Distributed Systems

Techniques for scaling

Replication and caching: Make copies of data available at different machines

- Replicated file servers and databases
- Mirrored Web sites
- Web caches (in browsers and proxies)
- File caching (at server and client)

Scaling: The problem with replication

Applying replication is easy, except for one thing

- Having multiple copies (cached or replicated), leads to inconsistencies:
- Modifying one copy makes that copy different from the rest.
- Always keeping copies consistent and in a general way requires global synchronization on each modification.
- Global synchronization precludes large-scale solutions.

Developing distributed systems

False (and often hidden) assumptions

- The network is reliable
- The network is secure
- The network is homogeneous
- The topology does not change
- Latency is zero
- Bandwidth is infinite
- Transport cost is zero
- There is one administrator

Three types of distributed systems

- High performance distributed computing systems

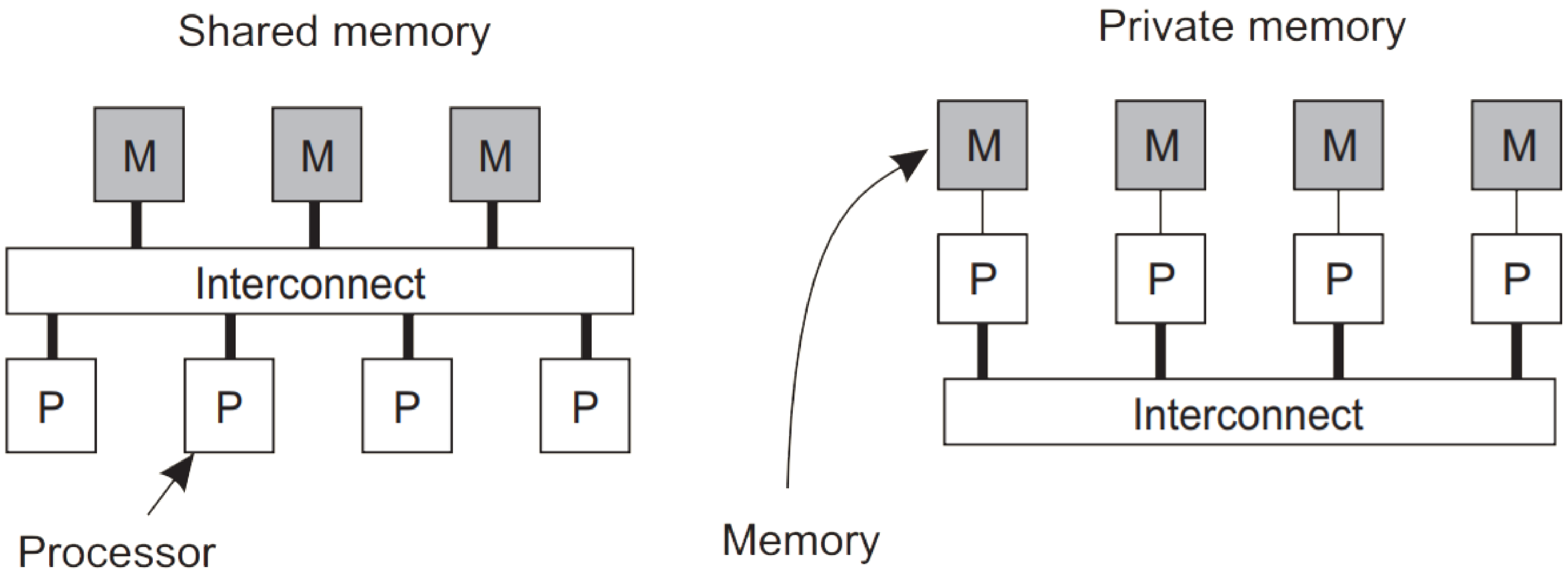
  HDCS
- Distributed information systems

