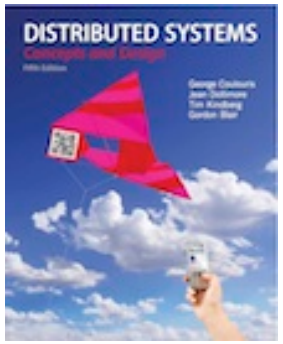
Slides for Chapter 2: Architectural Models



From Coulouris, Dollimore, Kindberg and Blair

Distributed Systems: Concepts and Design

Edition 5, © Addison-Wesley 2012

Figure 2.1
Generations of distributed systems

Distributed systems:	*Early*	*Internet-scale*	*Contemporary*
Scale	Small	Large	Ultra-large
Heterogeneity	Limited (typically relatively homogenous configurations)	Significant in terms of platforms, languages and middleware	Added dimensions introduced including radically different styles of architecture
Openness	Not a priority	Significant priority with range of standards introduced	Major research challenge with existing standards not yet able to embrace complex systems
Quality of service	In its infancy	Significant priority with range of services introduced	Major research challenge with existing services not yet able to embrace complex systems

Figure 2.2
Communicating entities and communication paradigms

Communicating entities (what is communicating)		*Communication paradigms (how they communicate)*		
System-oriented entities	*Problem-oriented entities*	*Interprocess communication*	*Remote invocation*	*Indirect communication*
Nodes	Objects	Message passing	Request-reply	Group communication
Processes	Components	Sockets	RPC	Publish-subscribe
	Web services	Multicast	RMI	Message queues
				Tuple spaces
				DSM

