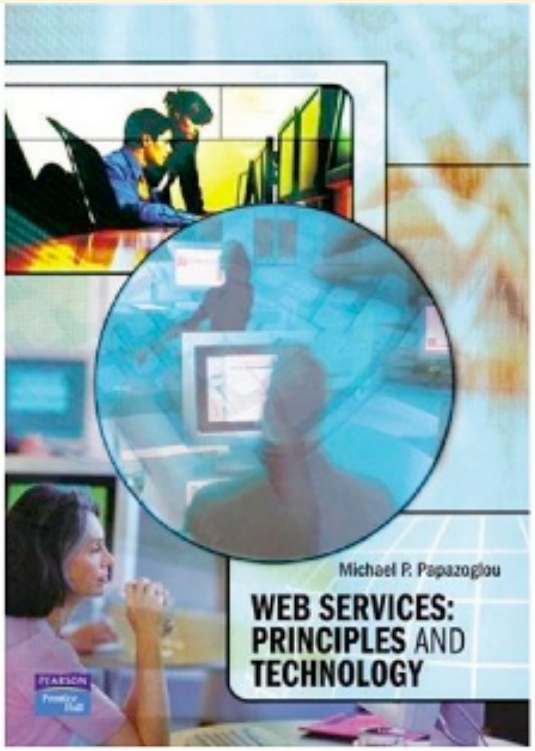
Web Services:
Principles & Technology

Chapter 2
Distributed Computing Infrastructure

Mike P. Papazoglou
mikep@uvt.nl



UNIVERSITEIT VAN TILBURG

Topics

- *Distributed computing and Internet protocols*
- *The client–server model*
- *Inter-process communication*
- *Synchronous forms of middleware*
- *Asynchronous forms of middleware*
- *Request–reply messaging*
- *Message-oriented middleware*
- *Enterprise application and e-Business integration*

Distributed Computing

- A distributed system is characterized as a collection of heterogeneous networked computers, which communicate and coordinate their actions by passing messages.
  - Distribution is transparent to the user so that the system appears as a single integrated facility.

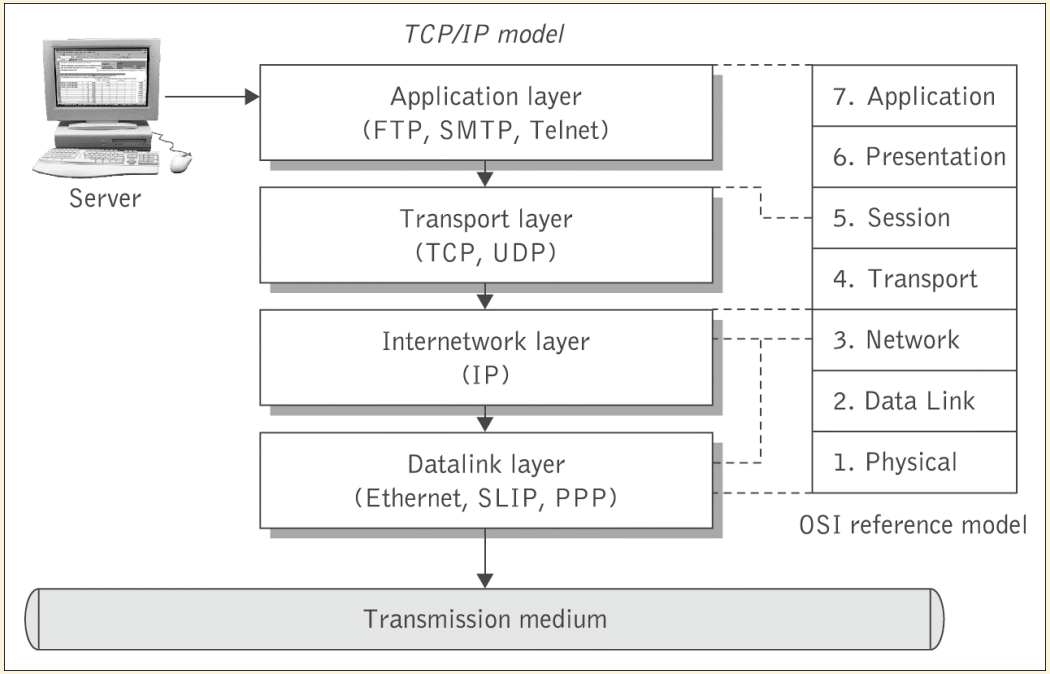
- One important characteristic of a distributed system is that processes are not executed on a single processor, but rather span a number of processors.
  - This requires inter-process communication mechanisms.

