DISTRIBUTED SYSTEMS
Principles and Paradigms
Second Edition
ANDREW S. TANENBAUM
MAARTEN VAN STEEN

Chapter 2
ARCHITECTURES

Architectural Styles

Important styles of architecture for distributed systems

- Layered architectures
- Object-based architectures
- Data-centered architectures
- Event-based architectures

Layered Architectures

- The basic idea for the layered style is simple: components are organized in a layered fashion where a component at layer L_i is allowed to call components at the underlying layer L_{i-1}

Layered Architectures (cont.)

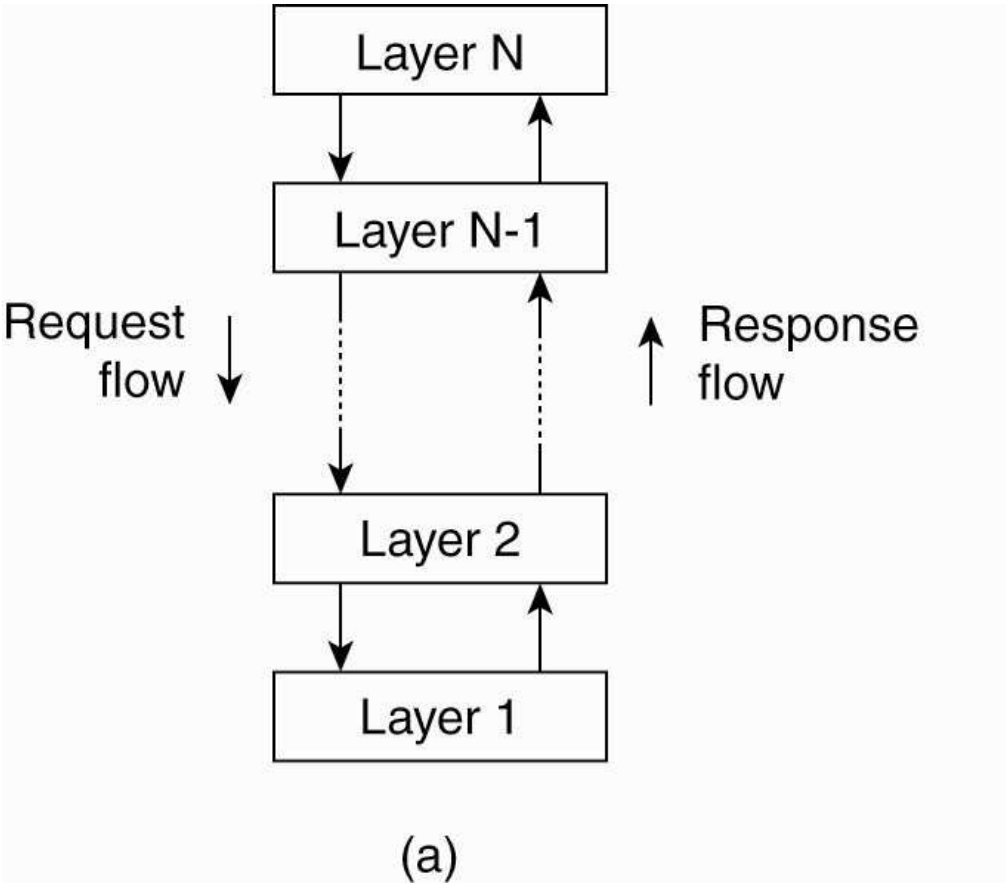

Figure 2-1. The (a) layered architectural style and …

Object-based Architectures

- Each object corresponds to what we have defined as a component, and these components are connected through a remote procedure call mechanism

- The layered and object-based architectures still form the most important styles for large software systems

Object-based Architectures (cont.)

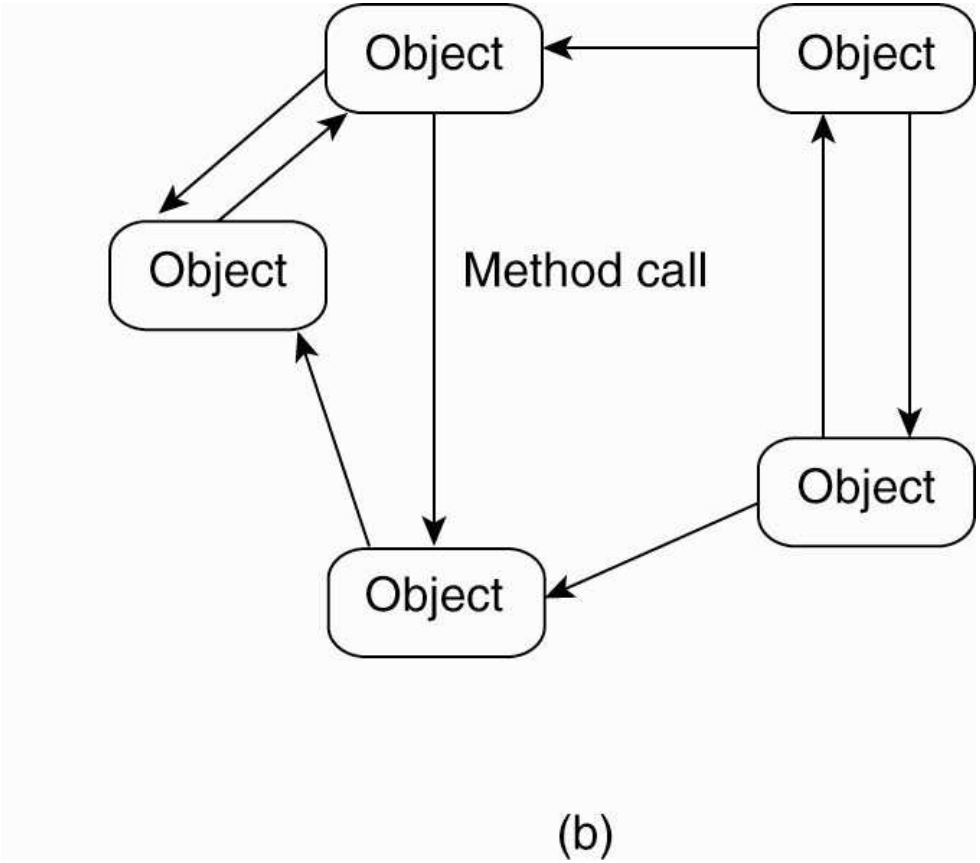

Figure 2-1. (b) The object-based architectural style.

Data-centered Architectures

- Data-centered architectures evolve around the idea that processes communicate through a common (passive or active) repository

- (**Example**): A wealth of networked applications have been developed that rely on a shared distributed file system in which virtually all communication takes place through files

Event-based Architectures

- In event-based architectures, processes essentially communicate through the propagation of events, which optionally also carry data
- For distributed systems, event propagation has generally been associated with what are known as publish/subscribe systems

Event-based Architectures (cont.)