Figure 2.3
Clients invoke individual servers

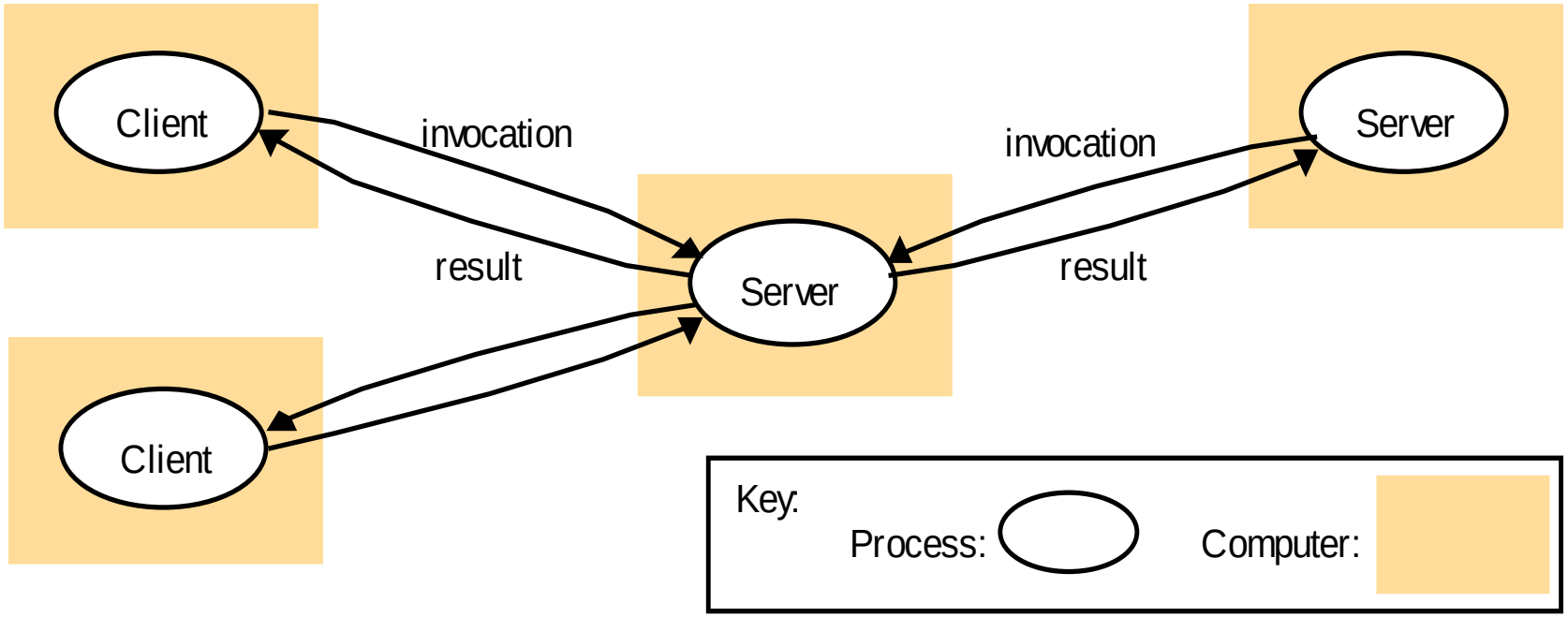


Figure 2.4a
Peer-to-peer architecture