Internet Protocols

- Internet protocols are essentially methods of data transport across the Internet. They define the standards by which the different components in a distributed system communicate across the Internet with each other & with remote components.
- The most prominent of the Internet protocols is transport control protocol over Internet protocol (or TCP/IP), which provide for the reliable delivery of streams of data from one host to another across the Internet:
  - The Internet protocol (IP) enables the unreliable delivery of individual packets from one host to another.
    - IP makes no guarantees as to whether the packet will be delivered, how long it will take, or if multiple packets will arrive in the order they were sent.
  - The transport control protocol (TCP) adds the notions of connection and reliability.

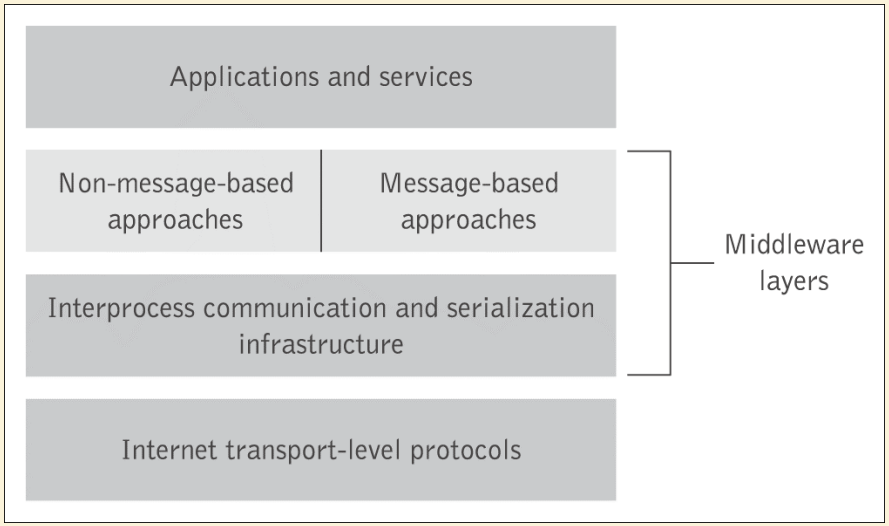
The TCP/IP protocol stack and its relation to the ISO Reference Model

- The data link layer provides the interface to the actual network hardware.
- The inter-network layer is responsible for routing “blocks of data” from one host to another.
- The transport layer provides end-to-end data transfer by delivering data between the client and server sides of an application.
- The application layer is responsible for supporting network applications.



Middleware

- Middleware provides a functional set of interfaces to allow an application to
  - locate applications transparently across the network;
  - shield software developers from low-level, tedious and error-prone platform details;
  - provide a consistent set of higher level abstractions that are much closer to application requirements;
  - leverage previous developments and reuse them;
  - provide services such as reliability, availability, authentication, and security.

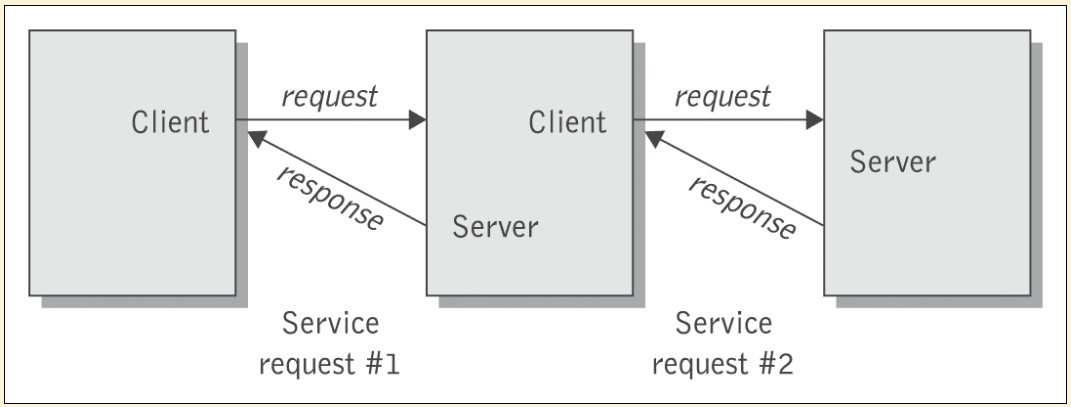


Topics

- *Distributed computing and Internet protocols*
- *The client–server model*
- *Inter-process communication*
- *Synchronous forms of middleware*
- *Asynchronous forms of middleware*
- *Request–reply messaging*
- *Message-oriented middleware*
- *Enterprise application and e-Business integration*

Client–server model

- A client/server architecture is an architecture in which processing and storage tasks are divided between two classes of network members, clients & servers.
- Client/server architecture involves client processes (service consumers) requesting service from server processes (service providers). Servers may in turn be clients of other servers.
  - The client machine runs software and applications that are stored locally. The client makes requests to servers and is also responsible for the user interface.
  - Some of the applications may be stored and executed on the server, but most of it is on the client. The server also provides the data for the application.