  DIS
- Distributed systems for pervasive computing

  DSPC

Parallel computing

- High-performance distributed computing started with parallel computing



Distributed shared memory systems

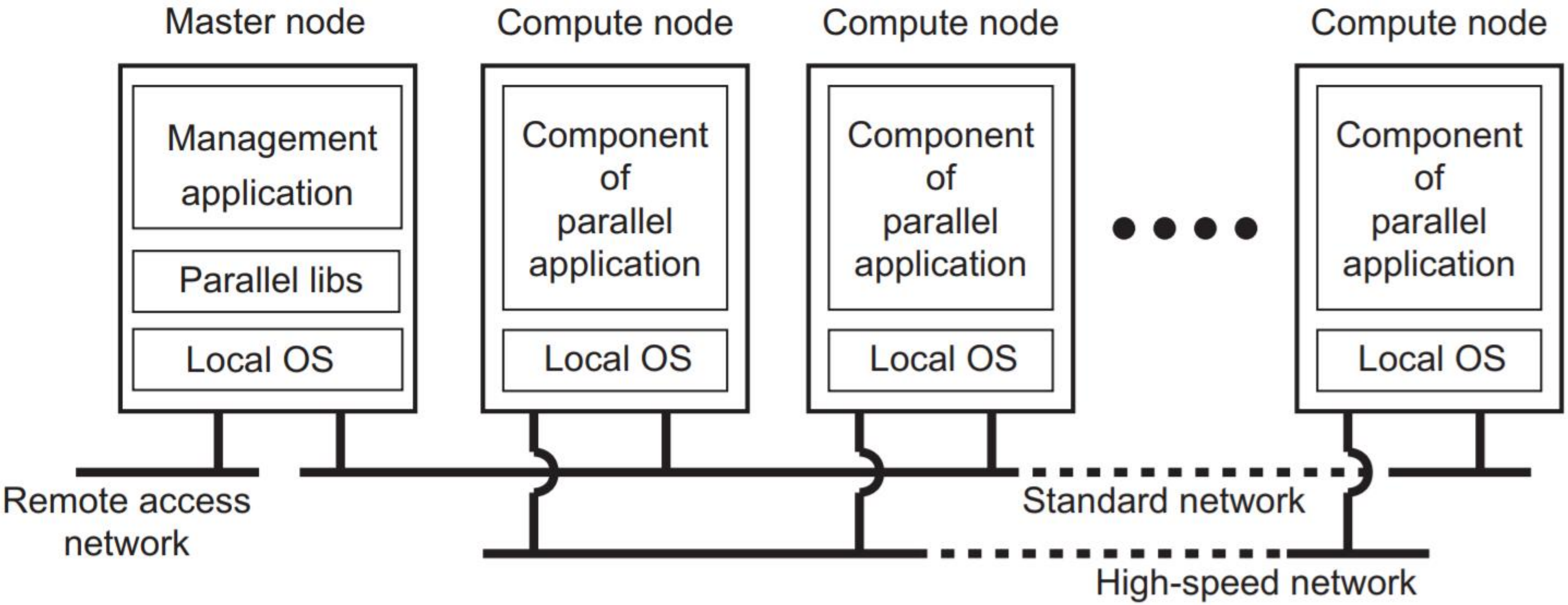
- Multiprocessors are relatively easy to program
- In comparison to multicomputers,
- yet have problems when increasing the number of processors (or cores).
- Solution: Try to implement a shared-memory model on top of a multicomputer.