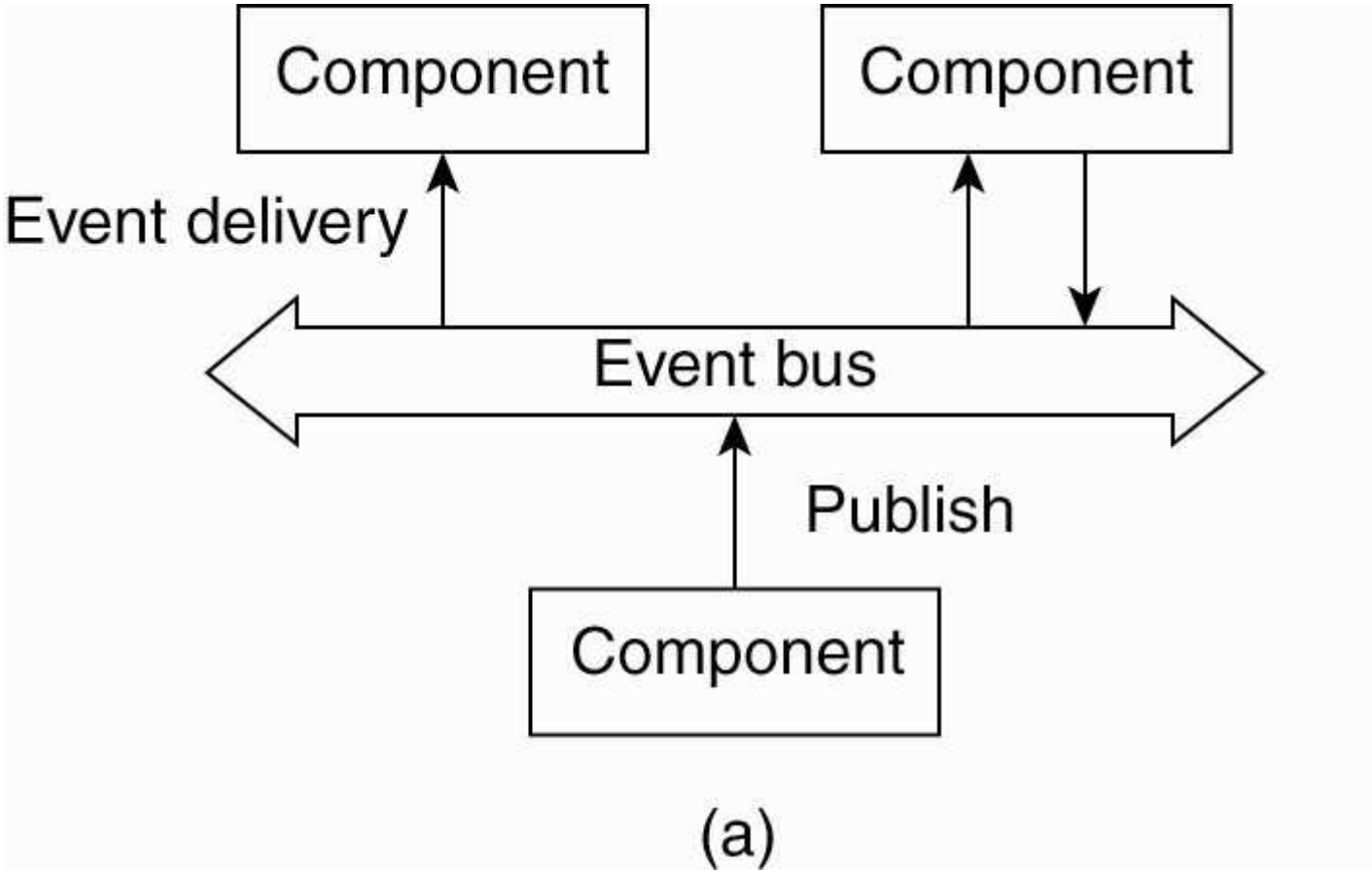

Figure 2-2. (a) The event-based architectural style and …

Shared Data-space Architectures

- Event-based architectures can be combined with data-centered architectures, yielding what is also known as shared data spaces

- The essence of shared data spaces is that processes are now also decoupled in time: they need not both be active when communication takes place

Shared Data-space Architectures (cont.)

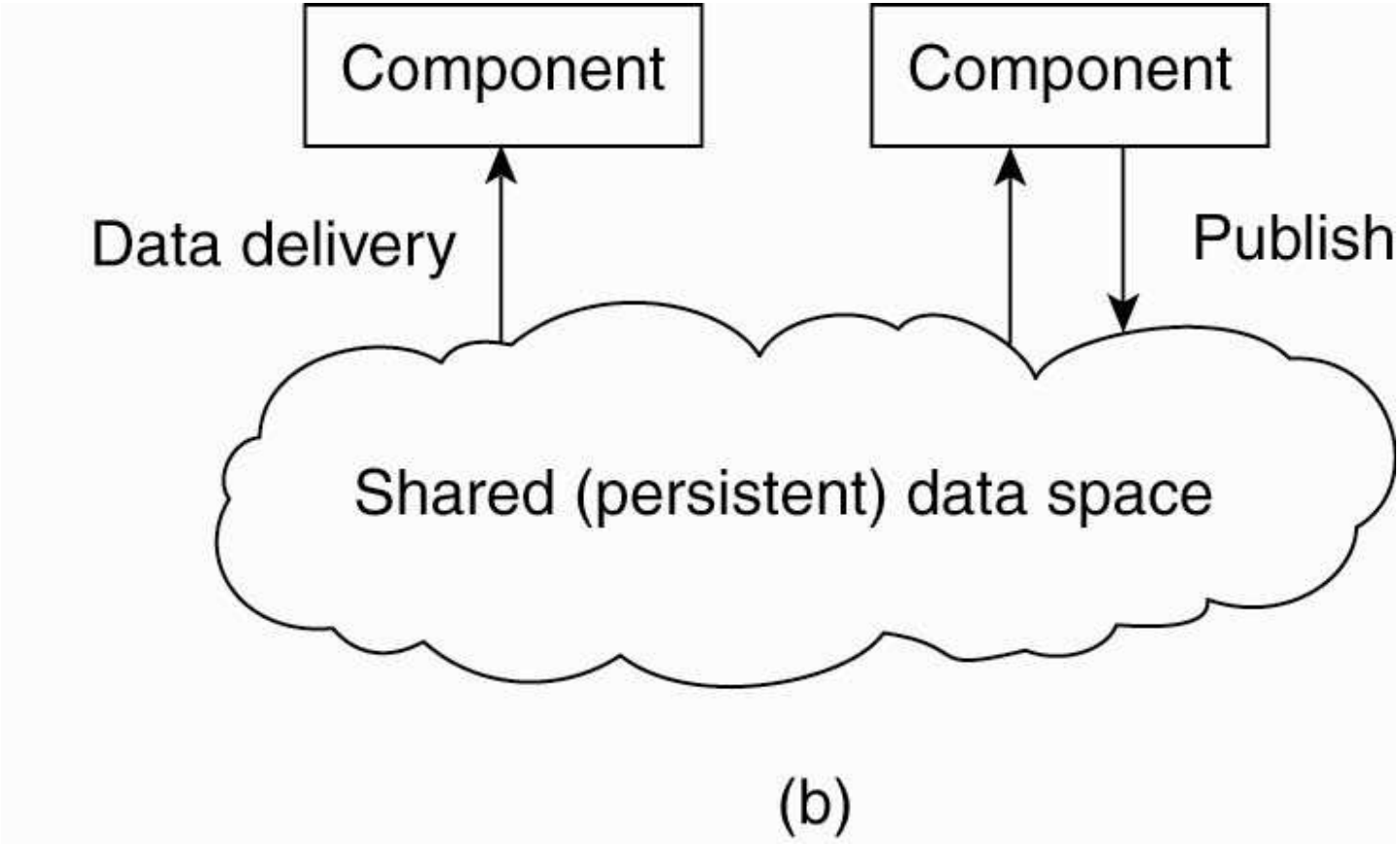

Figure 2-2. (b) The shared data-space architectural style.

System Architectures

- Centralized Architectures
- Decentralized Architectures
- Hybrid Architectures

Centralized Architectures

- In the basic client-server model, processes in a distributed system are divided into two (possibly overlapping) groups.
- A server is a process implementing a specific service, for example, a file system service or a database service.
- A client is a process that requests a service from a server by sending it a request and subsequently waiting for the server's reply.

