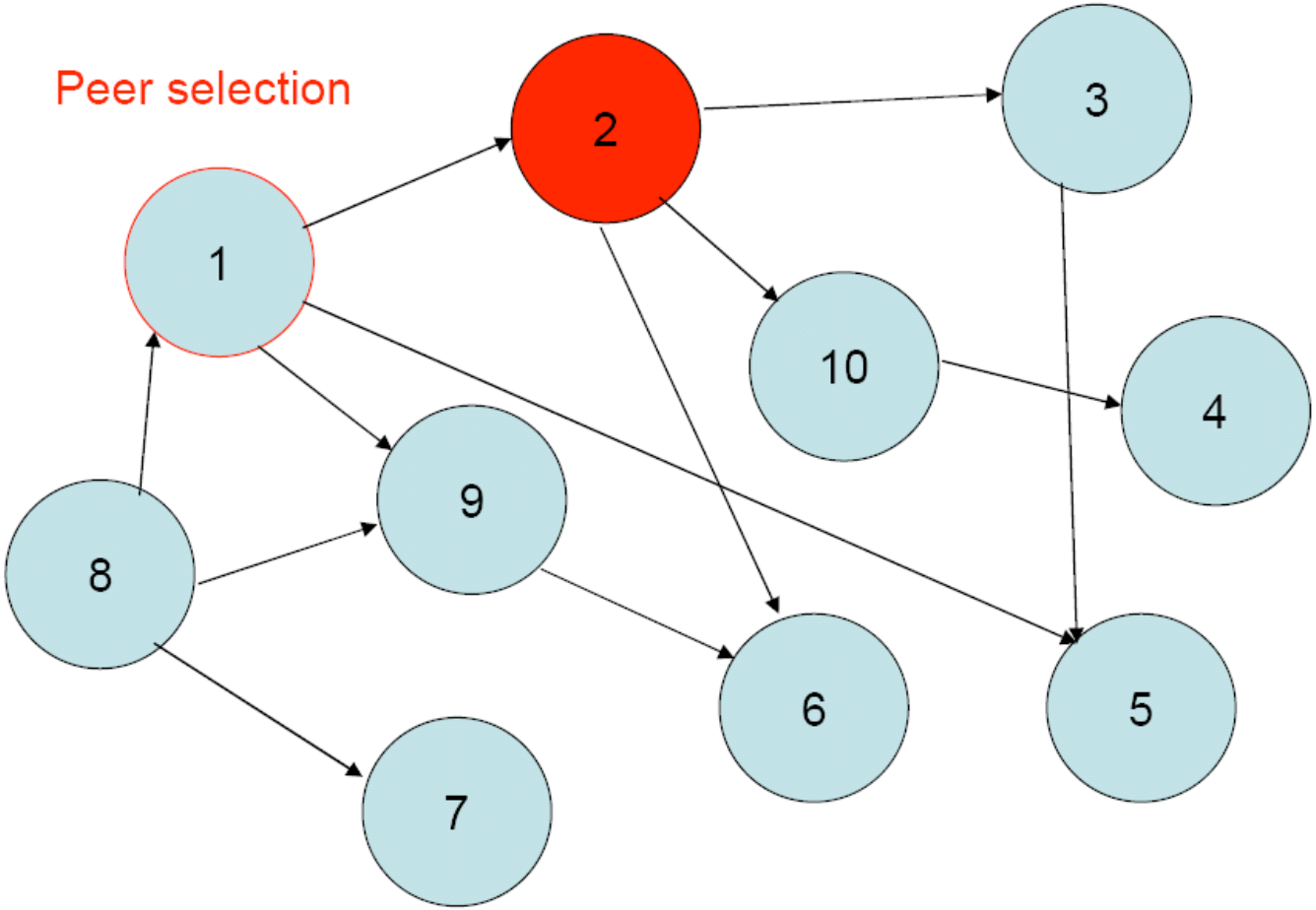


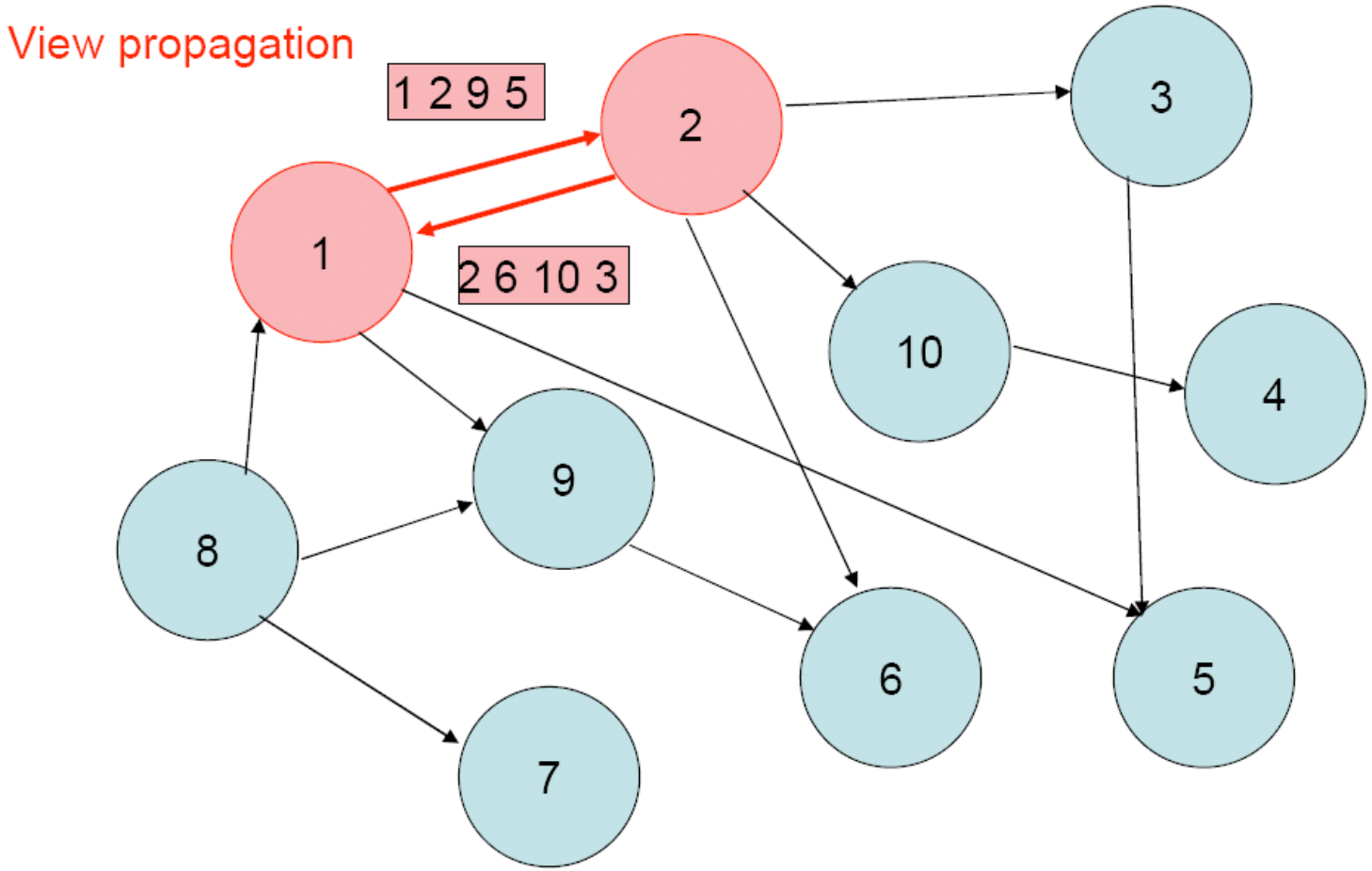
- Figure 2-7. The mapping of data items onto nodes in Chord.

