Figure 2.1  Software and hardware service layers in distributed systems

- Applications, services
- Middleware
- Operating system
- Computer and network hardware
Figure 2.2  Clients invoke individual servers

Key:

Process:  Computer:
Figure 2.3  A service provided by multiple servers
Figure 2.4 Web proxy server

Client → Proxy server → Web server

Client → Proxy server → Web server

Client → Proxy server → Web server

Client → Proxy server → Web server
Figure 2.5  A distributed application based on peer processes

Application
Coordination code

Application
Coordination code

Application
Coordination code
Figure 2.6  Web applets

a) client request results in the downloading of applet code

b) client interacts with the applet
Figure 2.7  Thin clients and compute servers

Network computer or PC

Thin Client --- network --- Application Process

Compute server
Figure 2.8  Spontaneous networking in a hotel
Figure 2.9 Real-time ordering of event.

Physical time

X

Y

Z

A

send

receive

send

receive

receive

send

receive

receive

receive

send

receive

send

receive

receive

receive

1

2

3

4

m1

m2

m3

m1

m2

m3

t1

t2

t3
Figure 2.10  Processes and channels

process p

send  m

Outgoing message buffer

Communication channel

Incoming message buffer

process q

receive
### Figure 2.11 Omission and arbitrary failures

<table>
<thead>
<tr>
<th>Class of failure</th>
<th>Affects</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fail-stop</td>
<td>Process</td>
<td>Process halts and remains halted. Other processes may detect this state.</td>
</tr>
<tr>
<td>Crash</td>
<td>Process</td>
<td>Process halts and remains halted. Other processes may not be able to detect this state.</td>
</tr>
<tr>
<td>Omission</td>
<td>Channel</td>
<td>A message inserted in an outgoing message buffer never arrives at the other end’s incoming message buffer.</td>
</tr>
<tr>
<td>Send-omission</td>
<td>Process</td>
<td>A process completes a <em>send</em>, but the message is not put in its outgoing message buffer.</td>
</tr>
<tr>
<td>Receive-omission</td>
<td>Process</td>
<td>A message is put in a process’s incoming message buffer, but that process does not receive it.</td>
</tr>
<tr>
<td>Arbitrary (Byzantine)</td>
<td>Process or channel</td>
<td>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.</td>
</tr>
</tbody>
</table>
### Figure 2.12  Timing failures

<table>
<thead>
<tr>
<th>Class of Failure</th>
<th>Affects</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock</td>
<td>Process</td>
<td>Process’s local clock exceeds the bounds on its rate of drift from real time.</td>
</tr>
<tr>
<td>Performance</td>
<td>Process</td>
<td>Process exceeds the bounds on the interval between two steps.</td>
</tr>
<tr>
<td>Performance</td>
<td>Channel</td>
<td>A message’s transmission takes longer than the stated bound.</td>
</tr>
</tbody>
</table>
Figure 2.13  Objects and principals

Client

Principal (user)

invocation

result

Network

Server

Principal (server)

Access rights

Object

Figure 2.14  The enemy

Process \( p \)  \( m \rightarrow \) Communication channel  \( m' \rightarrow \) Process \( q \)

Copy of \( m \)

The enemy
Figure 2.15  Secure channels

Principal A

Process p

Secure channel

Process q

Principal B