Centralized Architectures

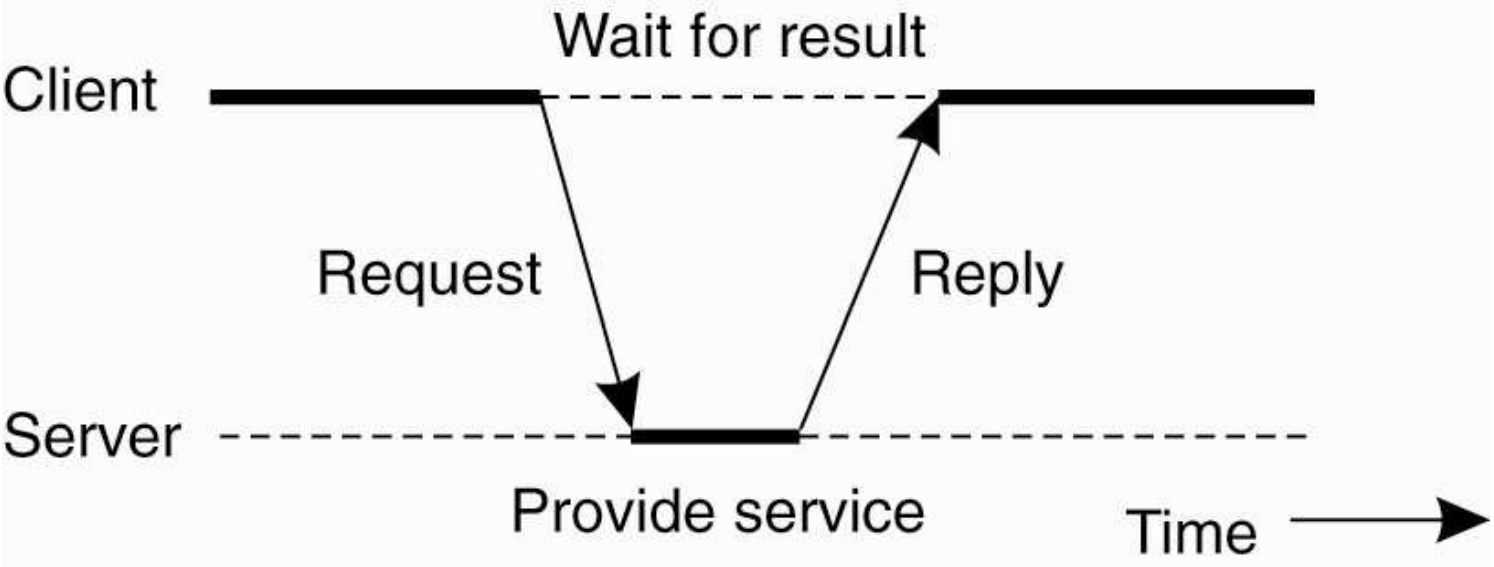

Figure 2-3. General interaction between a client and a server.

Application Layering

Recall previously mentioned layers of architectural style

- The user-interface level
- The processing level
- The data level

Application Layering (cont.)

- The user-interface level contains all that is necessary to directly interface with the user, such as display management.
- The processing level typically contains the applications.
- The data level manages the actual data that is being acted on.

Application Layering (cont.)

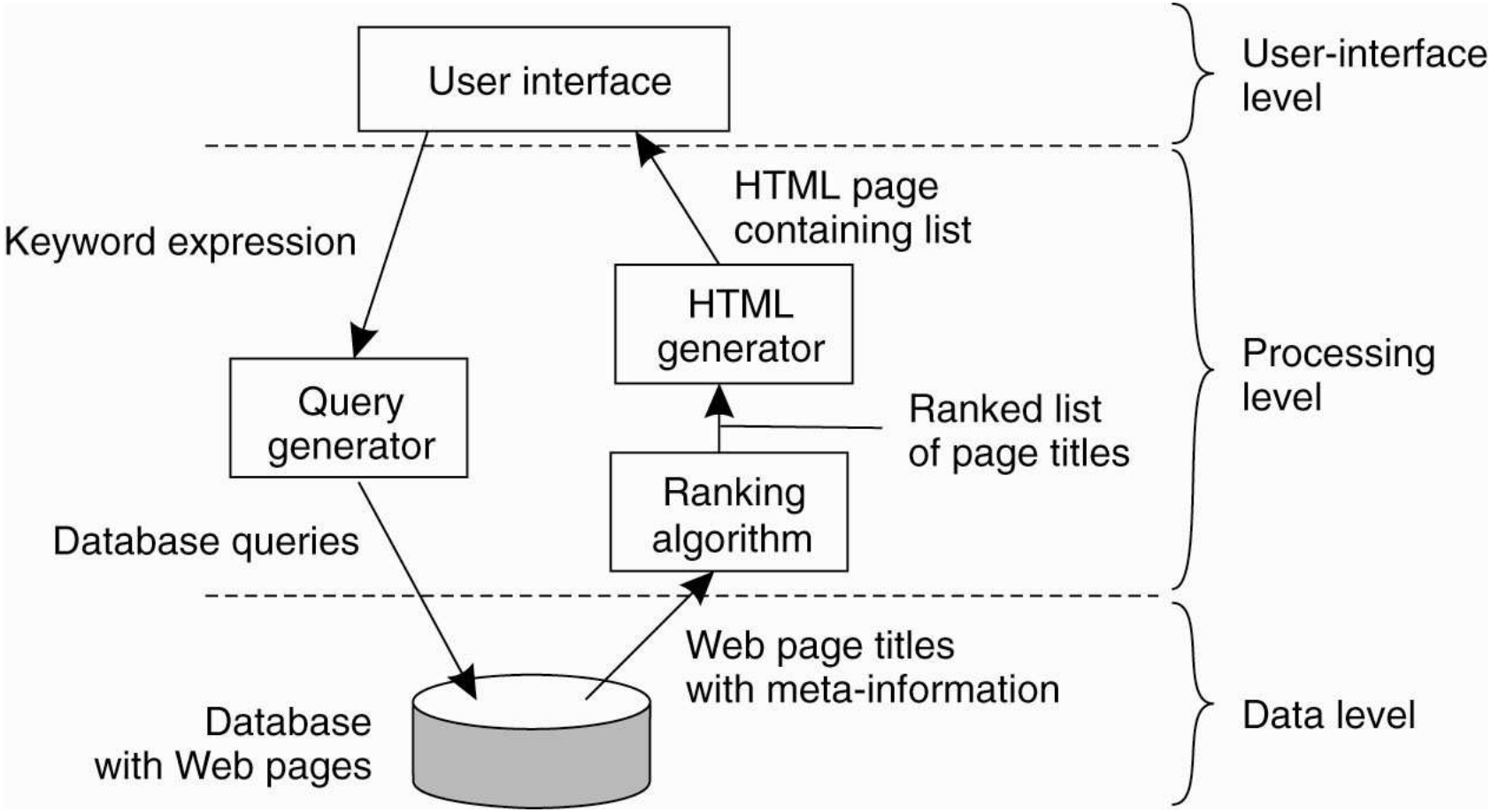

Figure 2-4. The simplified organization of an Internet search engine into three different layers.

Multitiered Architectures

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 (cont.)

