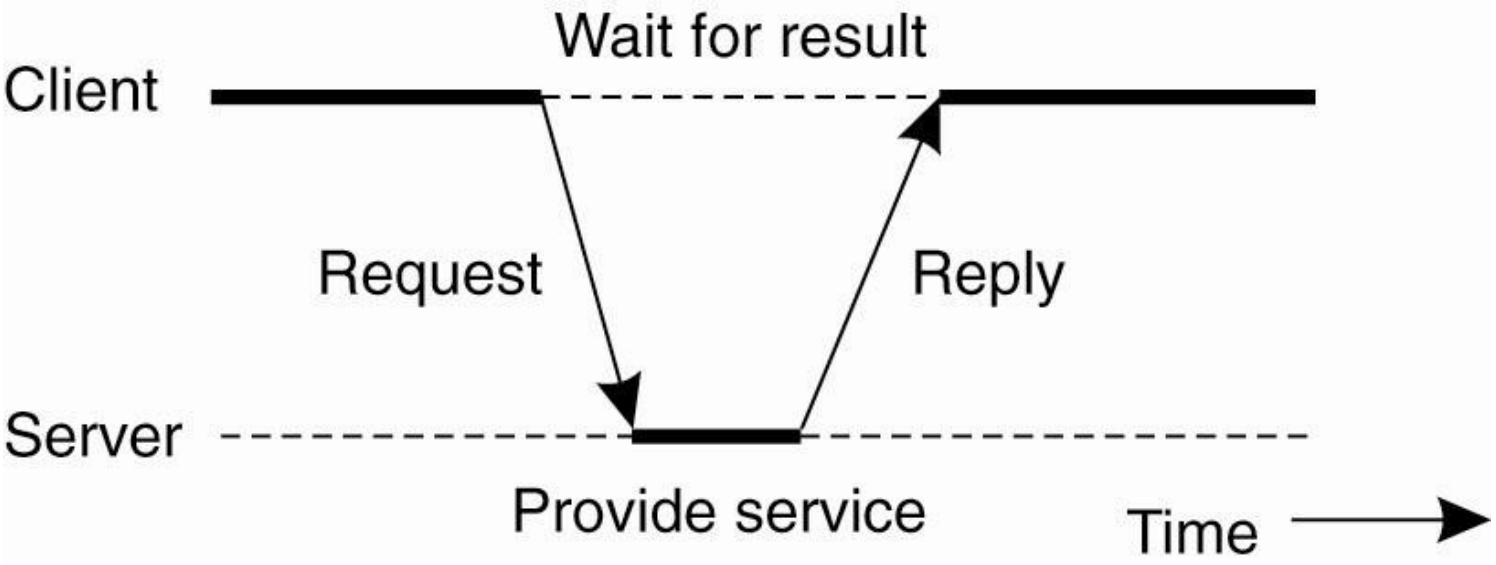
Distributed Computing

Client-server architecture

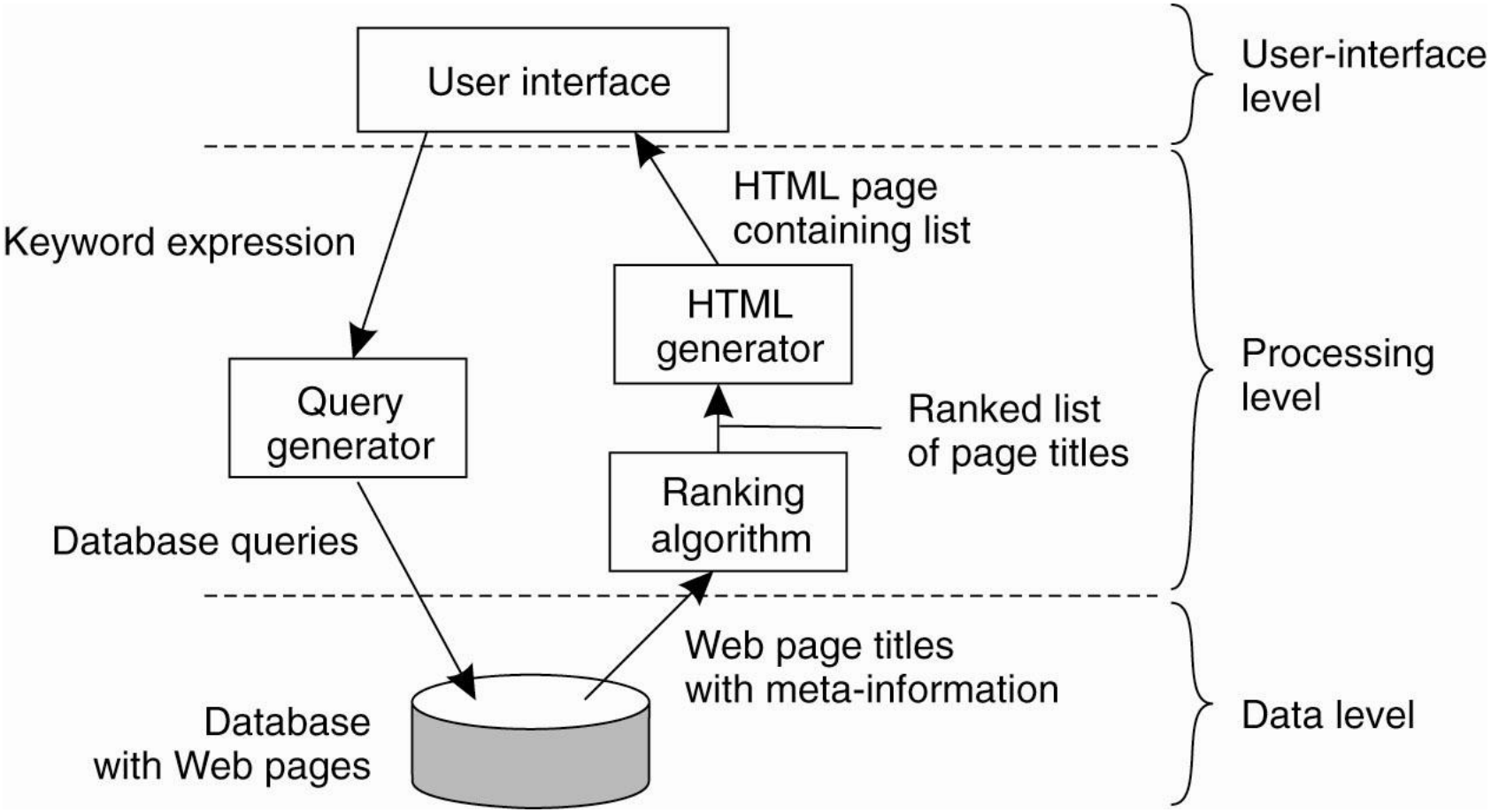
Client-server applications

Issues in Client-Server Application Design

- Many choices arise in the design and implementation of client/server apps
  - Application layering (two vs. three tier)
  - Whether the client is multi-threaded
  - Whether the server is multi-threaded