Problem

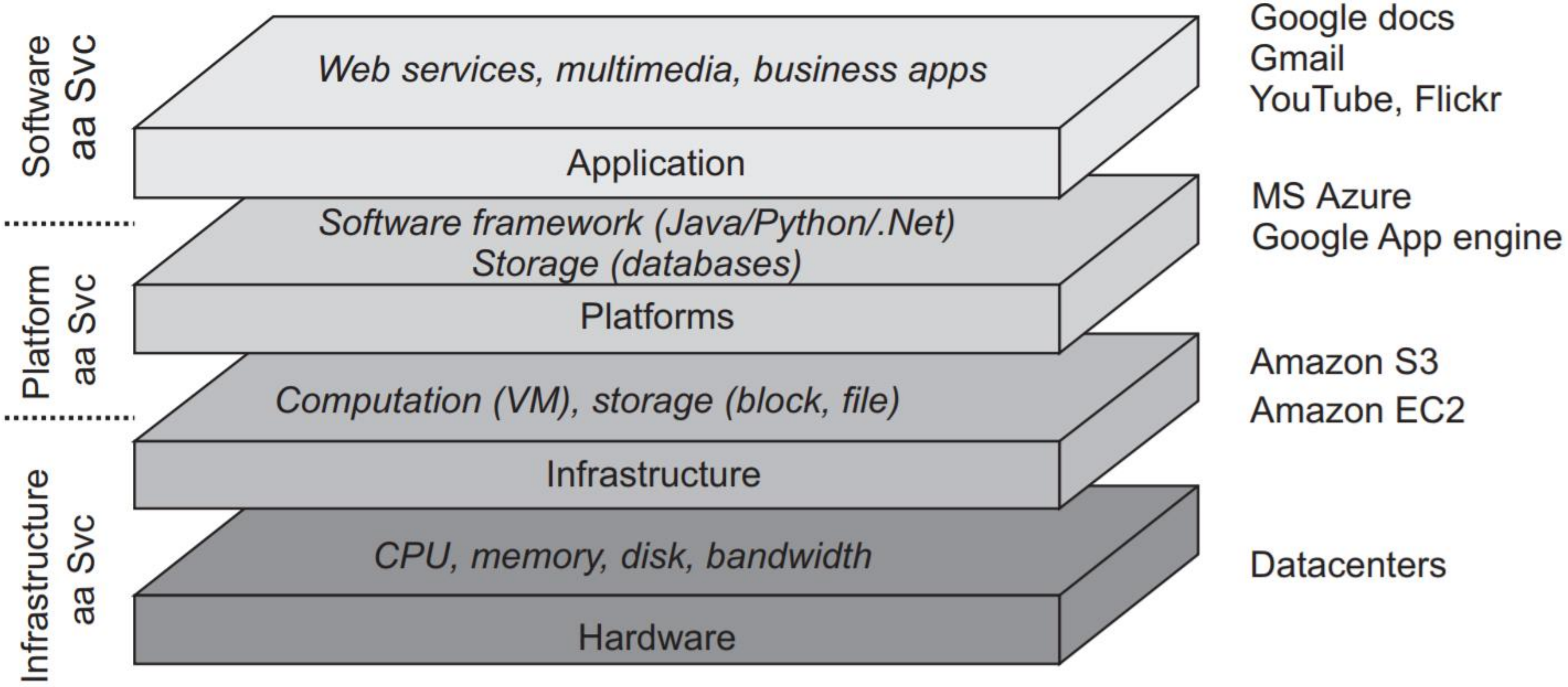
- Performance of distributed shared memory could never compete with that of multiprocessors, and failed to meet the expectations of programmers.
- It has been widely abandoned by now.