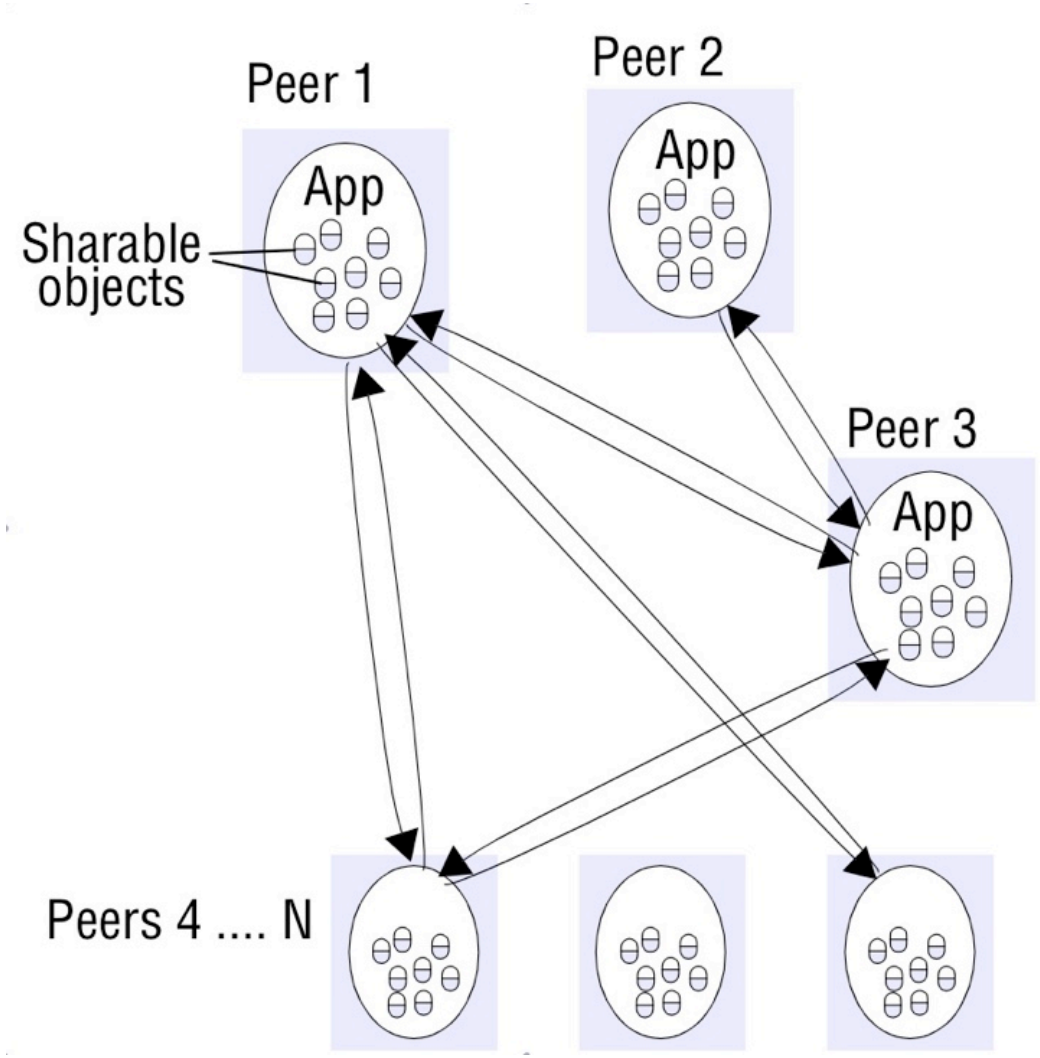


Figure 2.4b
A service provided by multiple servers

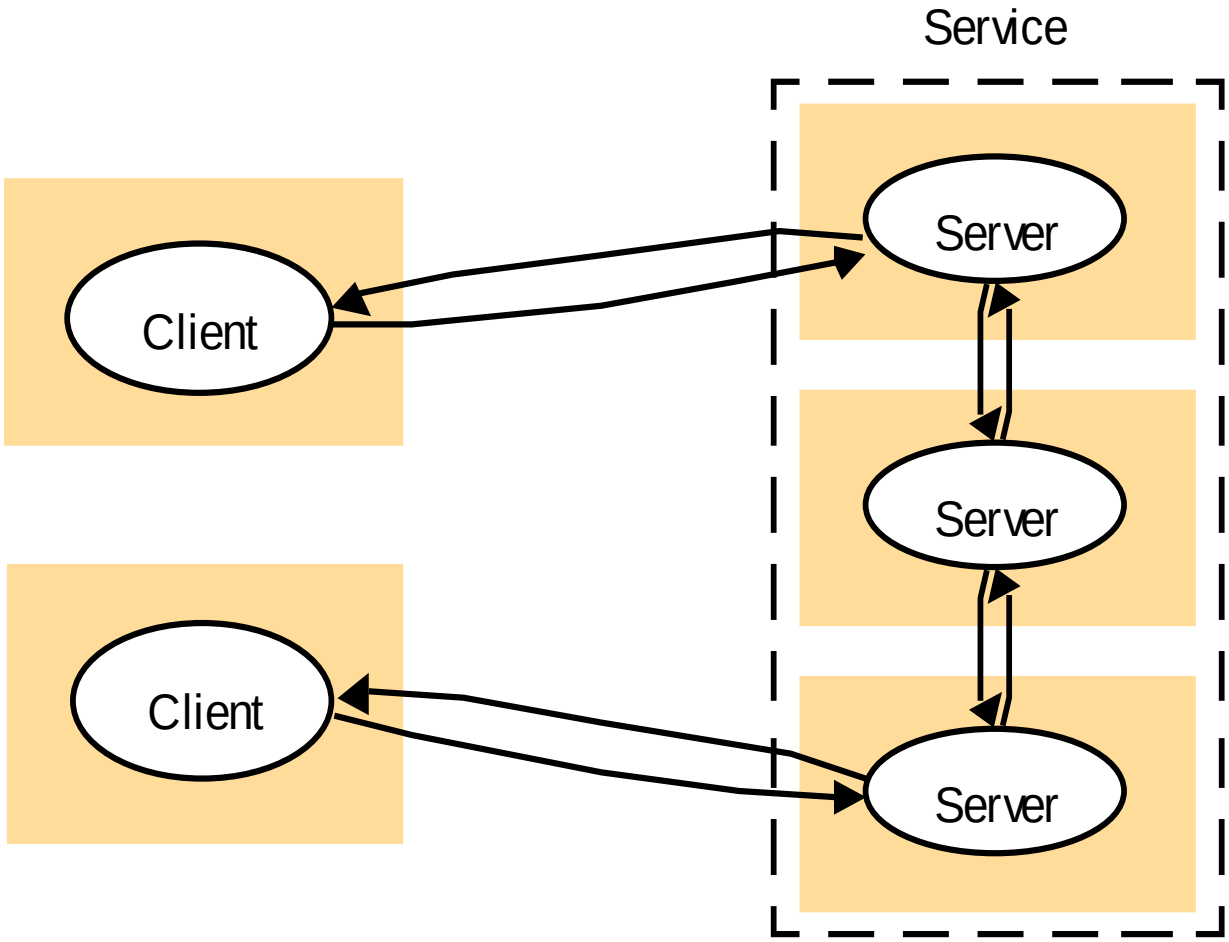