Source: From M. P. Papazoglou and P. M. A. Ribbers, *e-Business: Organizational and Technical Foundations*, J. Wiley & Sons, 2006. Reproduced with permission

Topics

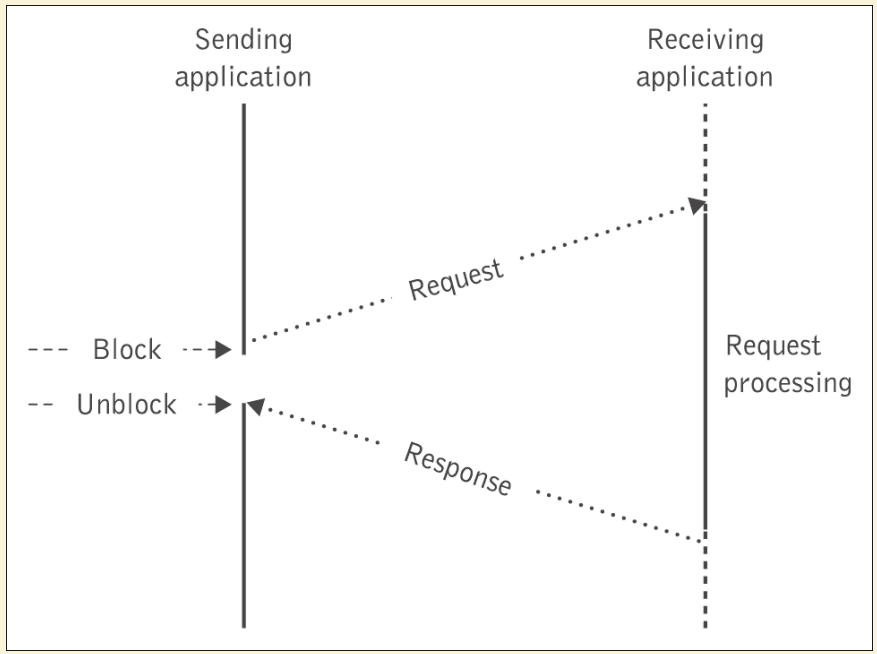
- *Distributed computing and Internet protocols*
- *The client–server model*
- *Inter-process communication*
- *Synchronous forms of middleware*
- *Asynchronous forms of middleware*
- *Request–reply messaging*
- *Message-oriented middleware*
- *Enterprise application and e-Business integration*

Messaging

- Distributed systems and applications communicate by exchanging messages. Messaging enables high-speed, asynchronous, program-to-program communication with reliable delivery.
- Message passing between a pair of processes is supported by two message communication operations: *send* and *receive*, defined in terms of destinations and messages.
- *Marshalling (serialization)* is the process of taking any form of structured data items and breaking up so that it can be transmitted as a stream of bytes over a communications network in such a way that the original structure can be reconstructed easily on the receiving end.
- *Unmarshalling (deserialization)* is the process of converting the assembled stream of bytes on arrival to produce an equivalent form of structured data at the destination point.

Synchronous and asynchronous messaging

- There are two basic modes of message communication:
- *Synchronous* communication – synchronized between two communicating application systems, which must both be up and running.
  - Execution flow at the client's side is interrupted to execute the call.
- *Asynchronous* communication – the caller employs a send and forget approach that allows it to continue to execute after it sends the message.
  - Here an application sends a request to another while it continues its own processing activities.