Cluster computing

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



Cloud computing



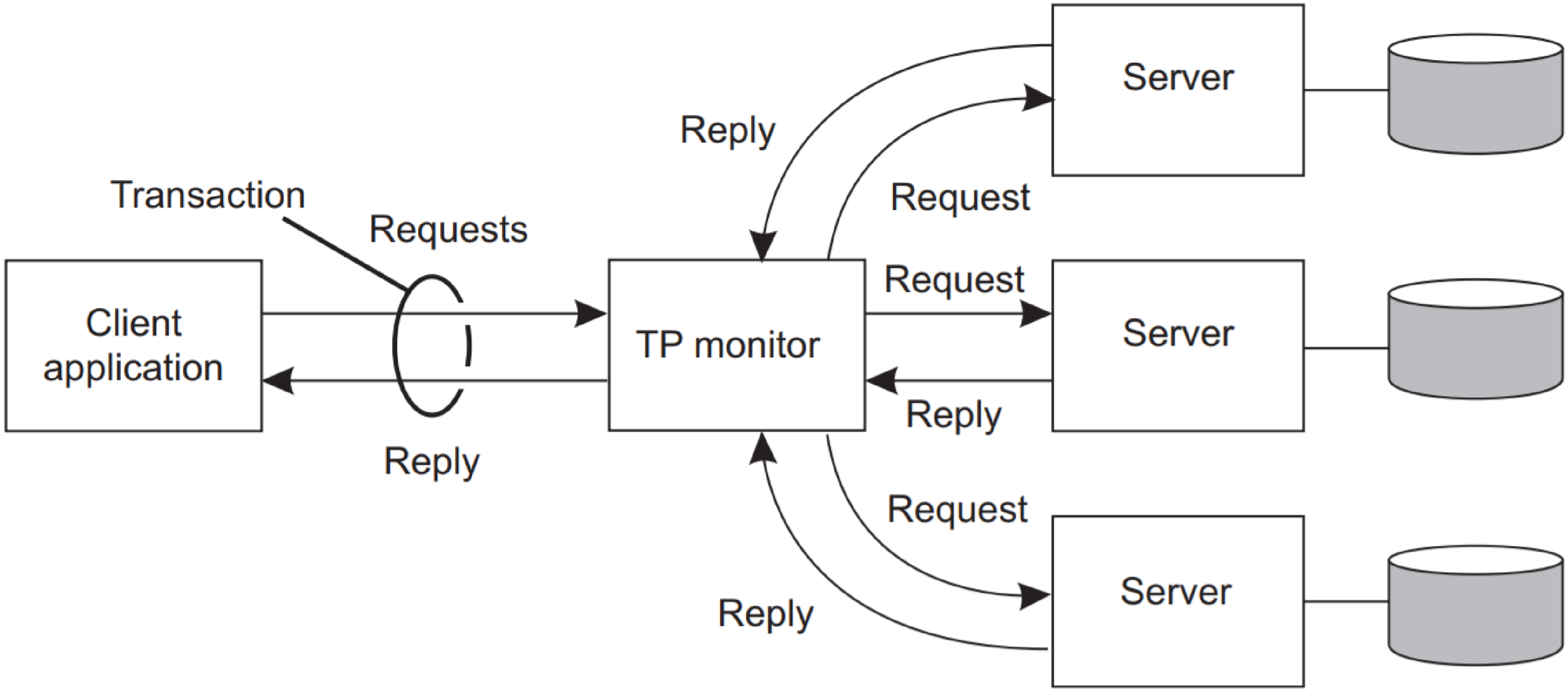
Cloud computing

- Hardware: Processors, routers, power and cooling systems. Customers normally never get to see these.
- Infrastructure: Deploys virtualization techniques. Evolves around allocating and managing virtual storage devices and virtual servers.
- 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 so-called buckets.
- Application: Actual applications, such as office suites (text processors, spreadsheet applications, presentation applications). Comparable to the suite of apps shipped with OSes.