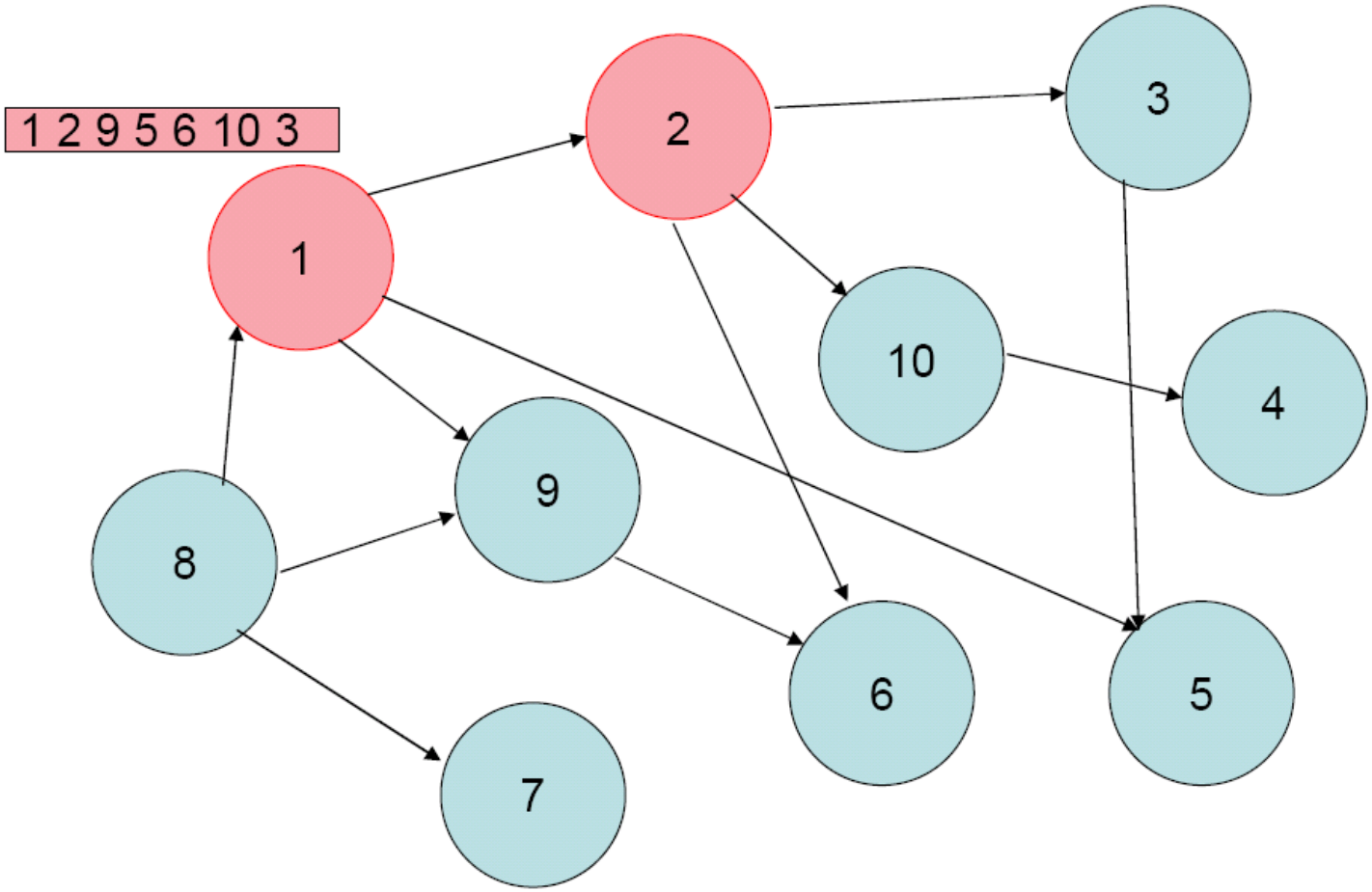
Unstructured Peer-to-Peer Architectures

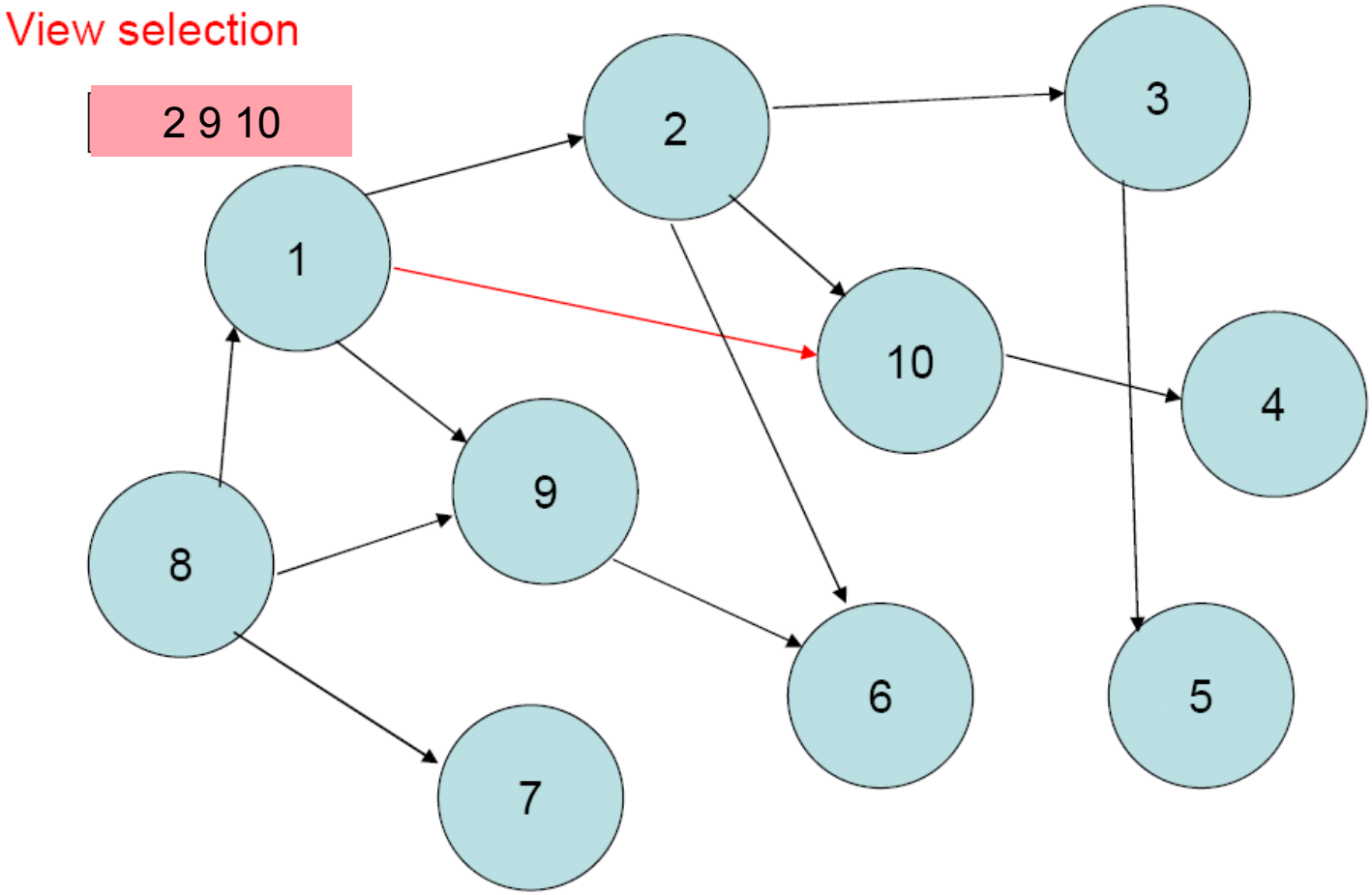
- Uses randomized algorithm for constructing overlay network
  - Goal: maintain random graph
- Framework for capturing different overlay construction algorithms
  - N peers
  - View of c entries
    - Freshness/hop count
    - IP
  - Active and passive thread at each node
- Generic protocol