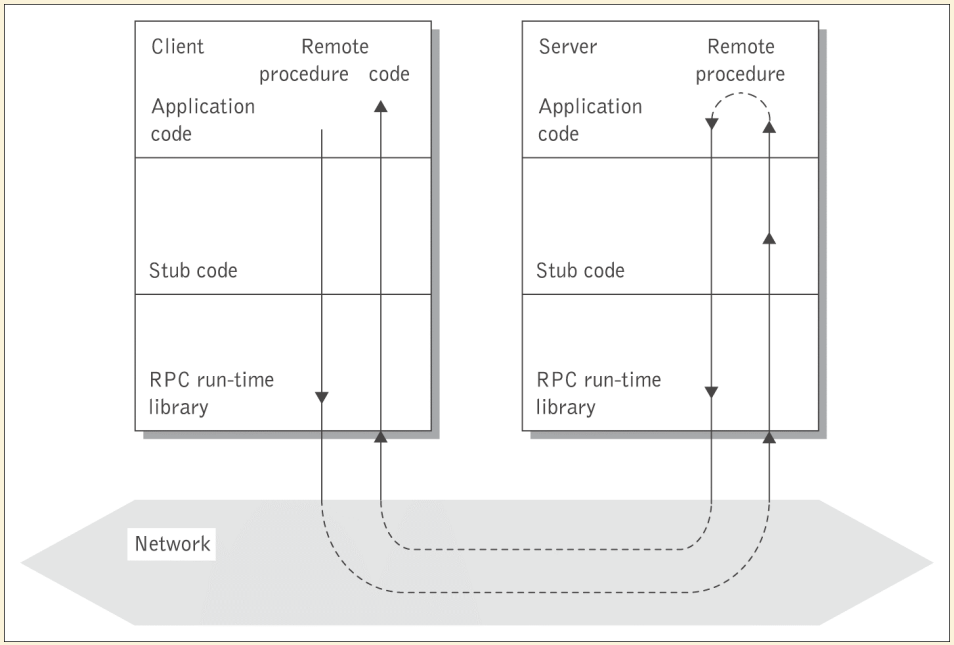


Topics

- *Distributed computing and Internet protocols*
- *The client–server model*
- *Inter-process communication*
- *Synchronous forms of middleware*
- *Asynchronous forms of middleware*
- *Request–reply messaging*
- *Message-oriented middleware*
- *Enterprise application and e-Business integration*

Remote procedure calls

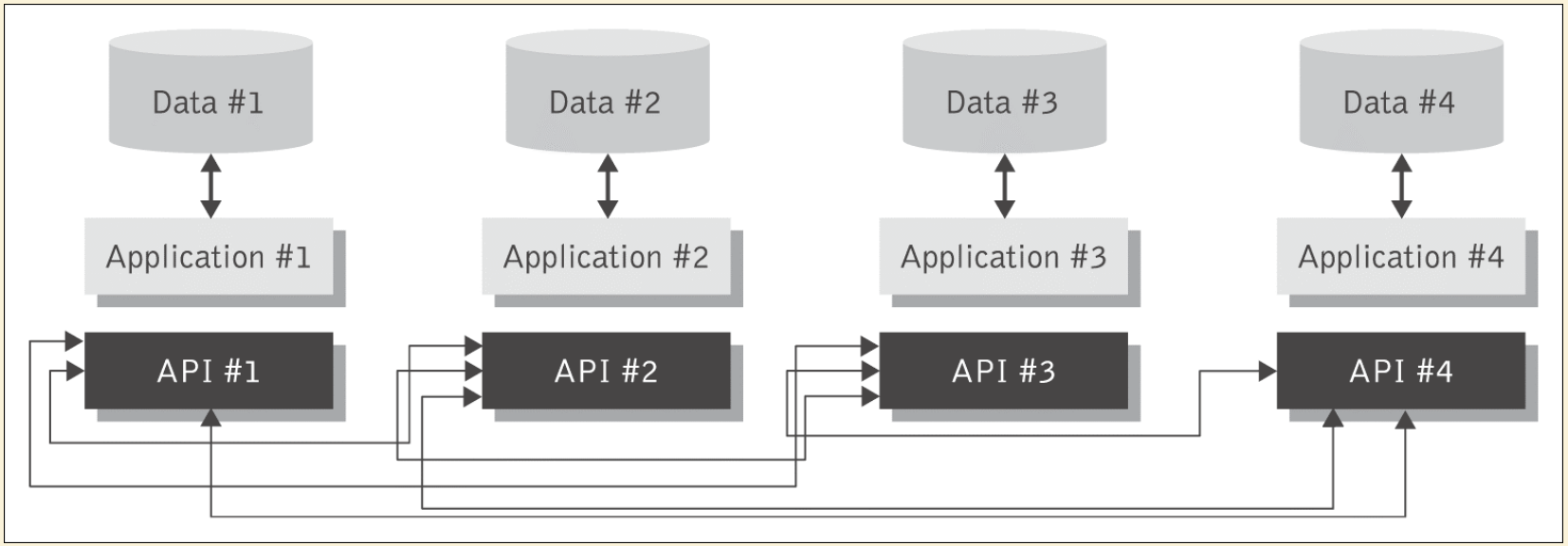
- RPC is a basic mechanism for inter-program communication, where the application elements use a request/wait-for-reply (synchronous) model of communication.



Source: From M. P. Papazoglou and P. M. A. Ribbers, *e-Business: Organizational and Technical Foundations*, J. Wiley & Sons, 2006. Reproduced with permission