Figure 2.5
Web proxy server

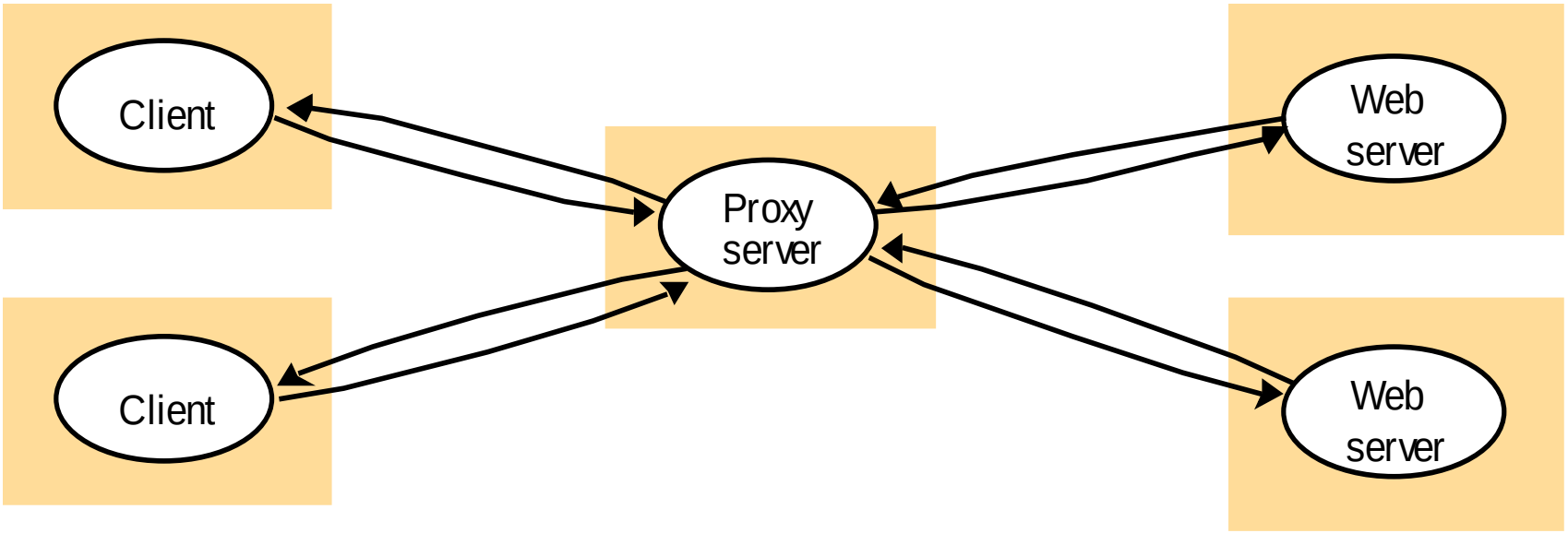
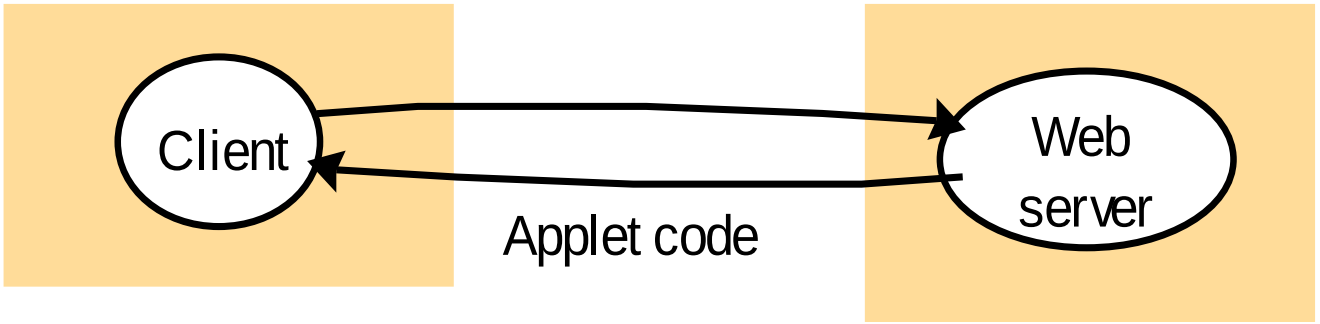