Active and PassiveThread

```
do forever
    wait(T time units)
    p ← selectPeer()
    if push then
        // 0 is the initial hop count
        myDescriptor ← (myAddress, 0)
        buffer ← merge(view,{myDescriptor})
        send buffer to p
    else
        // empty view to trigger response
        send {} to p
    if pull then
        receive view_p from p
        view_p ← increaseHopCount(view_p)
        buffer ← merge(view_p,view)
        view ← selectView(buffer)
```

(a) active thread

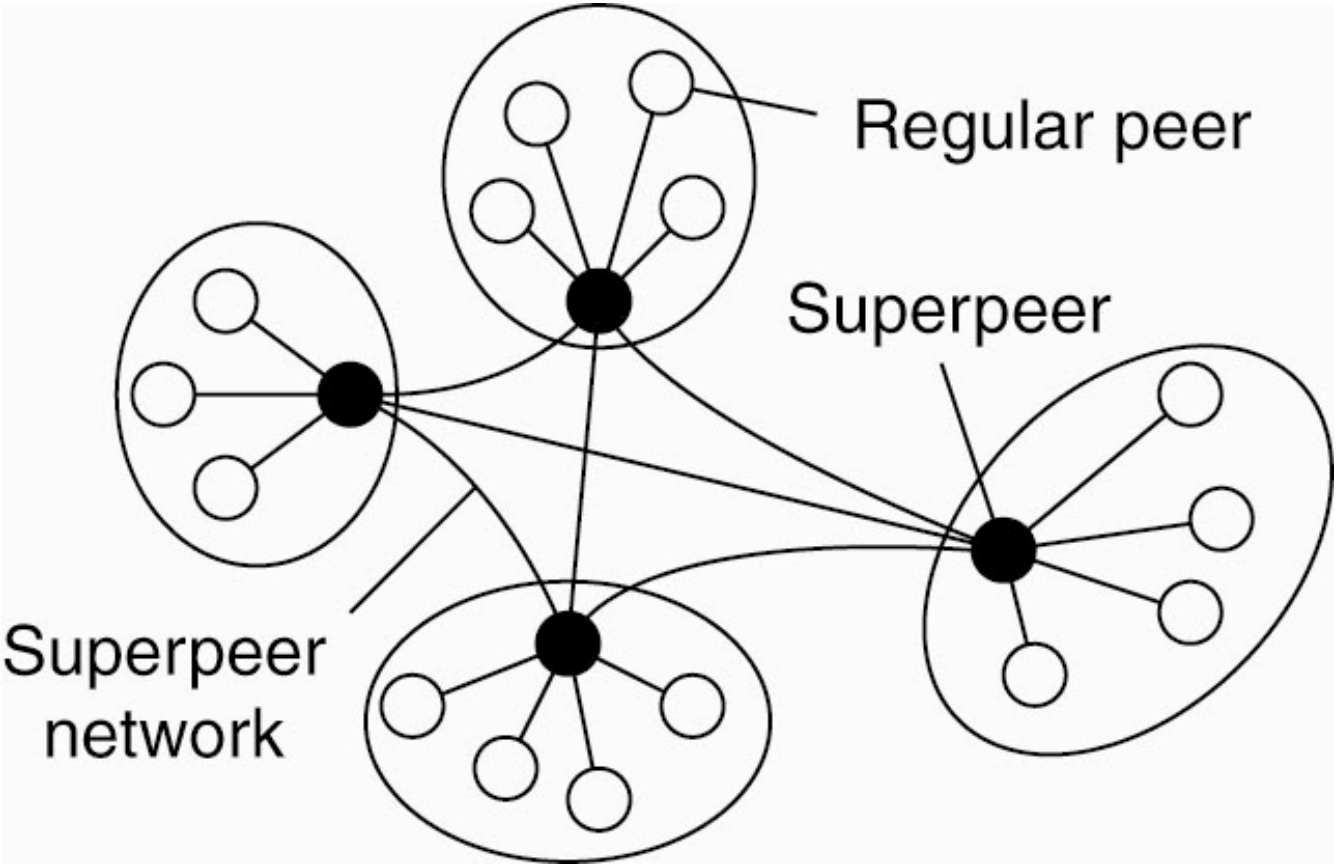
```
do forever
    (p, view_p) ← waitMessage()
    view_p ← increaseHopCount(view_p)
    if pull then
        // 0 is the initial hop count
        myDescriptor ← (myAddress, 0)
        buffer ← merge(view,{myDescriptor})
        send buffer to p
    buffer ← merge(view_p,view)
    view ← selectView(buffer)
```