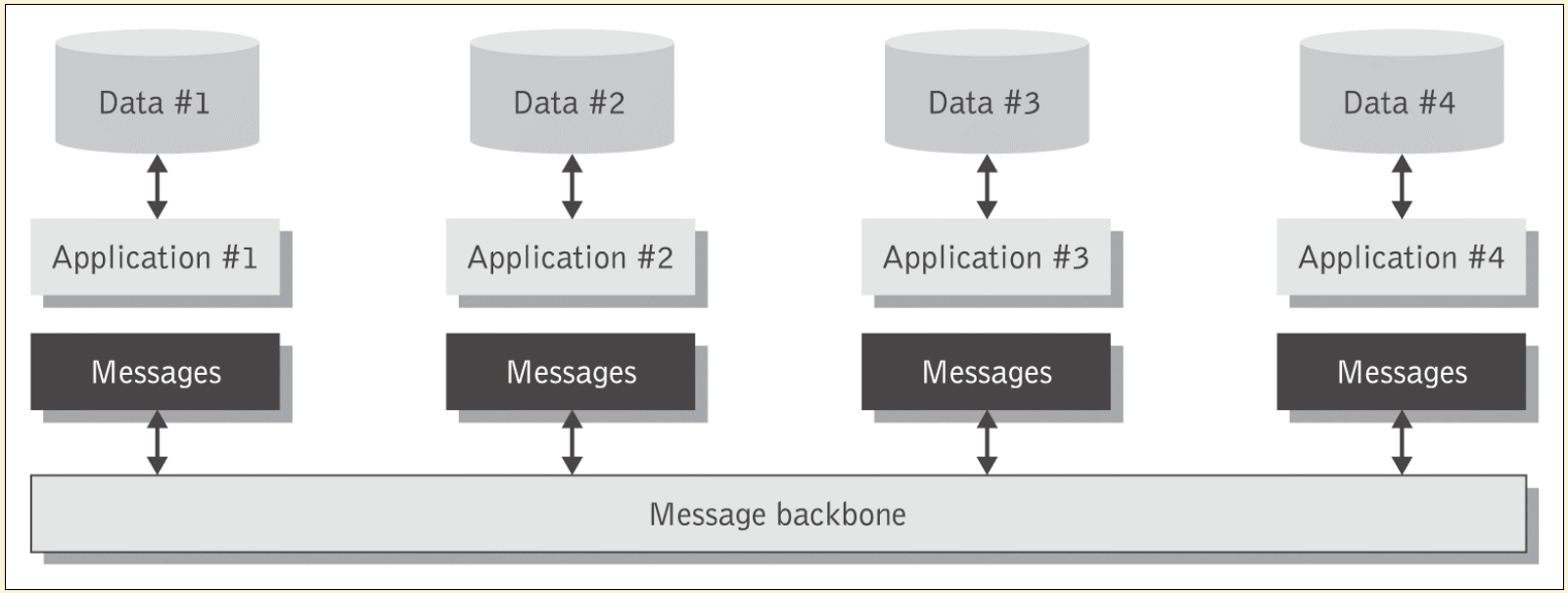
Tightly coupled RPC point-to-point integrations

- RPC-style programming leads to *tight coupling* of interfaces and applications.
- In an RPC environment each application needs to know the intimate details of the interface of every other application – the number of methods it exposes and the details of each method signature it exposes.



Asynchronous communication

- Asynchronous communication promotes *loose coupling* in which an application does not need to know the intimate details of how to reach and interface with other applications.
- Each participant in a multi-step business process flow needs only be concerned with ensuring that it can send a message to the messaging system.