Figure 2.6
Web applets

a) client request results in the downloading of applet code



b) client interacts with the applet

Figure 2.7
Software and hardware service layers in distributed systems

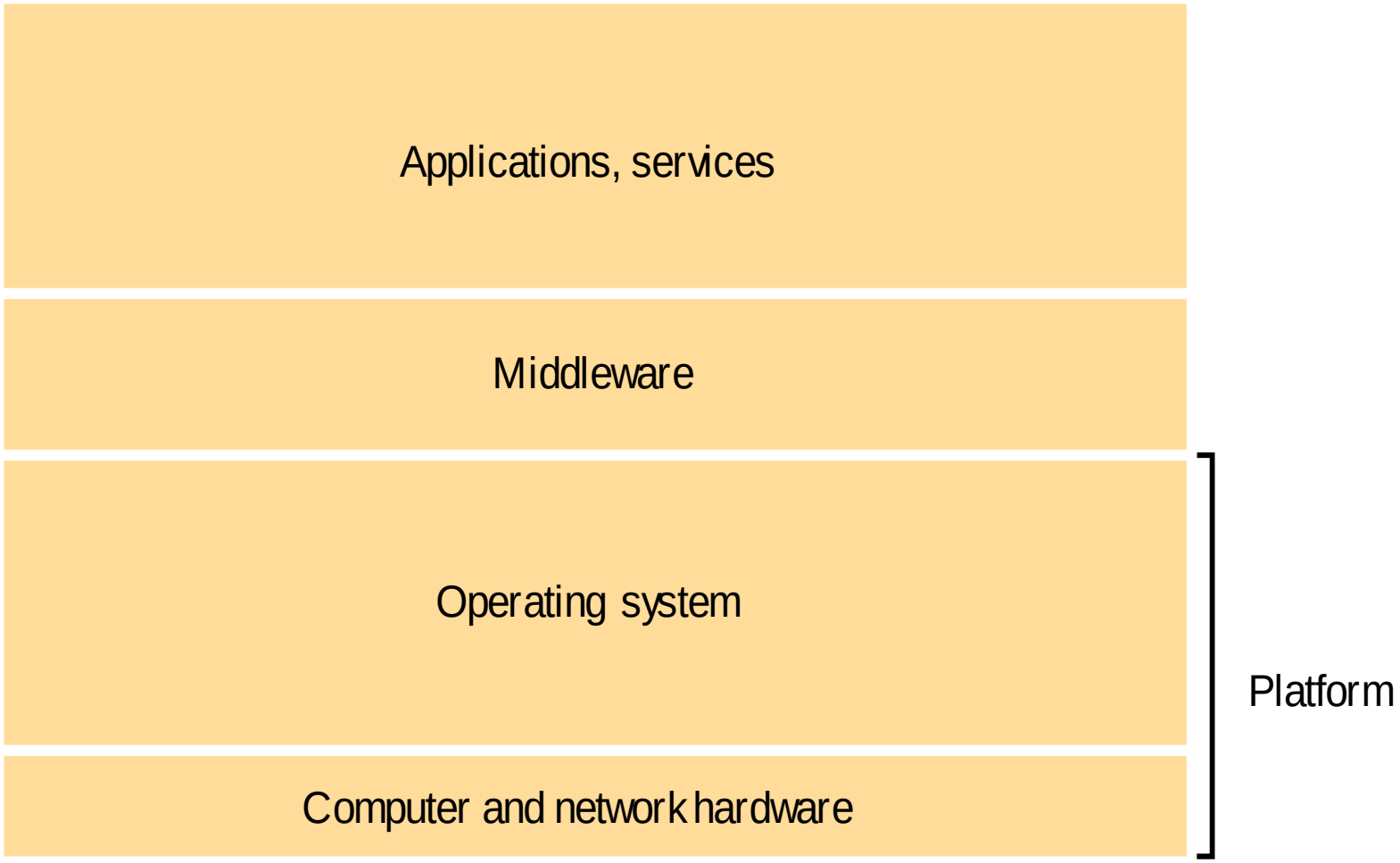


Figure 2.8
Two-tier and three-tier architectures

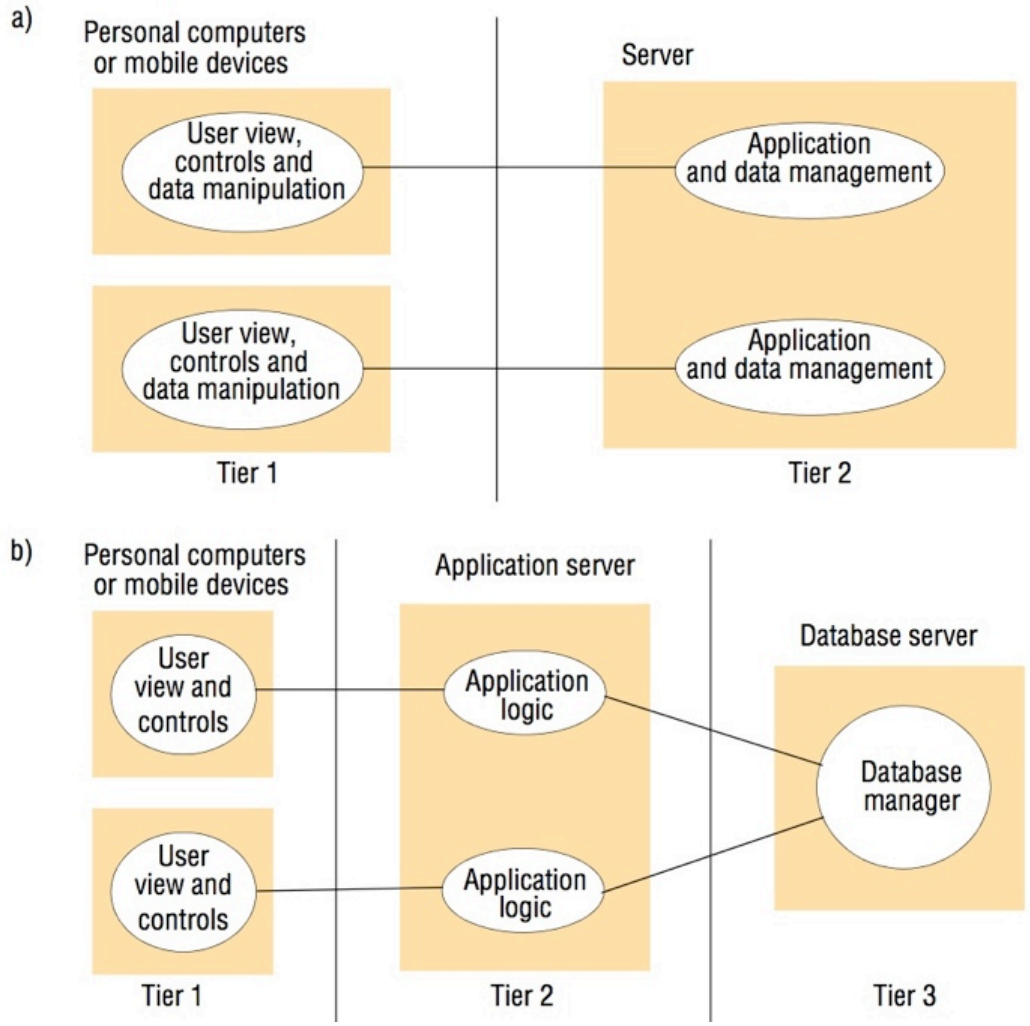


Figure 2.9
AJAX example: soccer score updates

```
new Ajax.Request('scores.php?
                    game=Arsenal:Liverpool',
                   {onSuccess: updateScore});
function updateScore(request) {
.....
( request contains the state of the Ajax request
including the returned result.
The result is parsed to obtain some text giving the
score, which is used to update the relevant portion
of the current page.)
.....
}
```