Application Layering



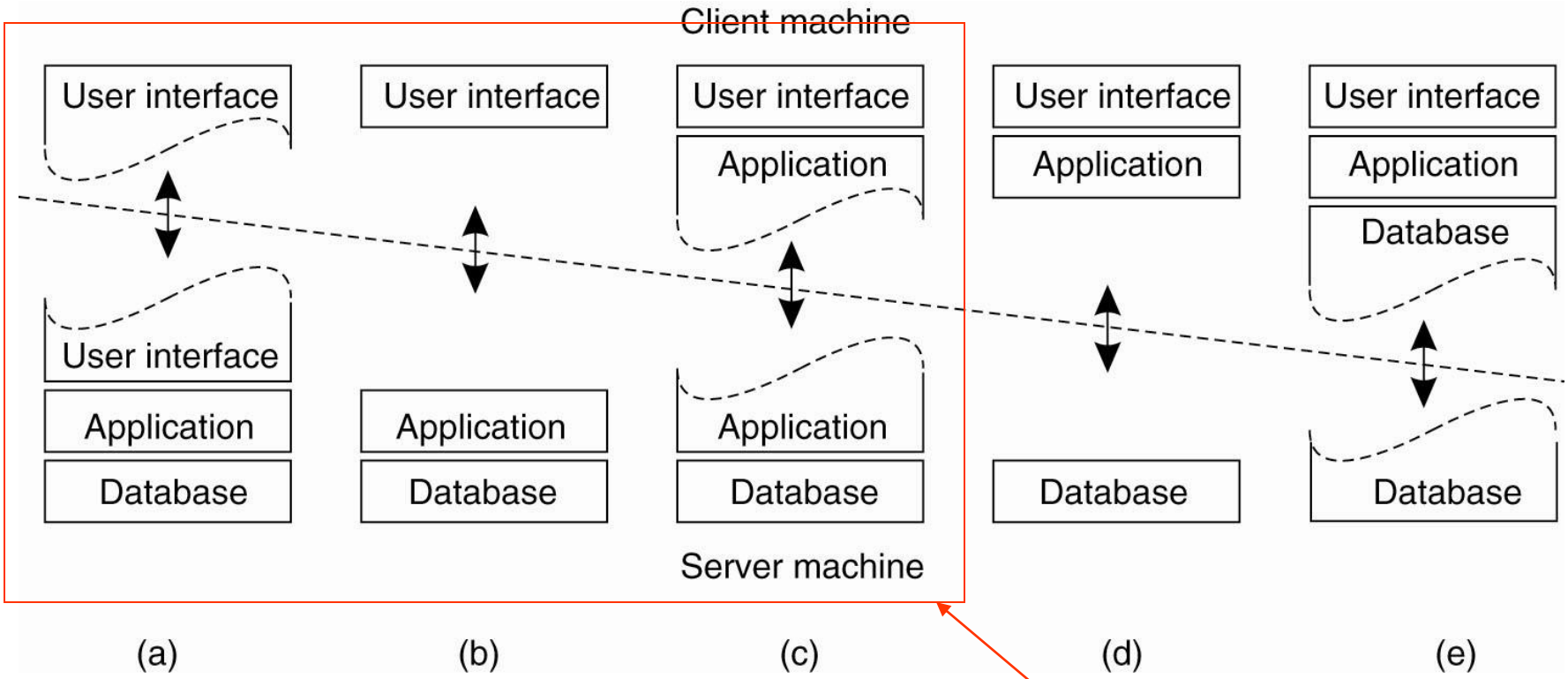
The simplified organization of an Internet search engine into three different layers.

Application Layering (2)