(b) passive thread

Design Space

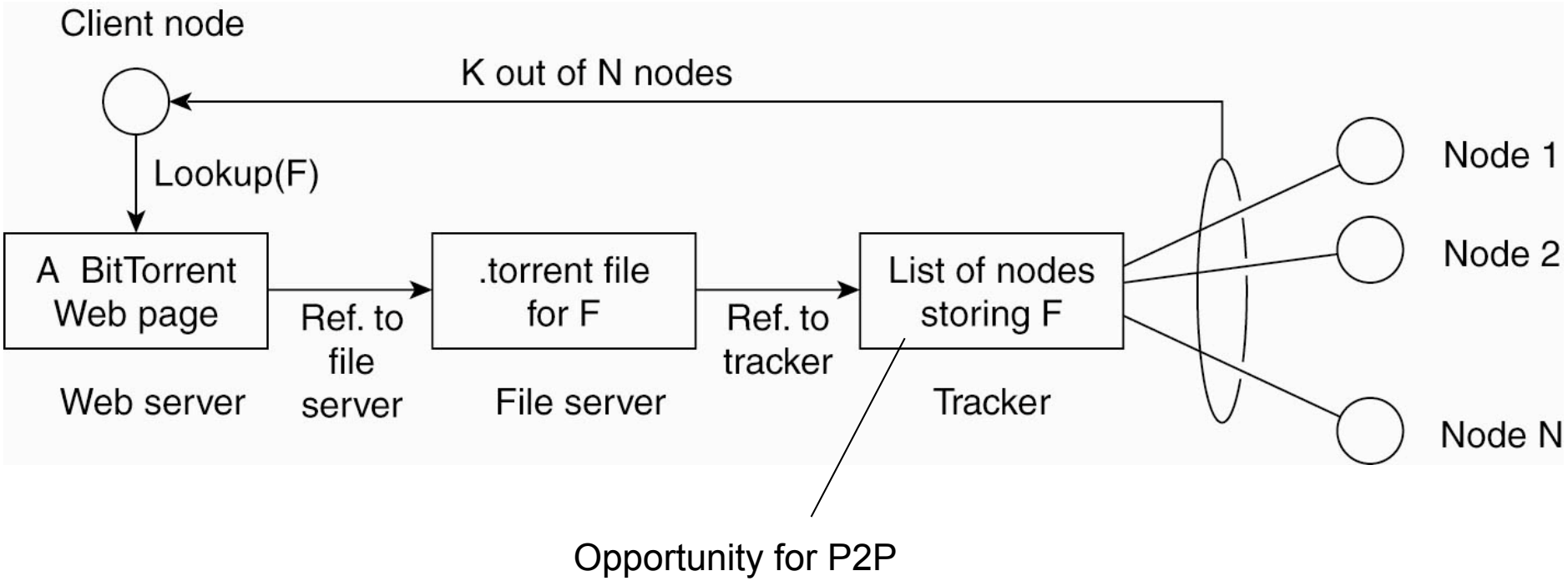
- selectPeer(), e.g.,
  - Random
    - Select random peer from view
  - Tail
    - Select node with highest hop count
- View propagation type
  - Push
    - Node sends it buffer to selected peer
  - Pushpull
    - Node and selected peer exchange information
- selectView(), e.g.,
  - Blind selection of subset
  - Remove oldest entries

Superpeers



- Figure 2-12. A hierarchical organization of nodes into a superpeer network.

Hybrid Architecture



- Figure 2-14. The principal working of BitTorrent [adapted with permission from Pouwelse et al. (2004)].

Summary

- Software Architecture defines how software components are distributed
  - Layered, Object based, Data centered, Event-based
- System Architecture described how to realize the software architecture
  - Structured (*n*-tier layered client-server)
  - Unstructured (peer-to-peer)
  - Hybrid