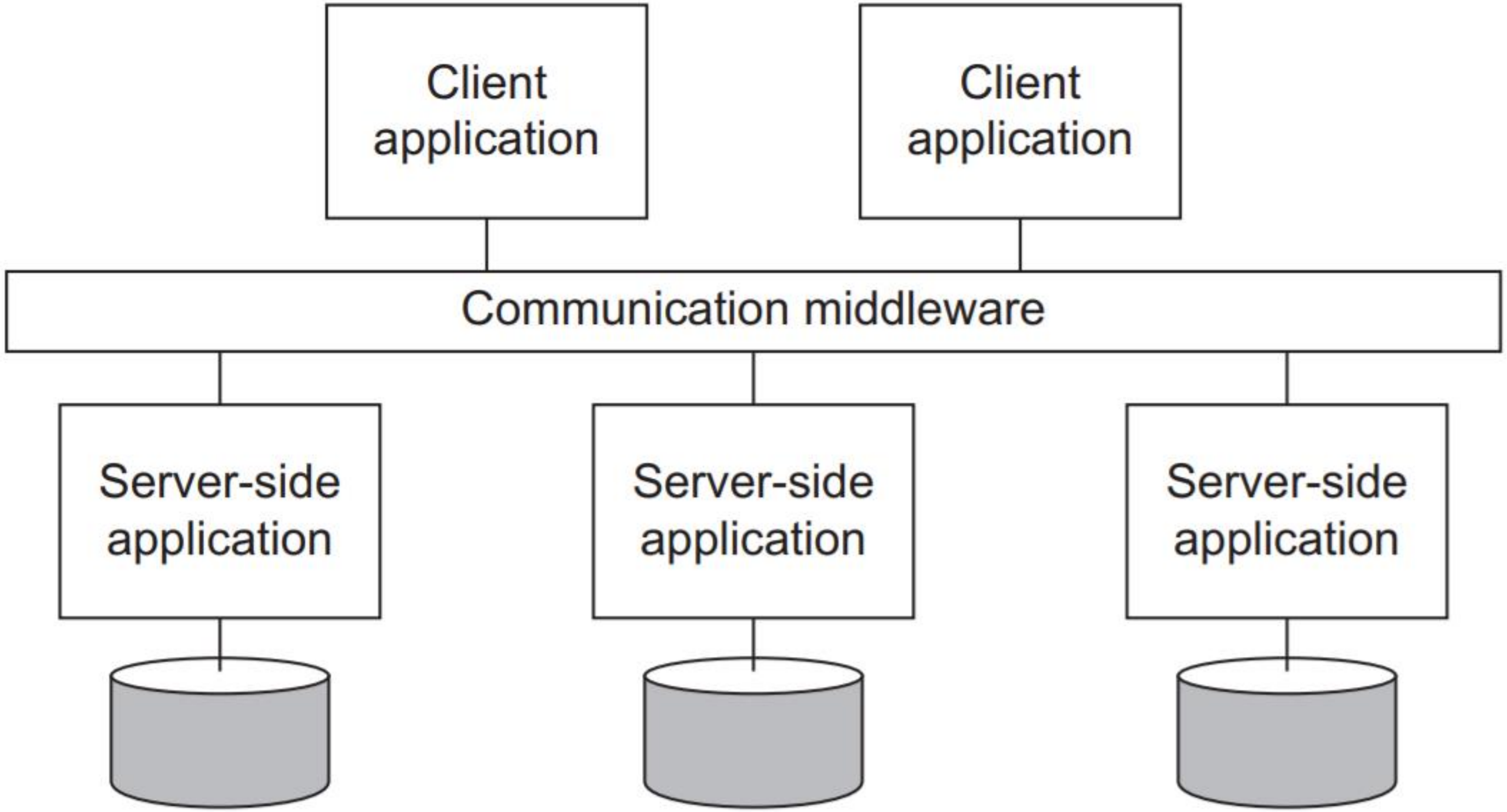
Integrating applications

- 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 that off; collect responses, and present a coherent result to the user.
- Allow direct application-to-application communication, leading to Enterprise Application Integration.

TPM: Transaction Processing Monitor



Middleware


Client application
Client application
Communication middleware
Server-side application
Server-side application
Server-side application

Middleware

Remote Procedure Call (RPC):

- Requests are sent through local procedure call, packaged as message, processed, responded through message, and result returned as return from call.