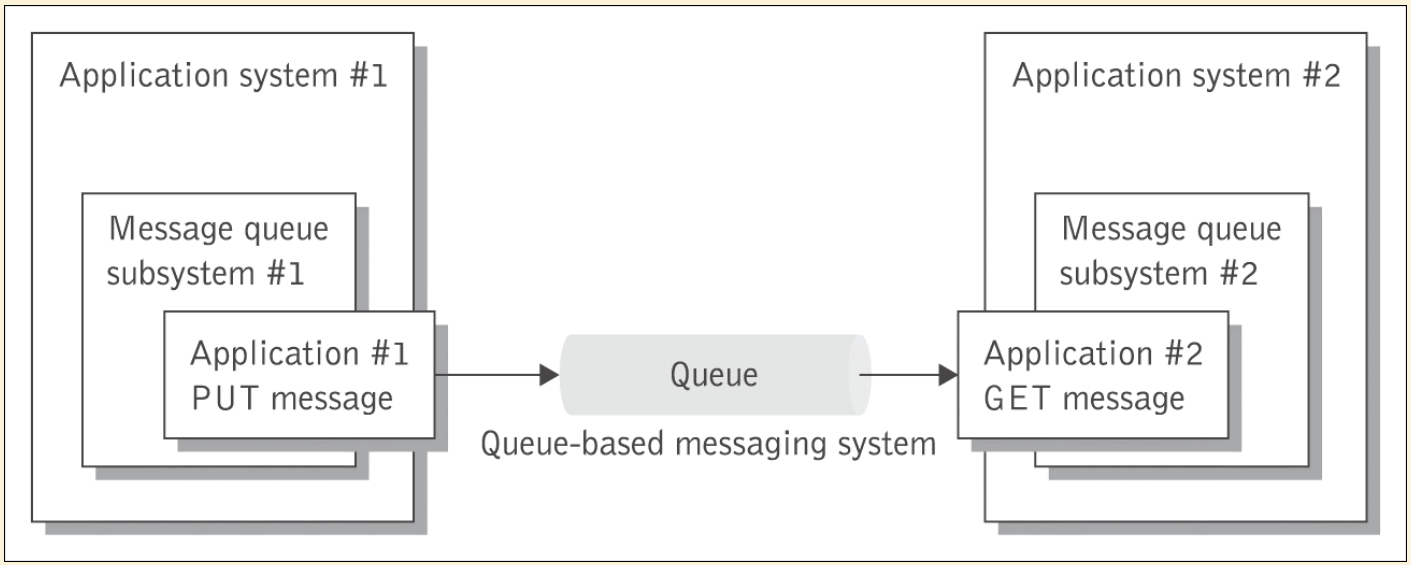


Topics

- *Distributed computing and Internet protocols*
- *The client–server model*
- *Inter-process communication*
- *Synchronous forms of middleware*
- *Asynchronous forms of middleware*
- *Request–reply messaging*
- *Message-oriented middleware*
- *Enterprise application and e-Business integration*

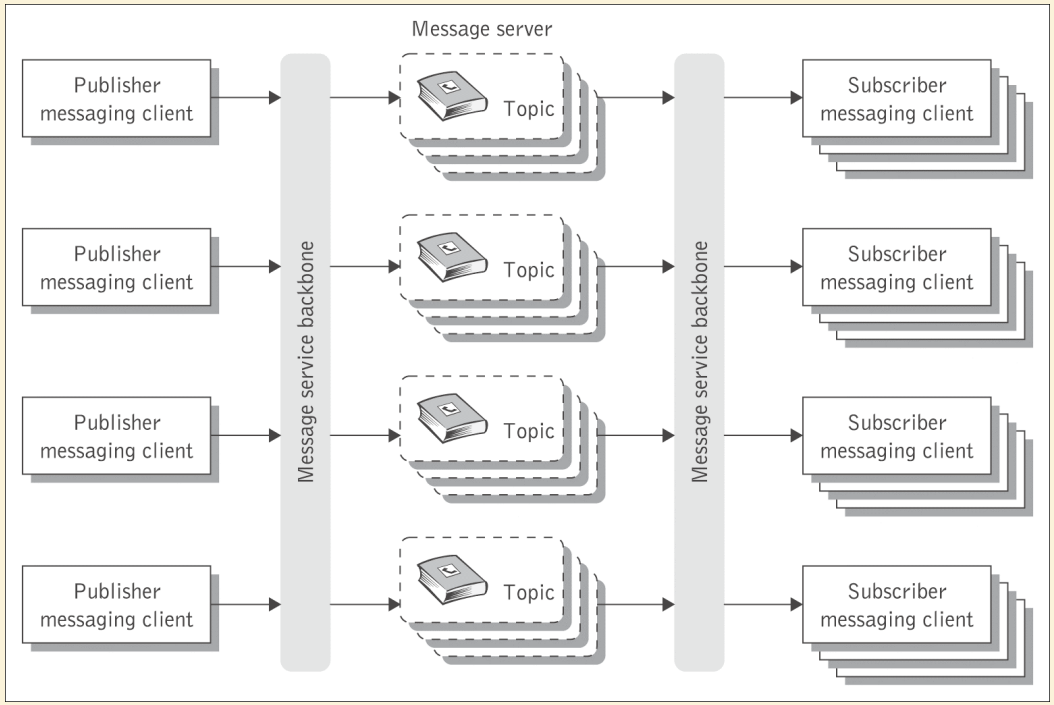
Store and forward messaging

- With the store and forward queuing mechanism, messages are placed on a virtual channel called a message queue by a sending application and are retrieved by the receiving application as needed.
  - The queue is a container that can hold the message until the recipient collects it.



Publish/Subscribe Messaging

- The application that produces information publishes it and all other applications that need this type of information subscribe to it.
  - Messages containing the new information are placed in a queue for each subscriber by the publishing application.
  - Each application may have a dual role: it may act as a publisher or subscriber of different types of information.