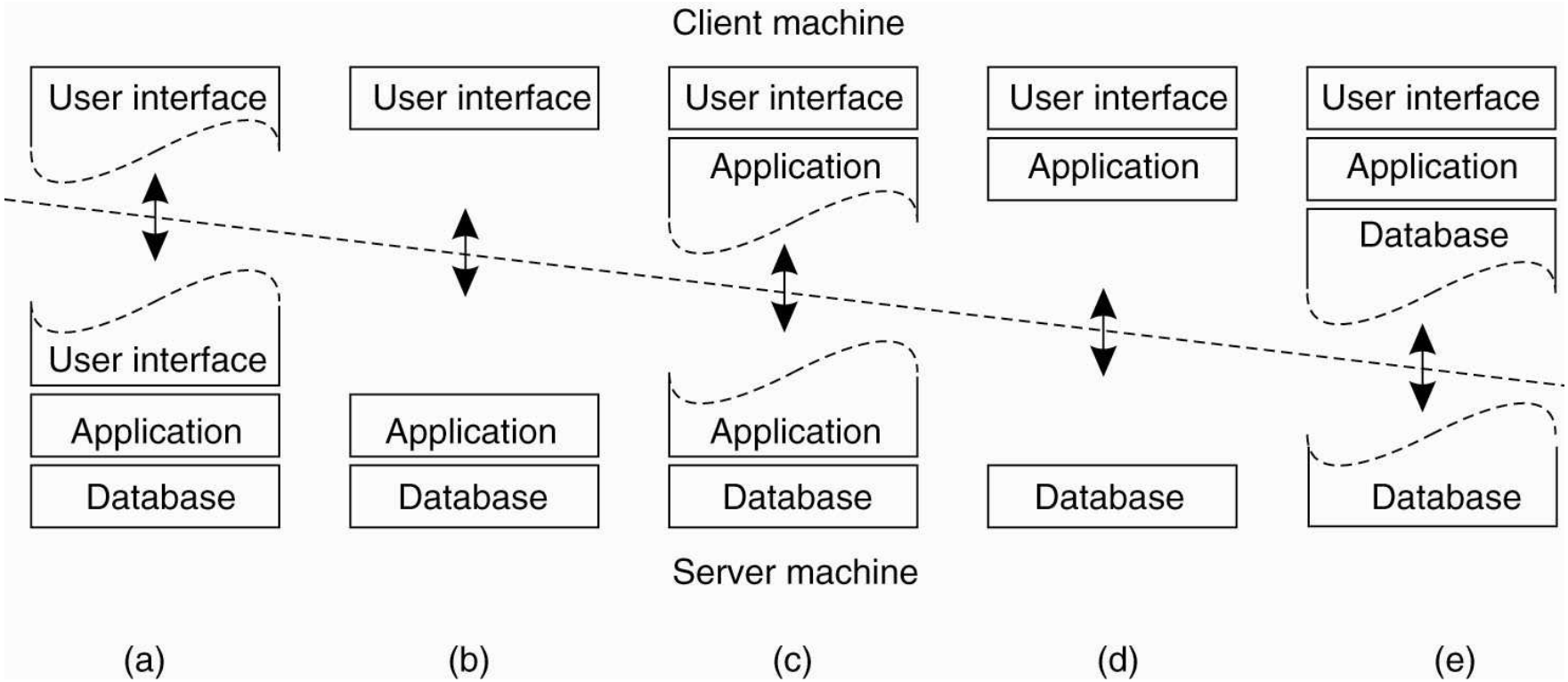

Figure 2-5. Alternative client-server organizations (a)–(e).

Multitiered Architectures (cont.)

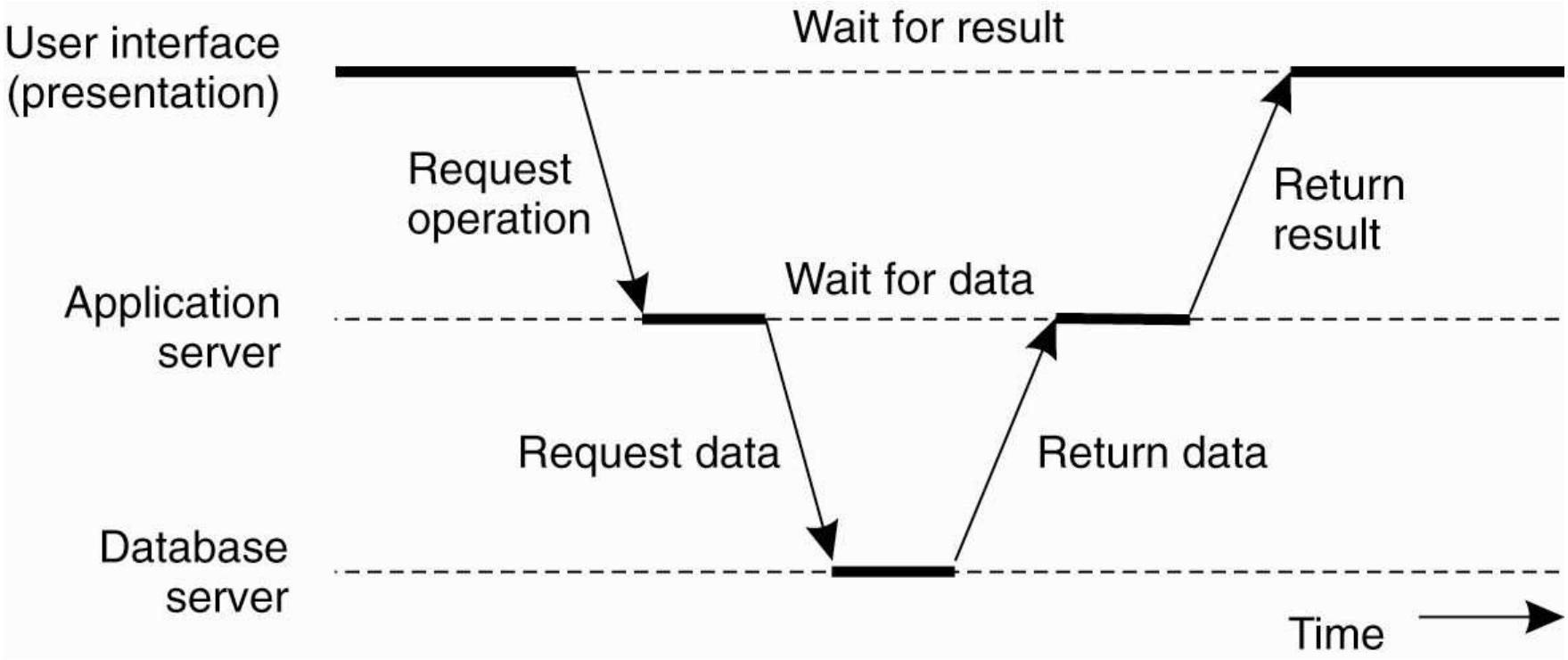

Figure 2-6. An example of a server acting as client.

Decentralized Architectures

- Vertical Distribution
  - The characteristic feature of vertical distribution is that it is achieved by placing logically different components on different machines.

- Horizontal Distribution
  - In this type of distribution, a client or server may be physically split up into logically equivalent parts, but each part is operating on its own share of the complete data set, thus balancing the load.

Peer-to-Peer Systems

- Peer-to-peer system is a class of modern system architectures that support horizontal distribution.
- From a high-level perspective, the processes that constitute a peer-to-peer system are all equal.
- functions that need to be carried out are represented by every process that constitutes the distributed system.

Peer-to-Peer Systems (cont.)

- Structured Peer-to-Peer Systems
  - In a structured peer-to-peer architecture, the overlay network is constructed using a deterministic procedure.
  - The most-used procedure is to organize the processes through a distributed hash table (DHT).
- Unstructured Peer-to-Peer Systems
  - Unstructured peer-to-peer systems largely rely on randomized algorithms for constructing an overlay network.
  - The main idea is that each node maintains a list of neighbors, but that this list is constructed in a more or less random way.