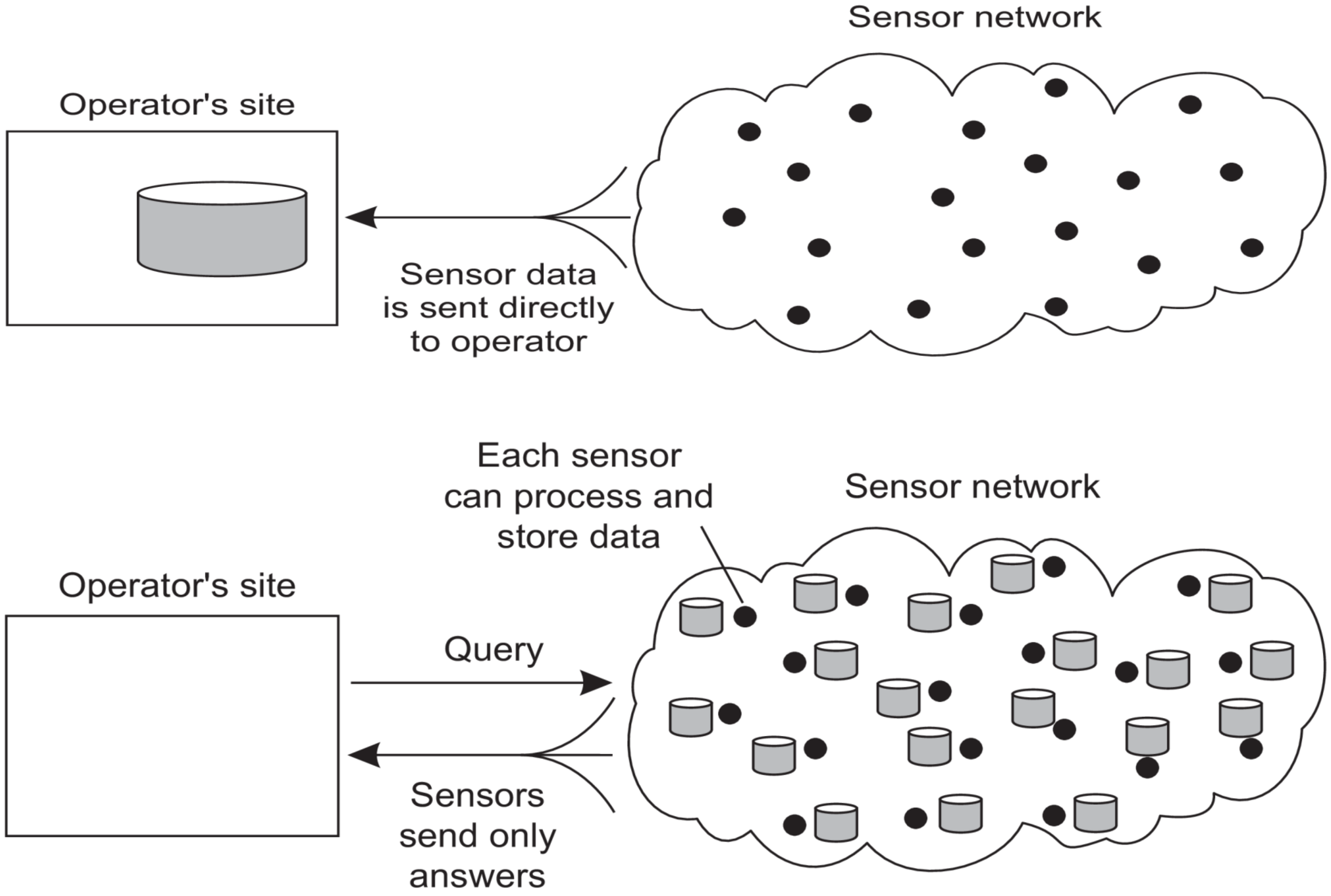
Message Oriented Middleware (MOM):

- Messages are sent to logical contact point (published), and forwarded to subscribed applications.

Distributed pervasive systems

- Ubiquitous computing systems: pervasive and continuously present, i.e., there is a continuous interaction between system and user.
- Mobile computing systems: pervasive, but emphasis is on the fact that devices are inherently mobile.
- Sensor (and actuator) networks: pervasive, with emphasis on the actual (collaborative) sensing and actuation of the environment.