Event-driven processing mechanisms

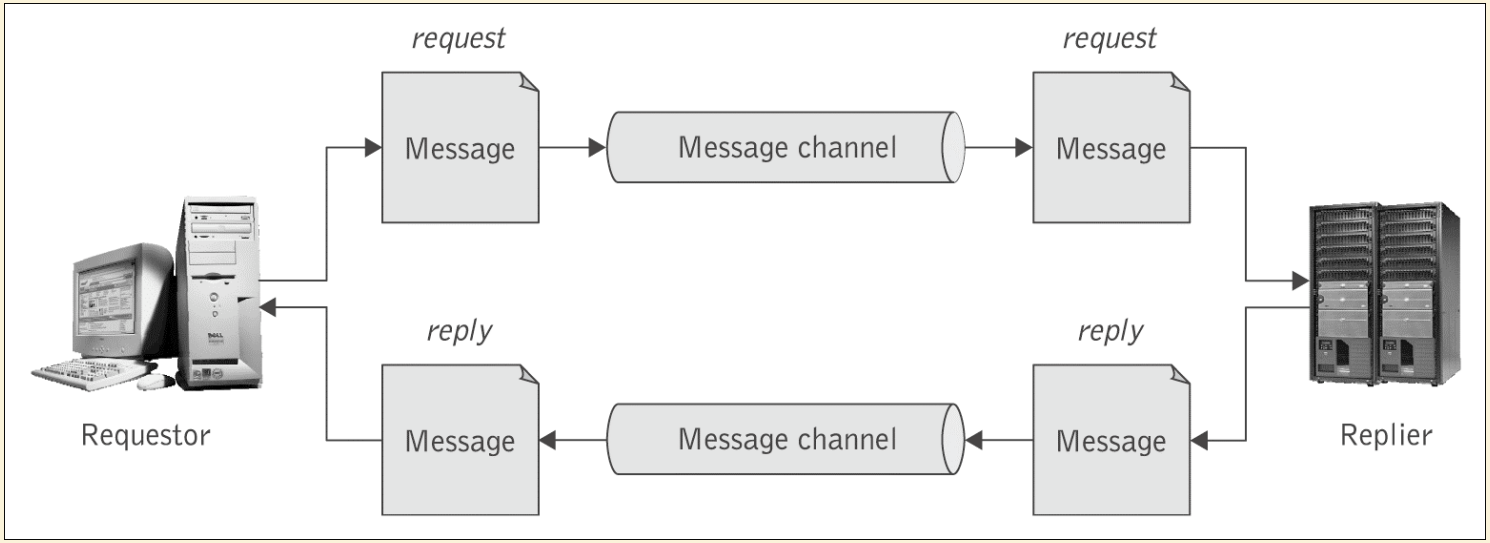
- The asynchrony, heterogeneity, and inherent loose coupling that characterize modern applications in a wide-area network requires event notification mechanisms.
- Event notification offers a many-to-many communication and integration facility. Clients in an event-notification scheme are of two kinds:
  - objects of interest, which are the producers of notifications, and
  - interested parties, which are the consumers of notifications.
- A client can act as both an object of interest and an interested party. An event notification service typically realizes the publish/subscribe asynchronous messaging scheme.

Topics

- *Distributed computing and Internet protocols*
- *The client–server model*
- *Inter-process communication*
- *Synchronous forms of middleware*
- *Asynchronous forms of middleware*
- *Request–reply messaging*
- *Message-oriented middleware*
- *Enterprise application and e-Business integration*

Asynchronous request/reply messaging

- Most asynchronous messaging mechanisms follow the “fire-and-forget” messaging principle where the sending application can conduct its work as usual once a message was asynchronously sent.
  - The sending application assumes that the message will arrive safely at its destination at some point in time.
  - This mode of messaging does not preclude the necessity to perform request/reply operations.

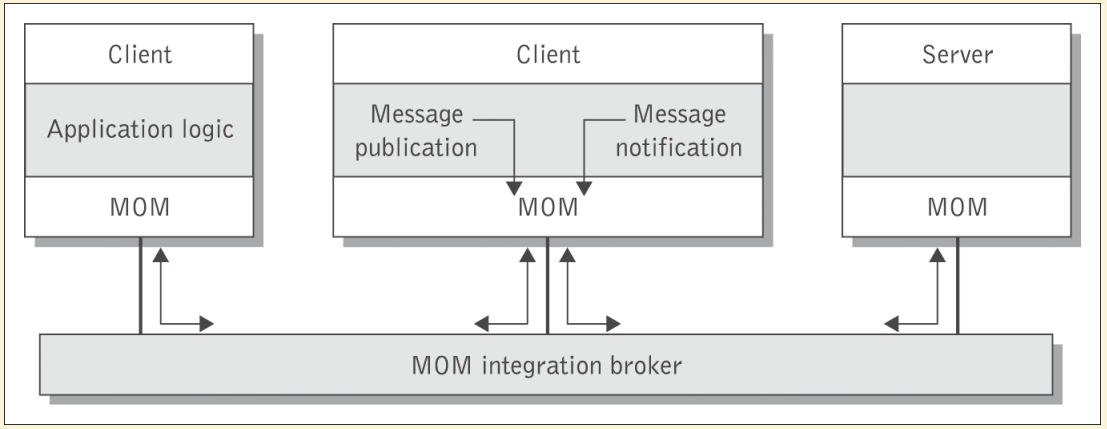


Topics

- *Distributed computing and Internet protocols*
- *The client–server model*
- *Inter-process communication*
- *Synchronous forms of middleware*
- *Asynchronous forms of middleware*
- *Request–reply messaging*
- *Message-oriented middleware*
- *Enterprise application and e-Business integration*

Message-oriented Middleware

- MOM is an infrastructure that involves the passing of data between applications using a common communication channel that carries self-contained messages.
- Messages are sent and received asynchronously.
- The messaging system (*integration broker*) is responsible for managing the connection points between clients and for managing multiple channels of communication between the connection points.