Structured Peer-to-Peer Architectures

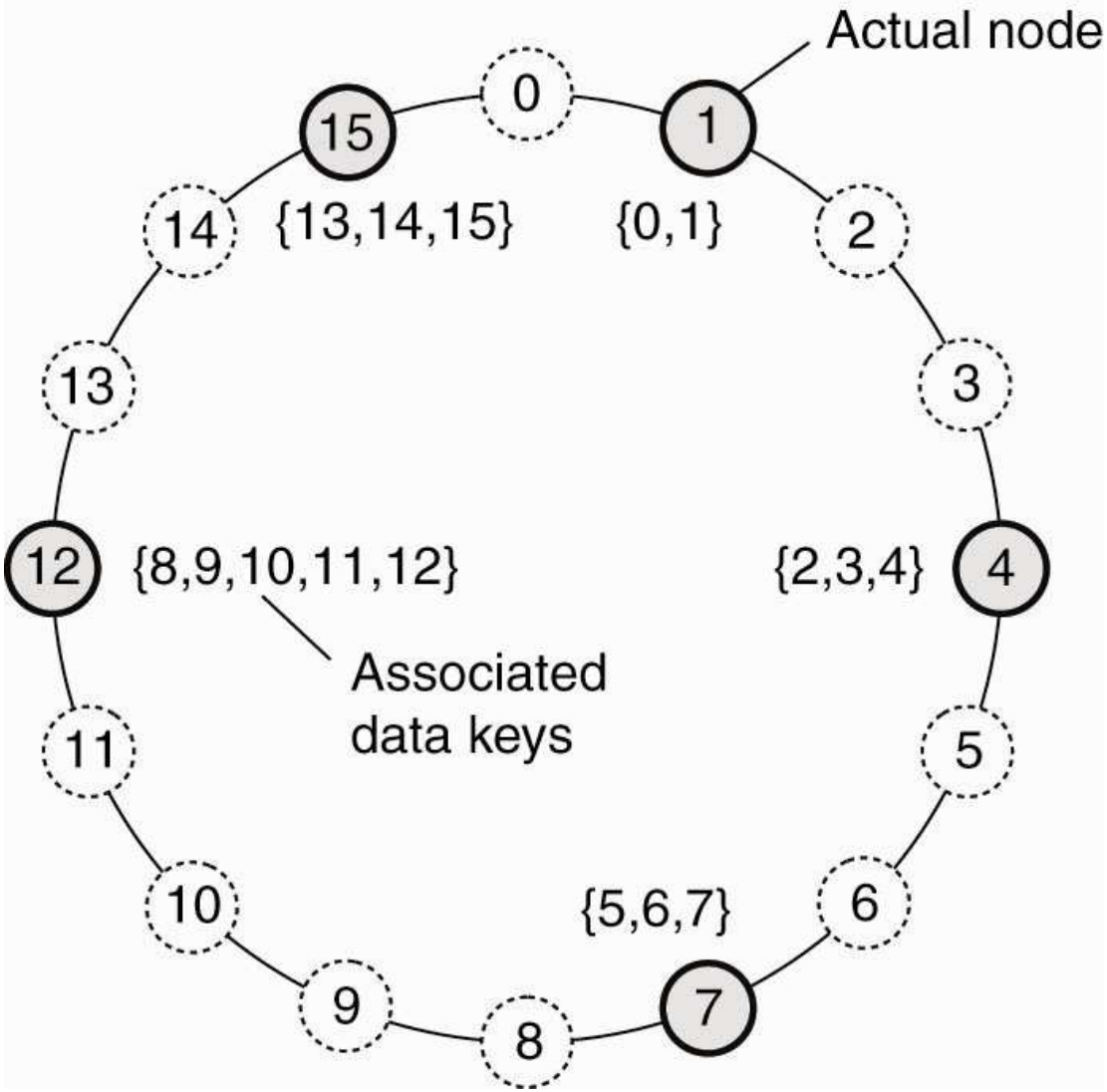

Figure 2-7. The mapping of data items onto nodes in Chord.

Structured Peer-to-Peer Architectures (cont.)

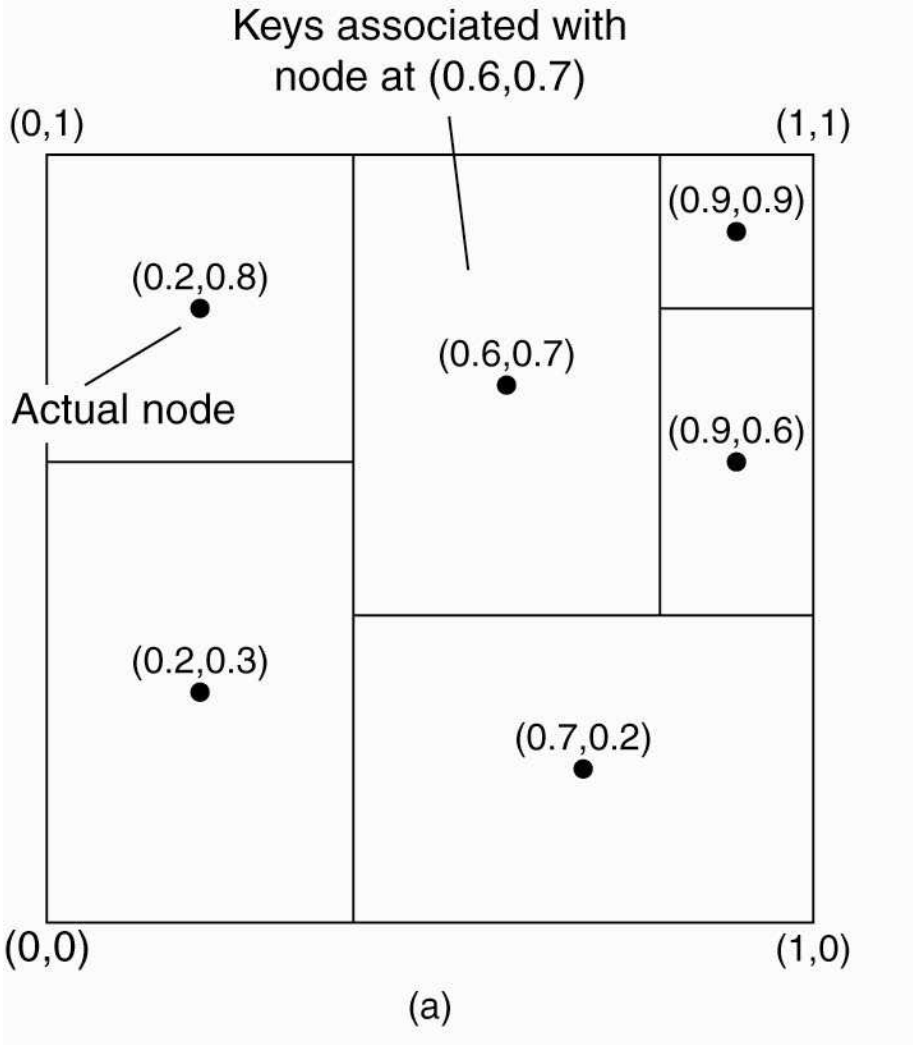

Figure 2-8. (a) The mapping of data items onto nodes in CAN.

Structured Peer-to-Peer Architectures (cont.)

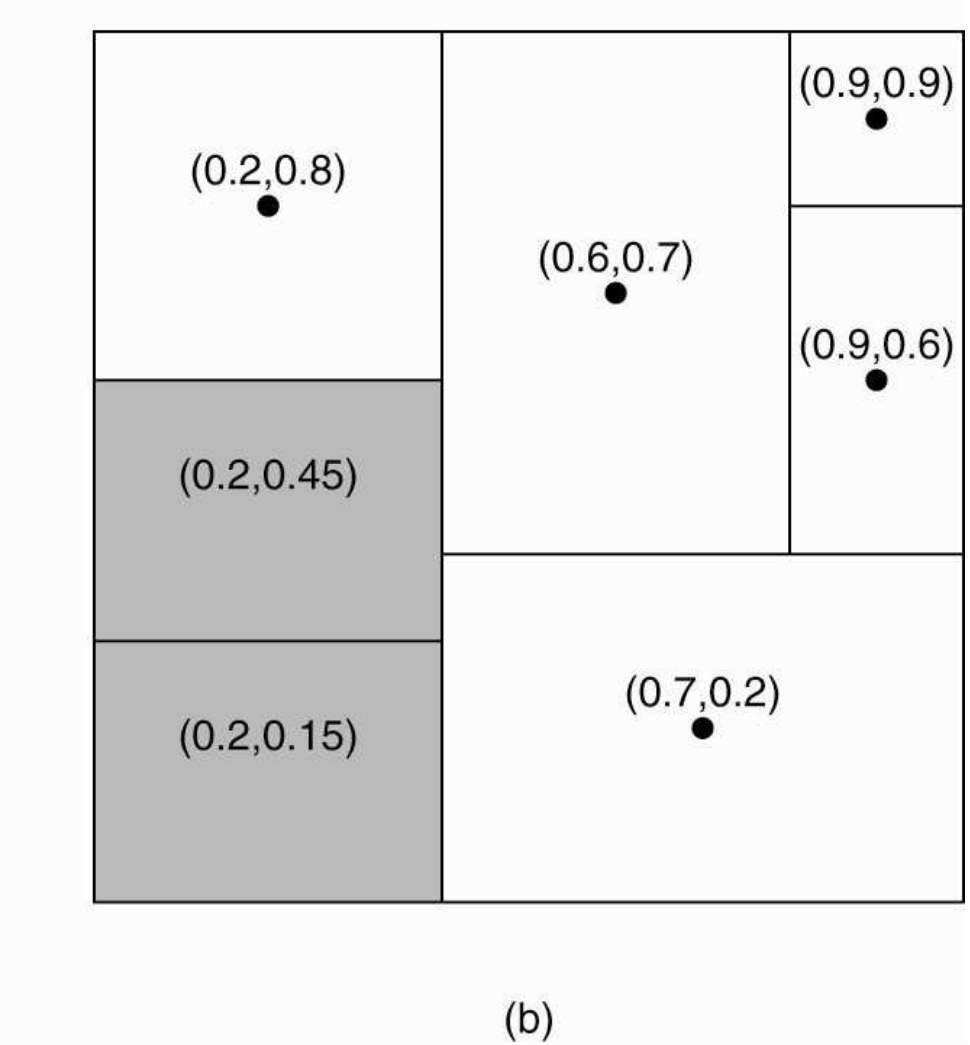

Figure 2-8. (b) Splitting a region when a node joins.