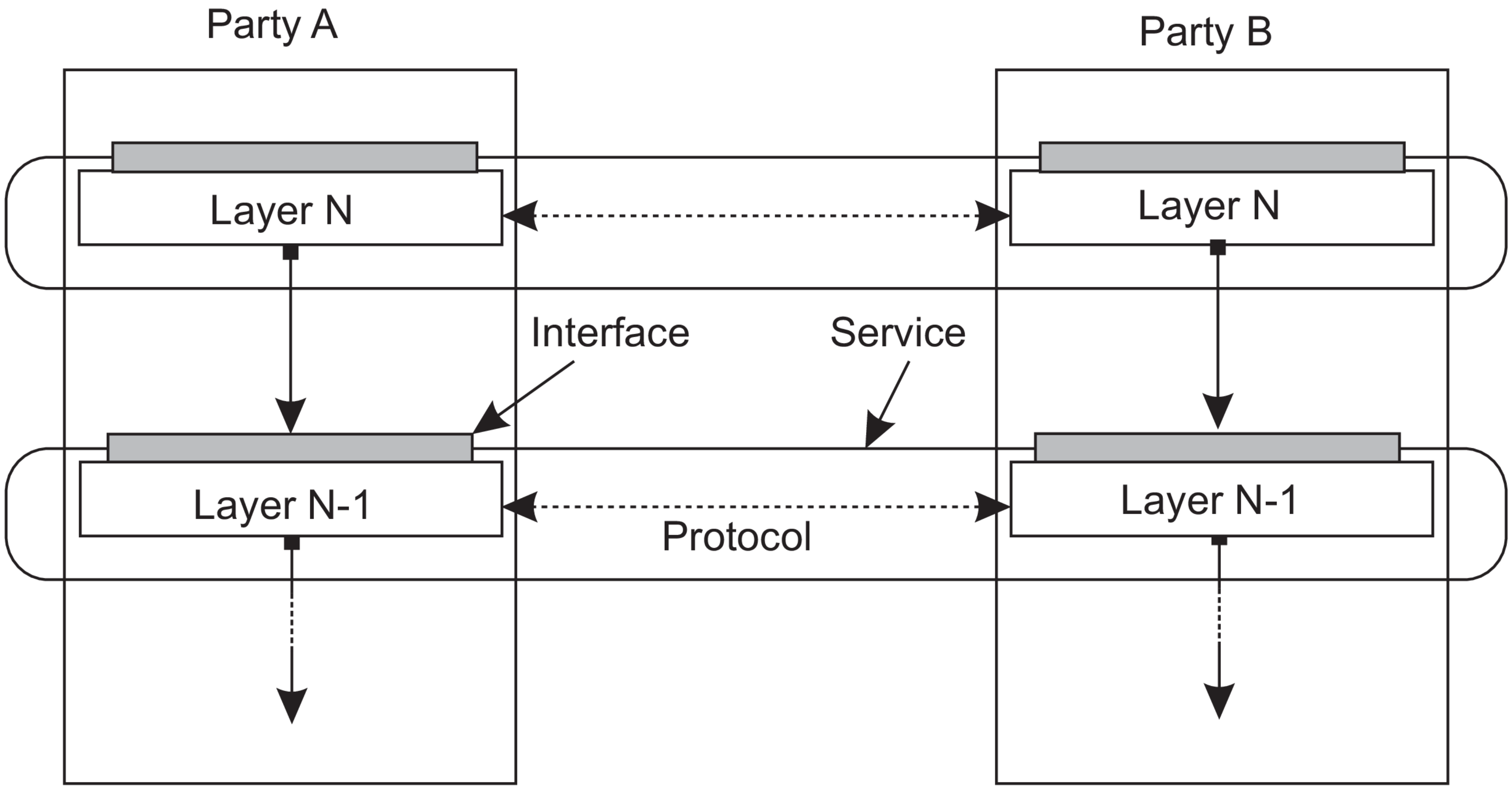
Sensor networks as distributed databases



Architectural styles

- (replaceable) components with well-defined interfaces
- the way that components are connected to each other
- the data exchanged between components
- how these components and connectors are jointly configured into a system.