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

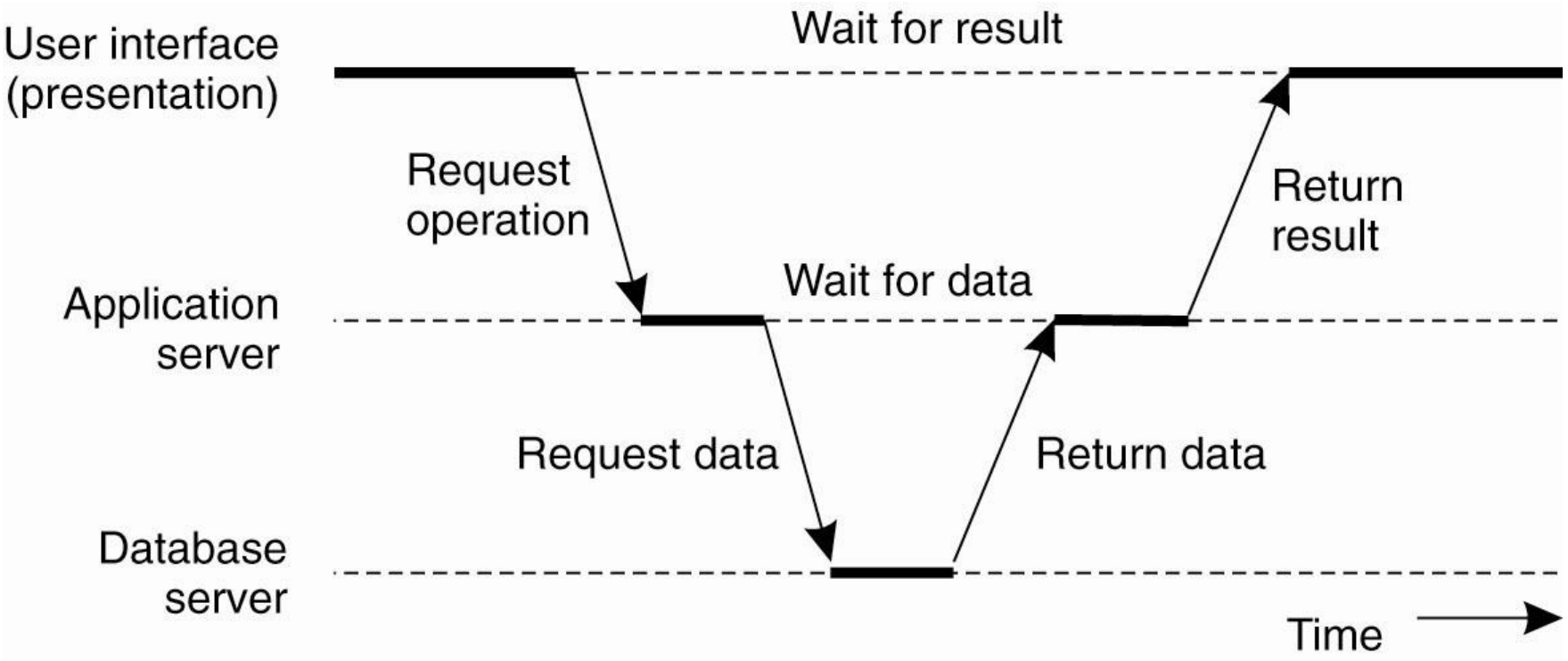
Application Layering (3)



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

thin client

Application Layering (4)