Figure 2.10
Thin clients and compute servers

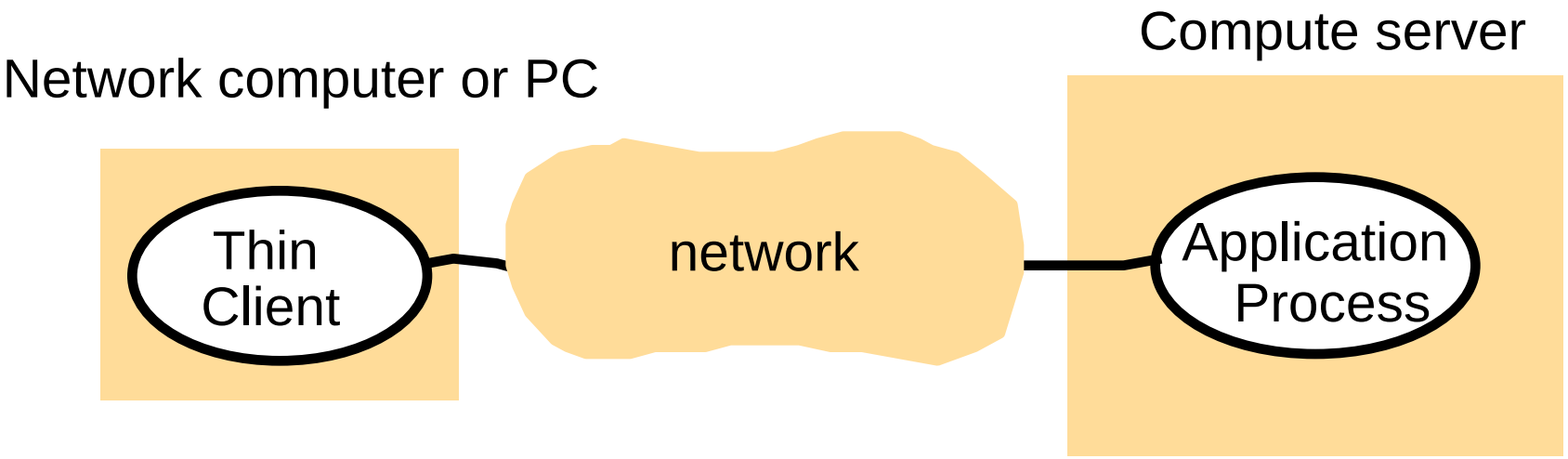


Figure 2.11
The web service architectural pattern

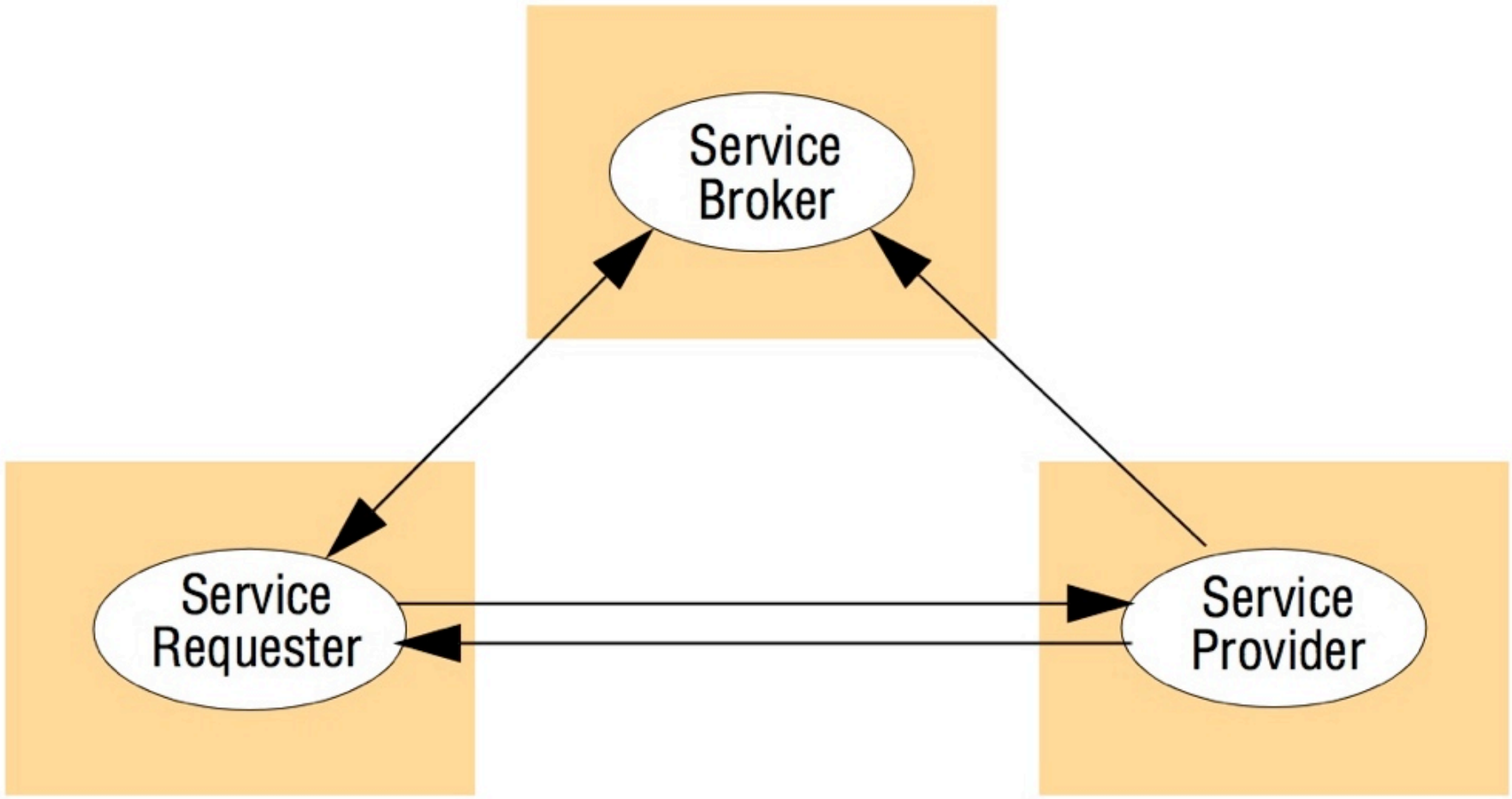


Figure 2.12
Categories of middleware

Major categories:	*Subcategory*	*Example systems*
Distributed objects (Chapters 5, 8)	Standard	RM-ODP
	Platform	CORBA
	Platform	Java RMI
Distributed components (Chapter 8)	Lightweight components	Fractal
	Lightweight components	OpenCOM
	Application servers	SUN EJB
	Application servers	CORBA Component Model
	Application servers	JBoss
Publish-subscribe systems (Chapter 6)	-	CORBA Event Service
	-	Scribe
	-	JMS
Message queues (Chapter 6)	-	Websphere MQ
	-	JMS
Web services (Chapter 9)	Web services	Apache Axis
	Grid services	The Globus Toolkit
Peer-to-peer (Chapter 10)	Routing overlays	Pastry
	Routing overlays	Tapestry
	Application-specific	Squirrel
	Application-specific	OceanStore
	Application-specific	Ivy
	Application-specific	Gnutella