Example: communication protocols

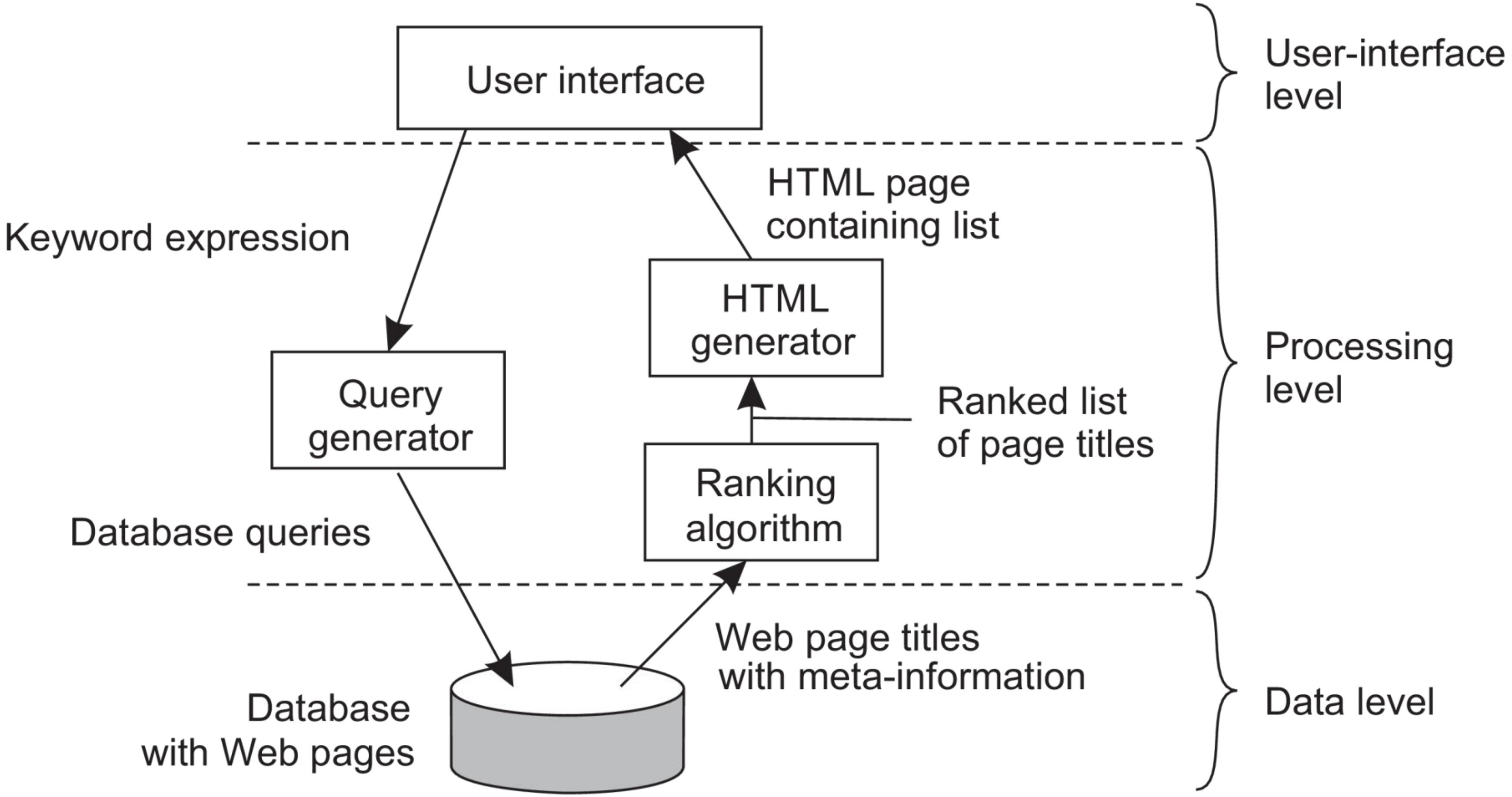


Application Layering

- Application-interface layer contains units for interfacing to users or external applications
- 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

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

Application Layering



RESTful architectures

- View a distributed system as a collection of resources, individually managed by components.
- Resources may be added, removed, retrieved, and modified by (remote) applications.

Operation	Description
POST	Create a new resource
GET	Retrieve the state of a resource in some representation
DELETE	Delete a resource
PUT	Modify a resource by transferring a new state