Source: From M. P. Papazoglou and P. M. A. Ribbers, e-Business: Organizational and Technical Foundations, J. Wiley & Sons Limited, 2006. Reproduced with permission

Message-oriented Middleware (continued)

- MOM provides the following functions:
  - event-driven processing, i.e., the publish/subscribe model;
  - reliability and serialization of messages;
  - subject-based (textual) names and attributes to abstract from physical names and addresses;
  - multiple communications protocols, e.g., store and forward, request/reply, publish/subscribe.
- An integration broker is an application-to-application middleware service capable of one-to-many, many-to-one and many-to-many message distribution.
  - It records and manages the contracts between publishers and subscribers of messages.
- An integration broker provides the following functions:
  - message transformation, business rules processing, routing services, naming services, adapter services, repository services, events, and alerts.

Topics

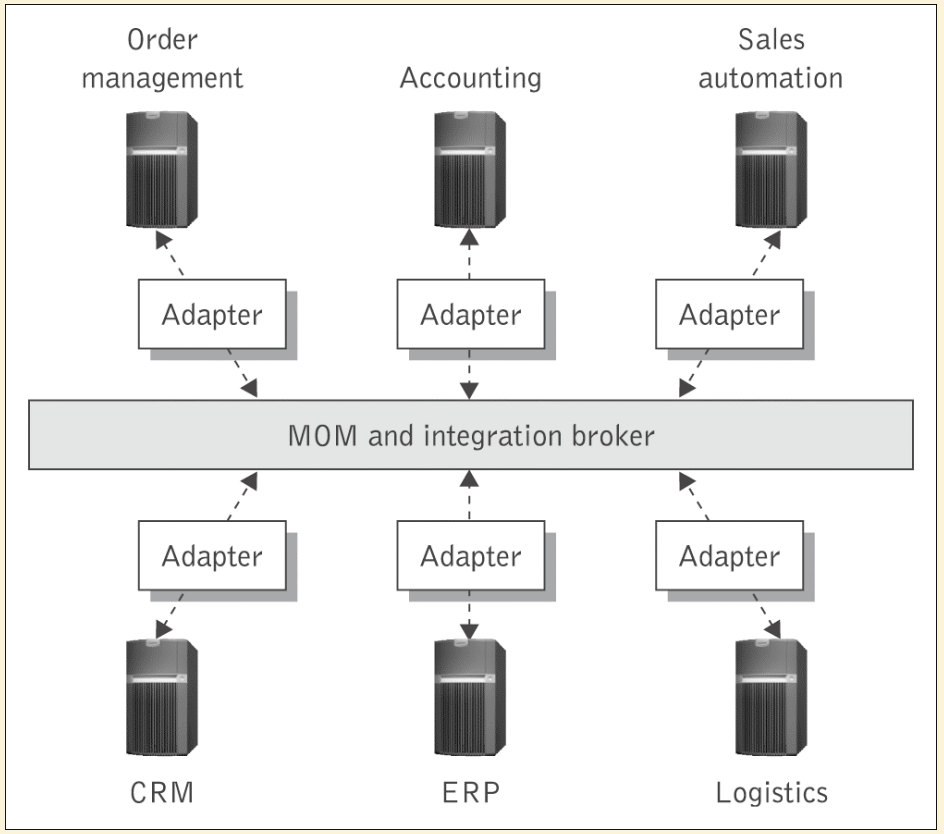
- *Distributed computing and Internet protocols*
- *The client–server model*
- *Inter-process communication*
- *Synchronous forms of middleware*
- *Asynchronous forms of middleware*
- *Request–reply messaging*
- *Message-oriented middleware*
- *Enterprise application and e-Business integration*

Enterprise Application Integration (EAI)

- EAI has emerged to help organizations eliminate islands of data and automation and integrate diverse custom and package applications (including legacy).
- The objective of EAI is to transform an organization's internal applications into a cohesive corporate framework.
- EAI enables applications throughout the enterprise to integrate seamlessly in the form of business processes.
- The internal applications in an enterprise that EAI attempts to integrate are called enterprise information systems. These include the following:
  - Custom applications
  - Legacy and database applications
  - Enterprise resource planning systems
  - Customer relationship management systems
  - Transaction systems.

EAI (continued)

- EAI uses a fast, robust communications backbone with integration broker technology, business process workflow, and facilities tools.
  - Integration brokers are used for message process flow & are responsible for brokering messages exchanged between two or more applications.
  - They provide the ability to
    - transform
    - store and route messages
    - apply business rules and
    - respond to events.



e-Business integration

- e-Business integration solutions grow on the back of successful internal EAI solutions and provide the capability to link together disparate processes between trading partners.
  - systems internal to an enterprise are able to interact with those of customers, suppliers, and partners.