Figure 2.13
Real-time ordering of events

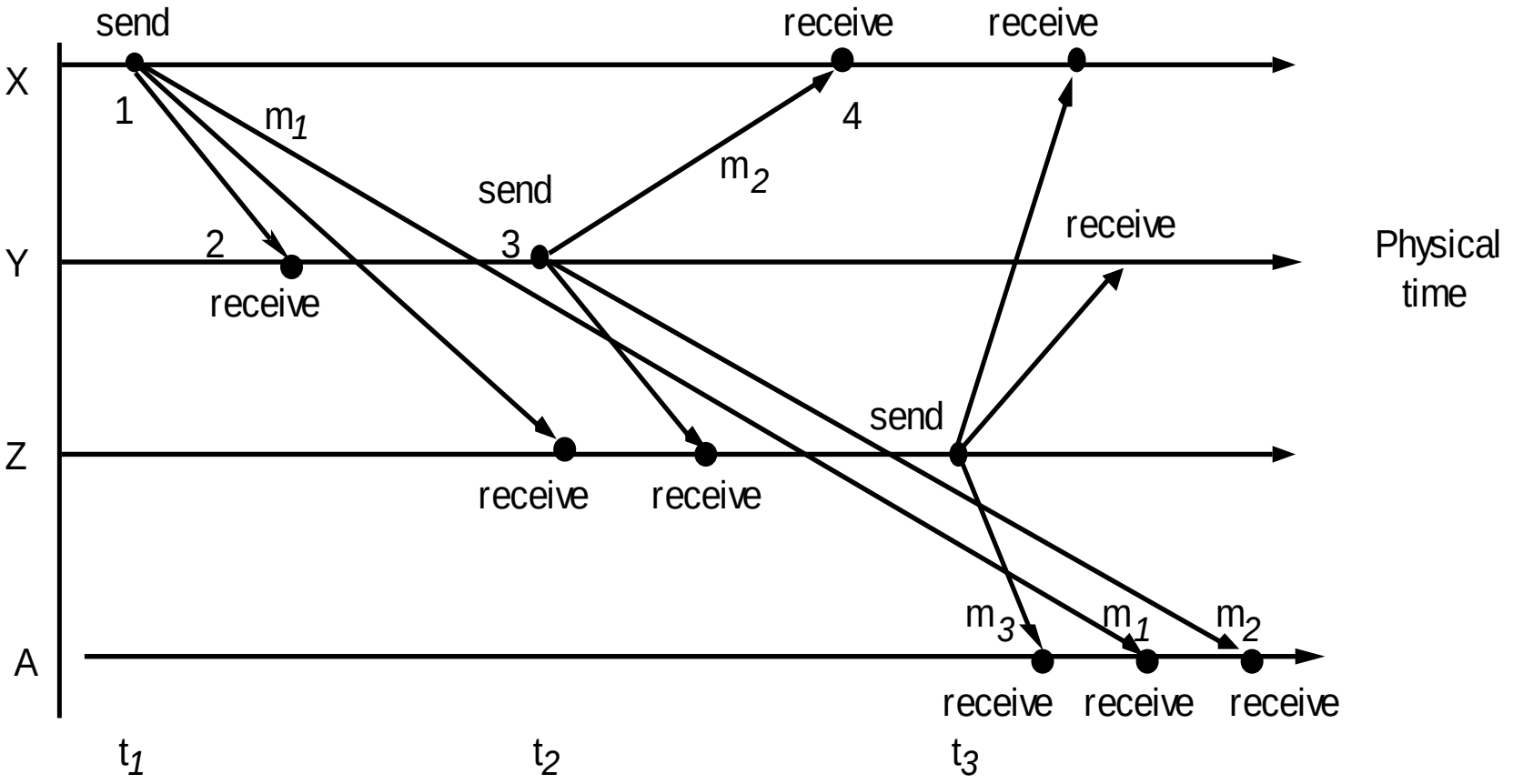


Figure 2.14
Processes and channels

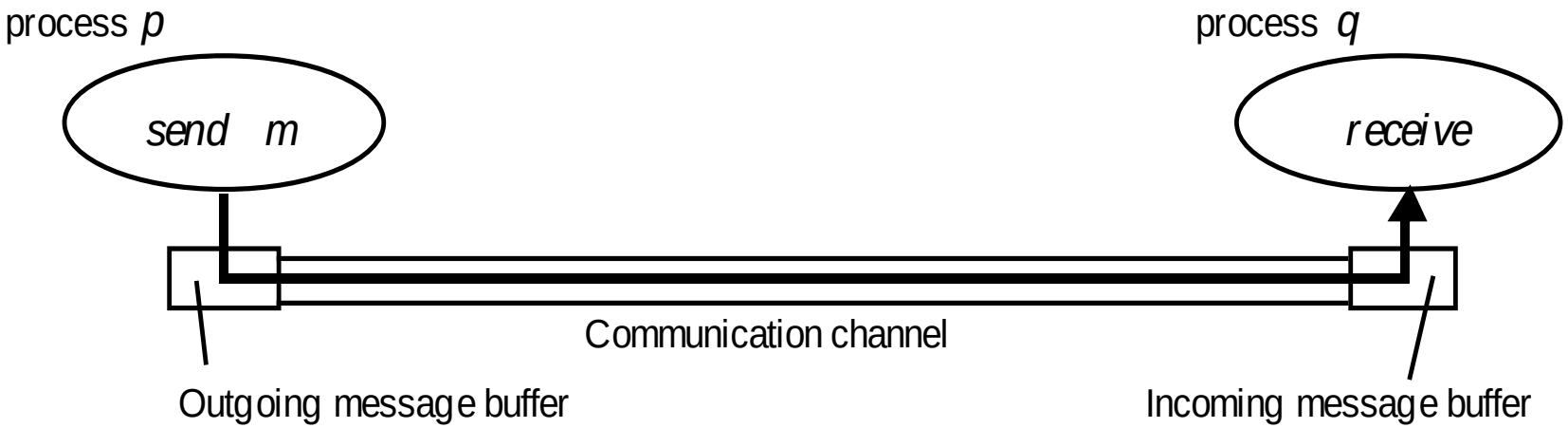


Figure 2.15
Omission and arbitrary failures

Class of failure	*Affects*	*Description*
Fail-stop	Process	Process halts and remains halted. Other processes may detect this state.
Crash	Process	Process halts and remains halted. Other processes may not be able to detect this state.
Omission	Channel	A message inserted in an outgoing message buffer never arrives at the other end's incoming message buffer.
Send-omission	Process	A process completes a *send*, but the message is not put in its outgoing message buffer.
Receive-omission	Process	A message is put in a process's incoming message buffer, but that process does not receive it.
Arbitrary (Byzantine)	Process or channel	Process/channel exhibits arbitrary behaviour: it may send/transmit arbitrary messages at arbitrary times, commit omissions; a process may stop or take an incorrect step.

Figure 2.11
Timing failures

Class of Failure	*Affects*	*Description*
Clock	Process	Process's local clock exceeds the bounds on its rate of drift from real time.
Performance	Process	Process exceeds the bounds on the interval between two steps.
Performance	Channel	A message's transmission takes longer than the stated bound.