Topology Management of Overlay Networks

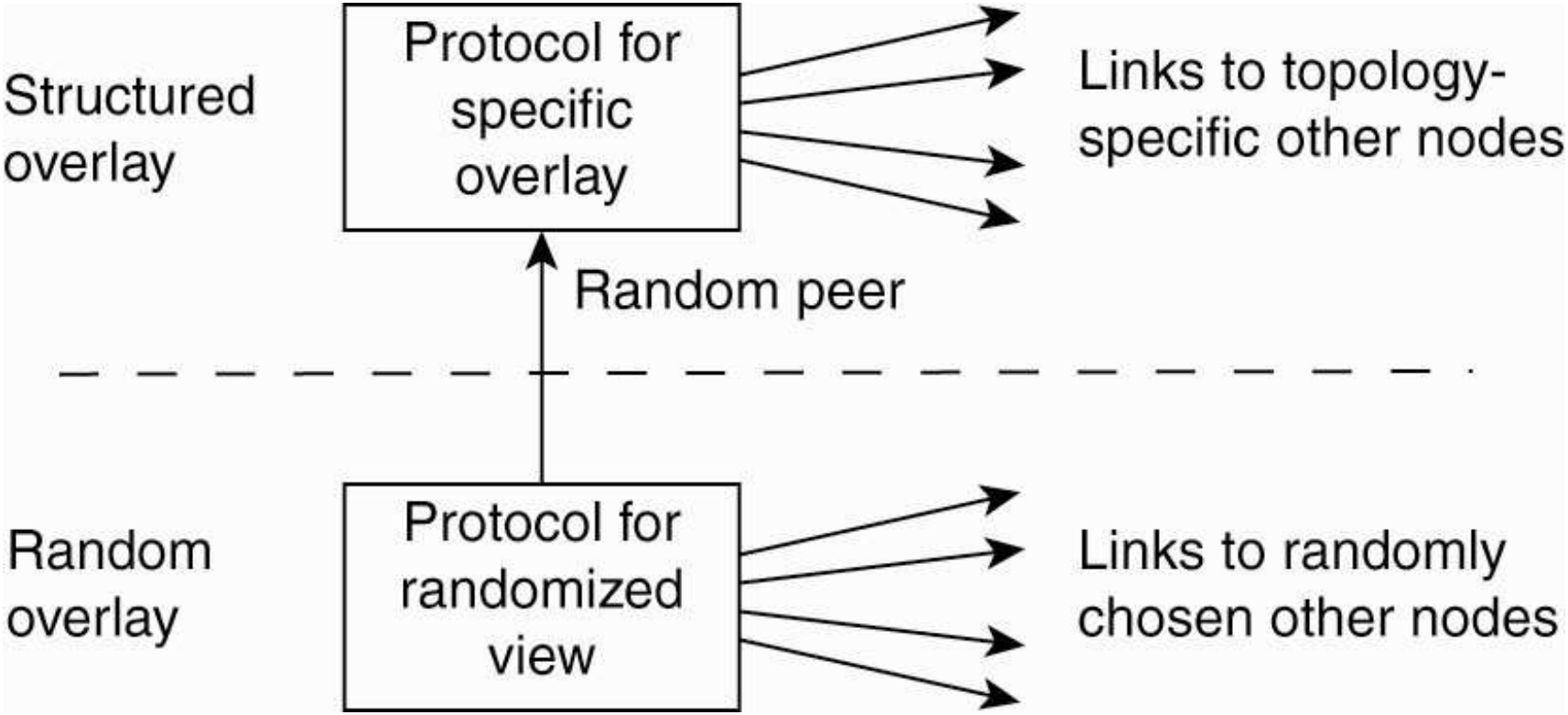

Figure 2-10. A two-layered approach for constructing and maintaining specific overlay topologies using techniques from unstructured peer-to-peer systems.

Topology Management of Overlay Networks (cont.)

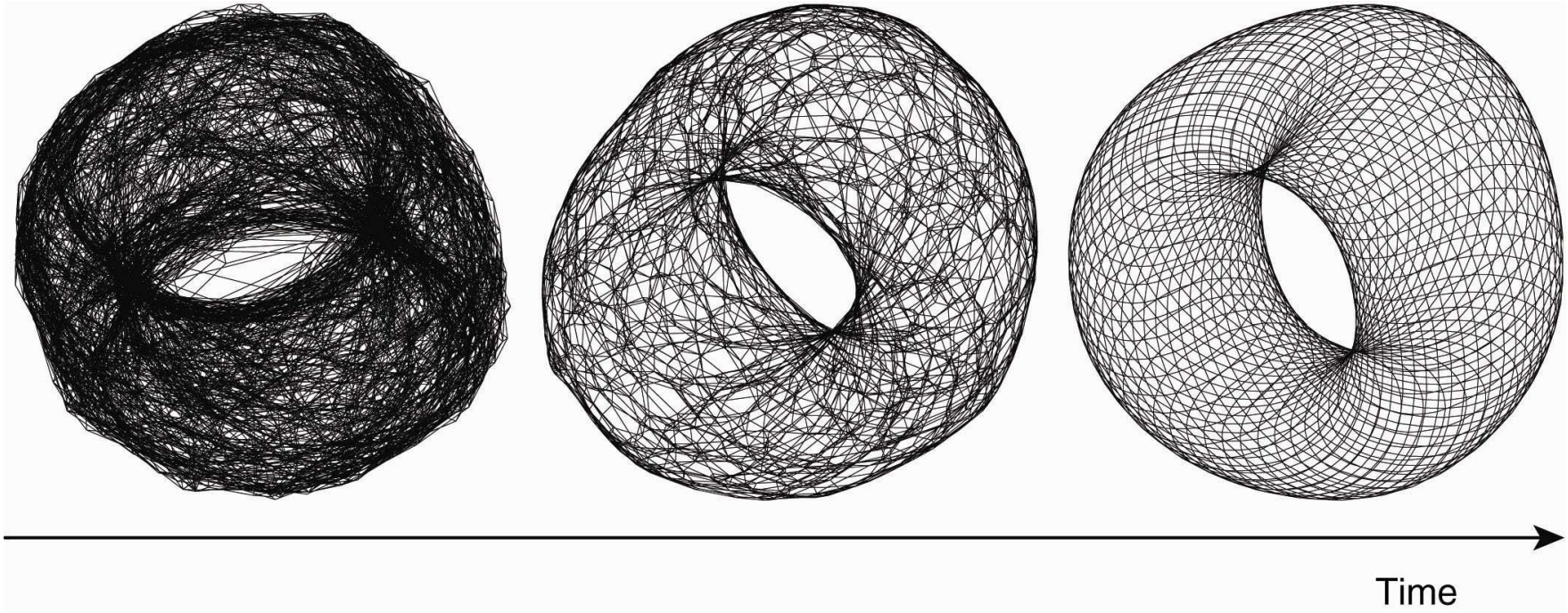

Figure 2-11. Generating a specific overlay network using a two-layered unstructured peer-to-peer system [adapted with permission from Jelasity and Babaoglu (2005)].