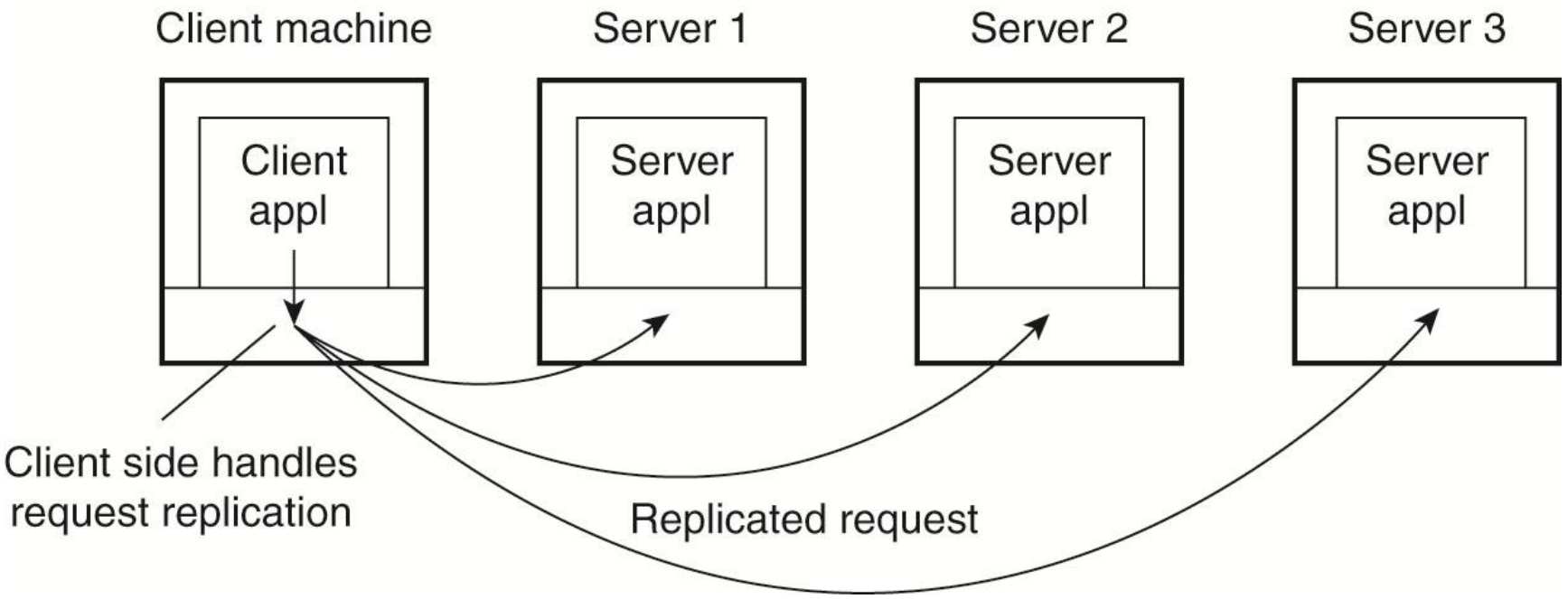


An example of a server acting as client.

Issues in Client design

- Goal: Provide the means for users to interact with remote servers
- Multithreading
  - hide communication latency
  - allow multiple simultaneous connections
- Must know or find out the location of the server
  - known endpoint (port) vs. a lookup mechanism
- Blocking (synchronous) request or non-blocking (asynchronous)
- Replication transparency

Client-Side Software for Distribution Transparency



Transparent replication of a server using a client-side solution.

Issues in Server Design

- **Providing endpoint information**
  - **Known endpoint**
  - **Daemon listening at endpoint**
  - **superserver that spawns threads**
- Connection-oriented or connection-less servers
  - TCP or UDP?
- Concurrent or iterative servers: handle multiple requests concurrently or one after the other?
- Stateful or stateless servers

Issues in Server Design

- Providing endpoint information
  - Known endpoint
  - Daemon listening at endpoint
  - superserver that spawns threads
- **Connection-oriented or connection-less servers**
  - **TCP or UDP?**
- Concurrent or iterative servers: handle multiple requests concurrently or one after the other?
- Stateful or stateless servers

Connection-oriented servers

- Protocol used determines level of reliability
- Overhead of setup and tear down of connections
- TCP provides reliable-data delivery
  - verifies that data arrives at other end, retransmits segments that don't
  - checks that data is not corrupted along the way
  - makes sure data arrives in order
  - eliminates duplicate packets
  - provides flow control to make sure sender does not send data faster than receiver can consume it
  - informs both client and server if underlying network becomes inoperable

Connection-less servers

- UDP unreliable – best effort delivery
- UDP relies on application to take whatever actions are necessary for reliability
- UDP used if
  - application protocol designed to handle reliability and delivery errors in an application-specific manner, e.g. audio and video on the internet
  - overhead of TCP connections too much for application
  - multicast

Issues in Server Design

- Providing endpoint information
  - Known endpoint
  - Daemon listening at endpoint
  - superserver that spawns threads
- Connection-oriented or connection-less servers
  - TCP or UDP?
- Concurrent or iterative servers: handle multiple requests concurrently or one after the other?
- **Stateful or stateless servers**

Stateful vs Stateless servers

- State ≡ Information that server maintains about the status of ongoing interactions with clients
- Stateful servers
  - client state information maintained can help server in performing request faster
  - state information needs to be preserved across (or reconstructed after) crashes
- Stateless servers
  - information on clients not maintained and can change state without having to inform clients
  - quicker and more reliable recovery after crashes
  - smaller memory requirements
  - Application protocol should have ***idempotent*** operations (operations that can be repeated multiple times without harm)