Figure 2.17
Objects and principals

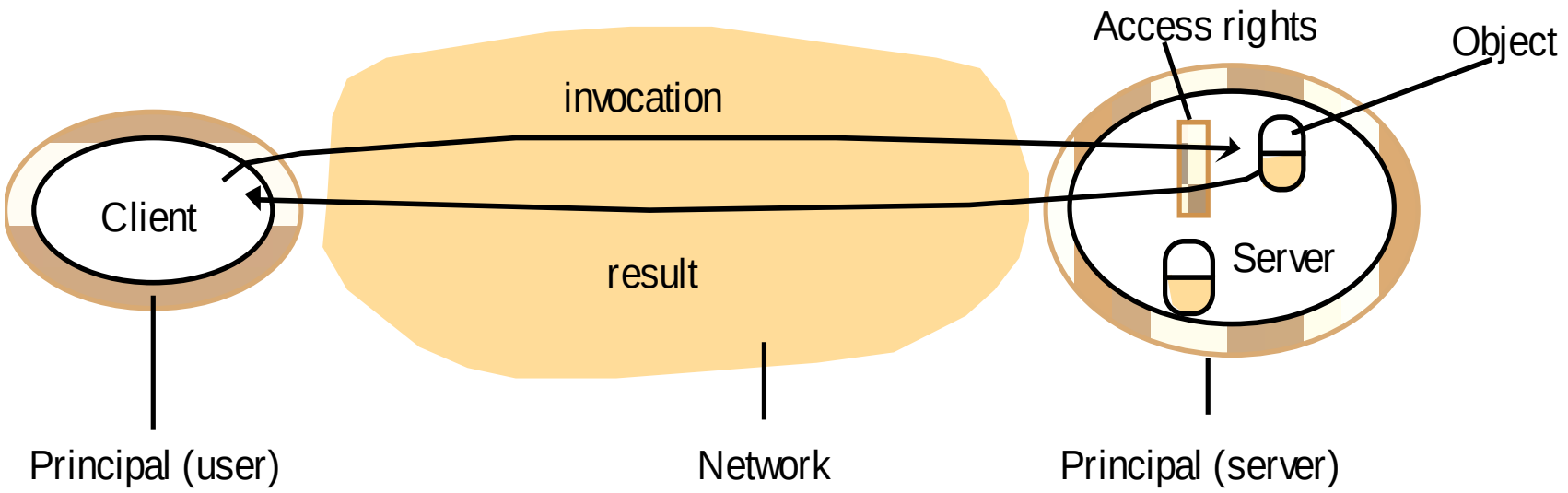


Figure 2.18
The enemy

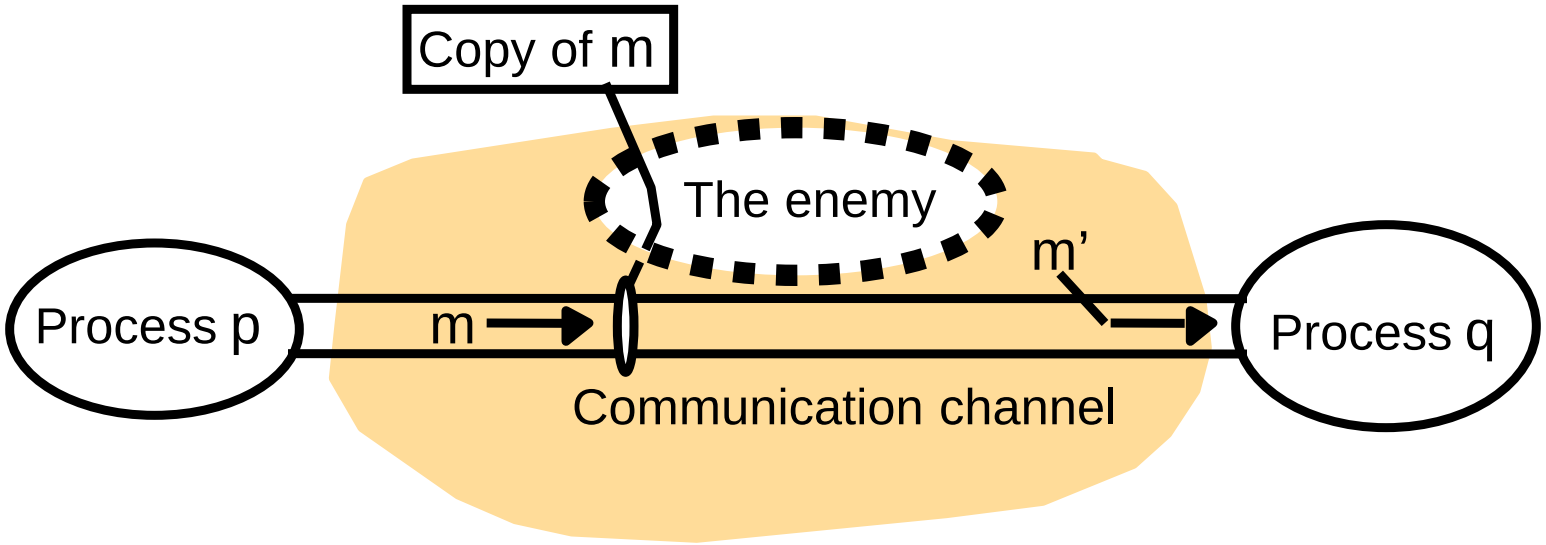


Figure 2.19
Secure channels