Superpeers

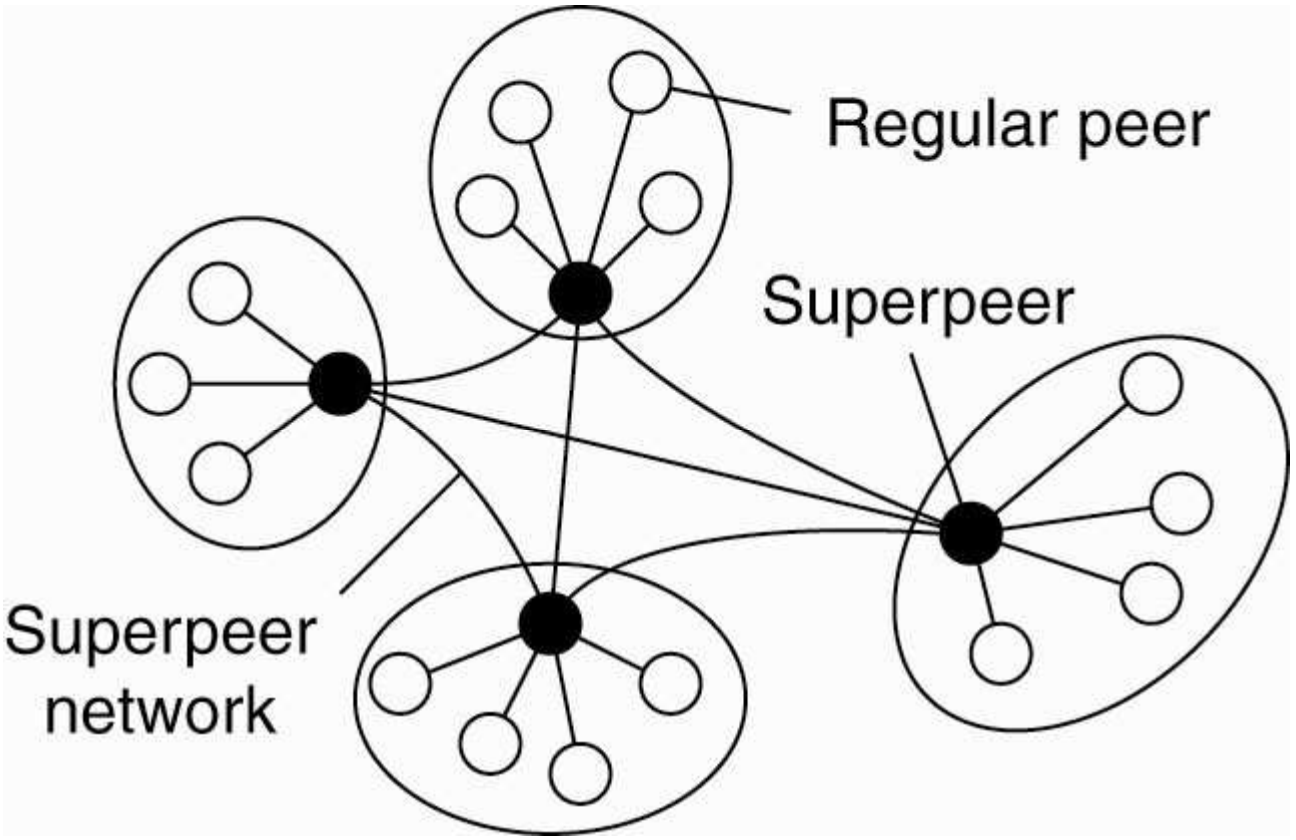

Figure 2-12. A hierarchical organization of nodes into a superpeer network.

Edge-Server Systems

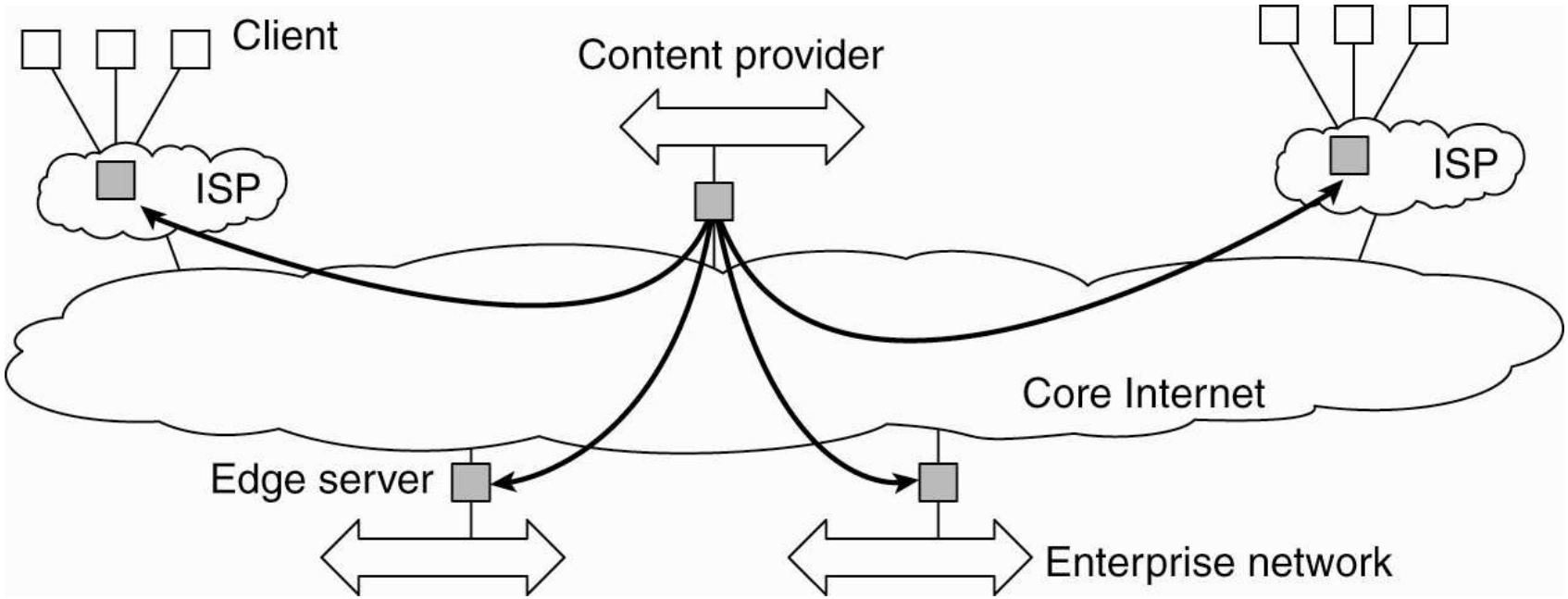

Figure 2-13. Viewing the Internet as consisting of a collection of edge servers.

Collaborative Distributed Systems

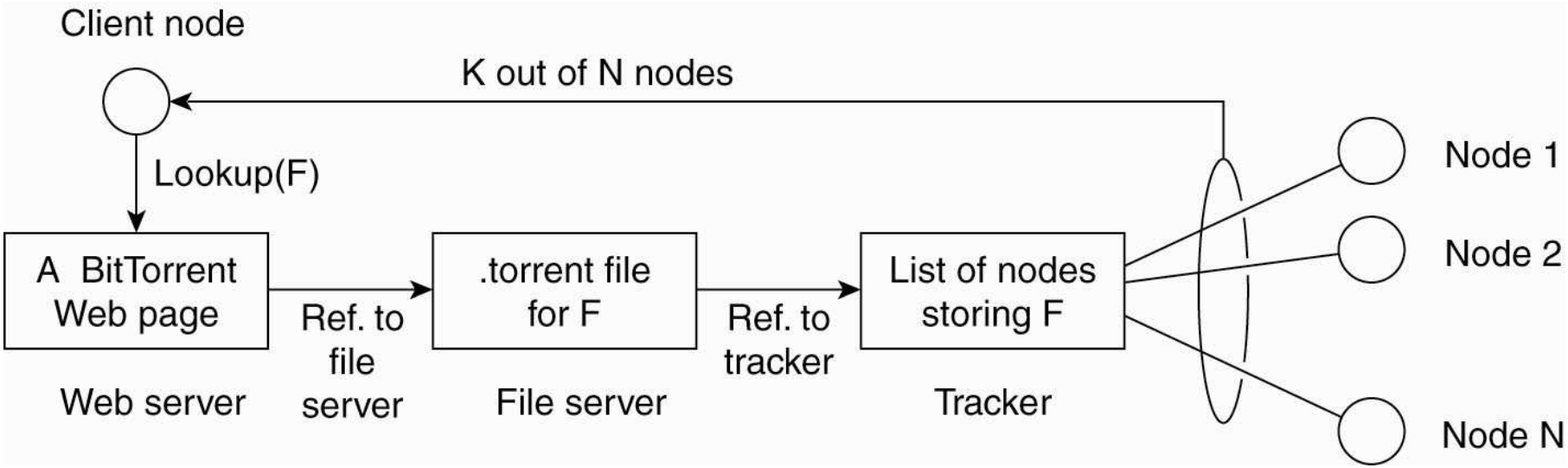

Figure 2-14. The principal working of BitTorrent [adapted with permission from Pouwelse et al. (2004)].

Collaborative Distributed Systems (cont.)

Components of Globule collaborative content distribution network:

- A component that can redirect client requests to other servers.
- A component for analyzing access patterns.
- A component for managing the replication of Web pages.

Interceptors

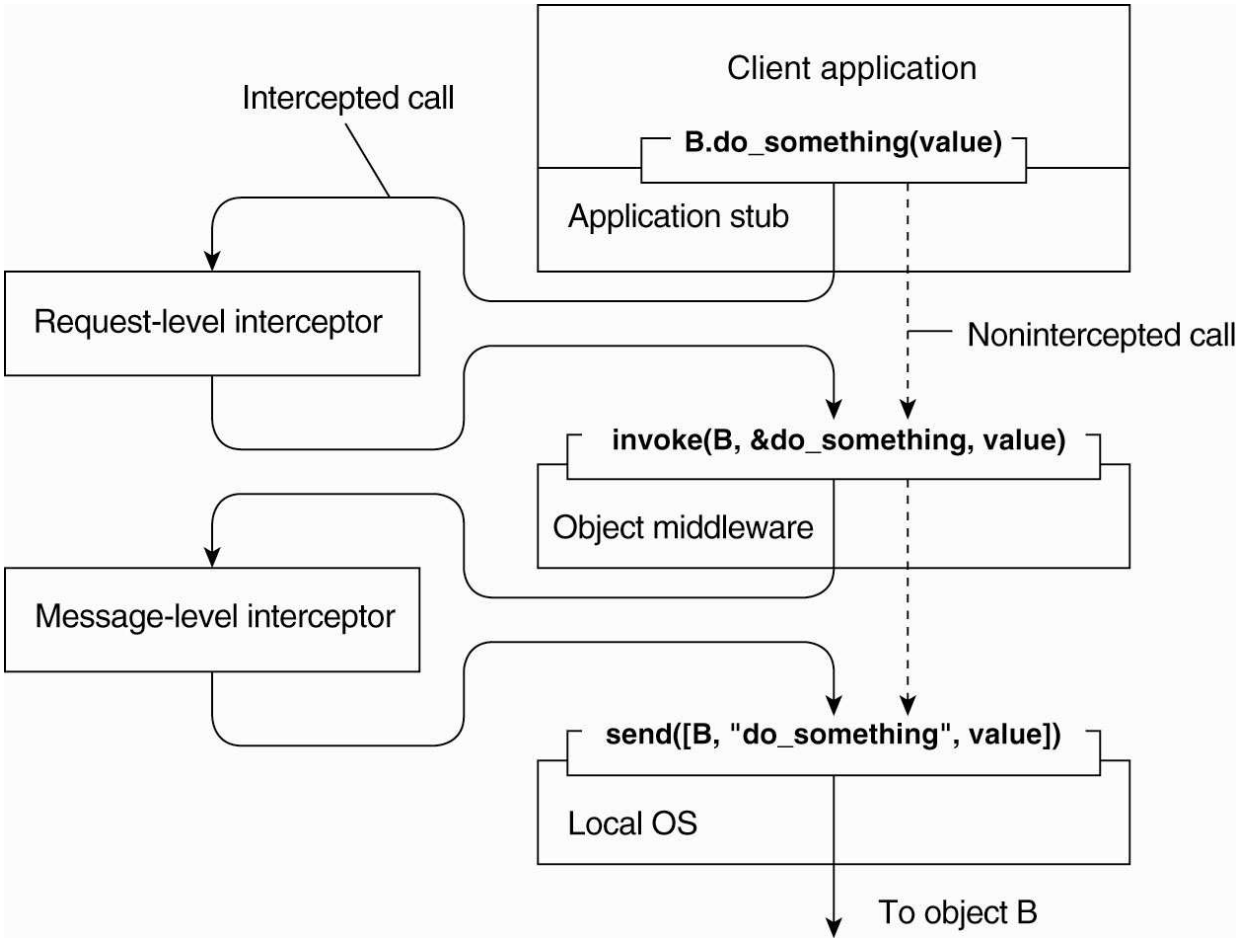

Figure 2-15. Using interceptors to handle remote-object invocations.

General Approaches to Adaptive Software

Three basic approaches to adaptive software:

- Separation of concerns
- Computational reflection
- Component-based design

The Feedback Control Model

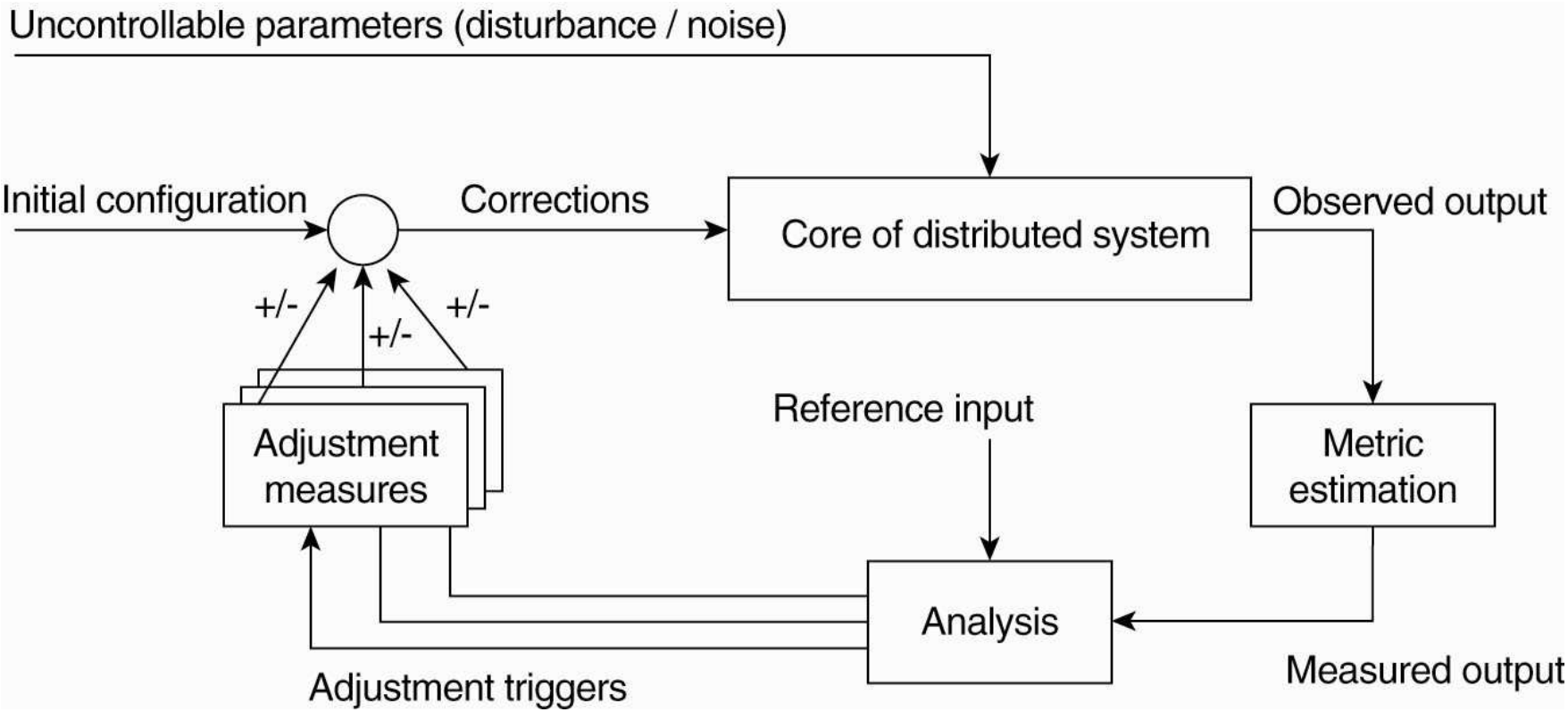

Figure 2-16. The logical organization of a feedback control system.