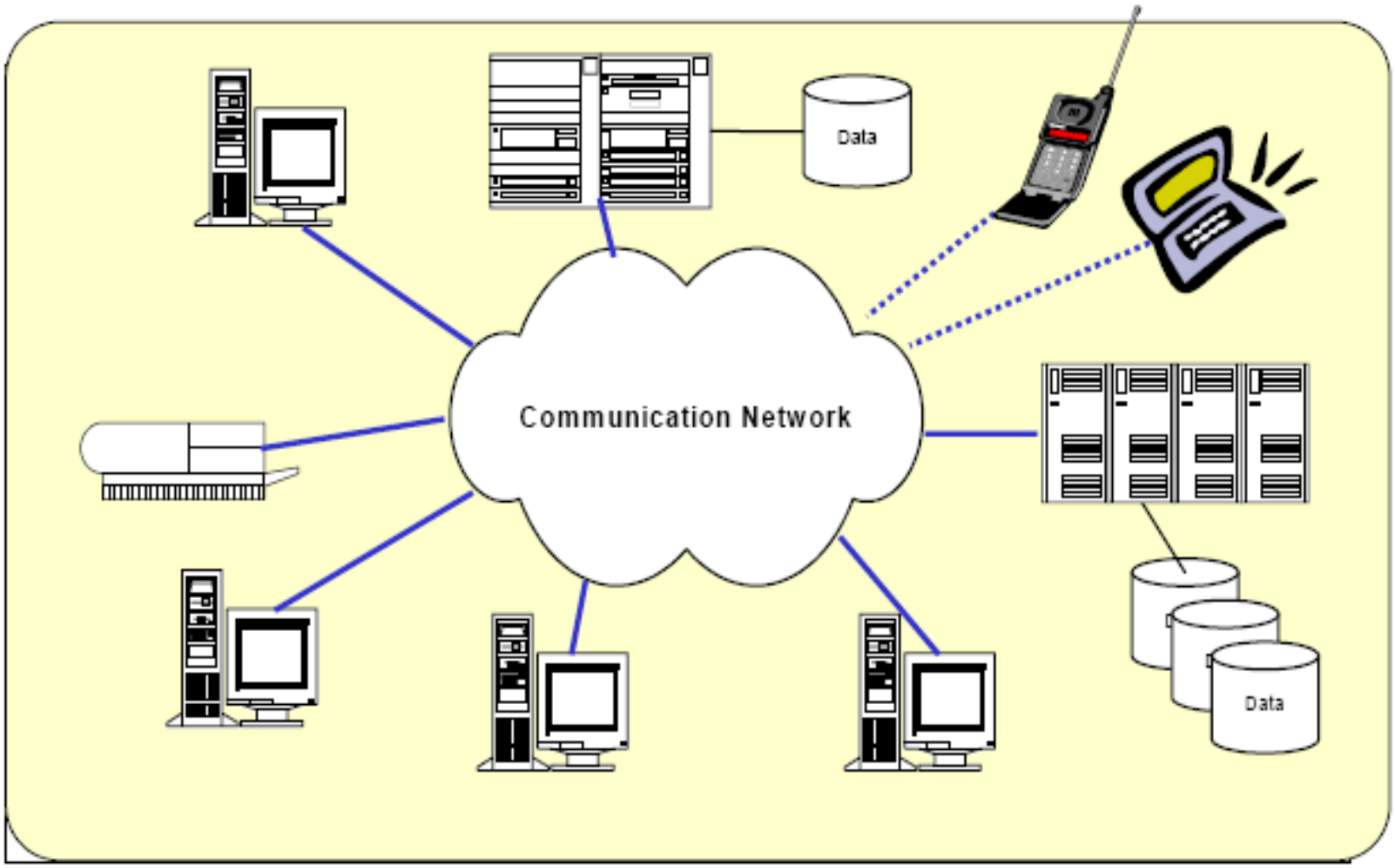
Distributed Systems

Examples
Characteristics

Werner Nutt

A Distributed System …



Examples and Counterexamples

- Internet ?
- Uni Bozen intranet ?
- Mobile phone networks ?
- Bank account management,
  including automatic teller machines ?
- Chat room ?
- Collaborative editing tool ?
- Threads in an application ?
- Applications running on a PC ?

Definition

A *distributed system* consists of

- autonomous components (hardware, software)
- that are located at networked computers (hosts)
- and that communicate and coordinate their actions only be passing messages.

Motivation

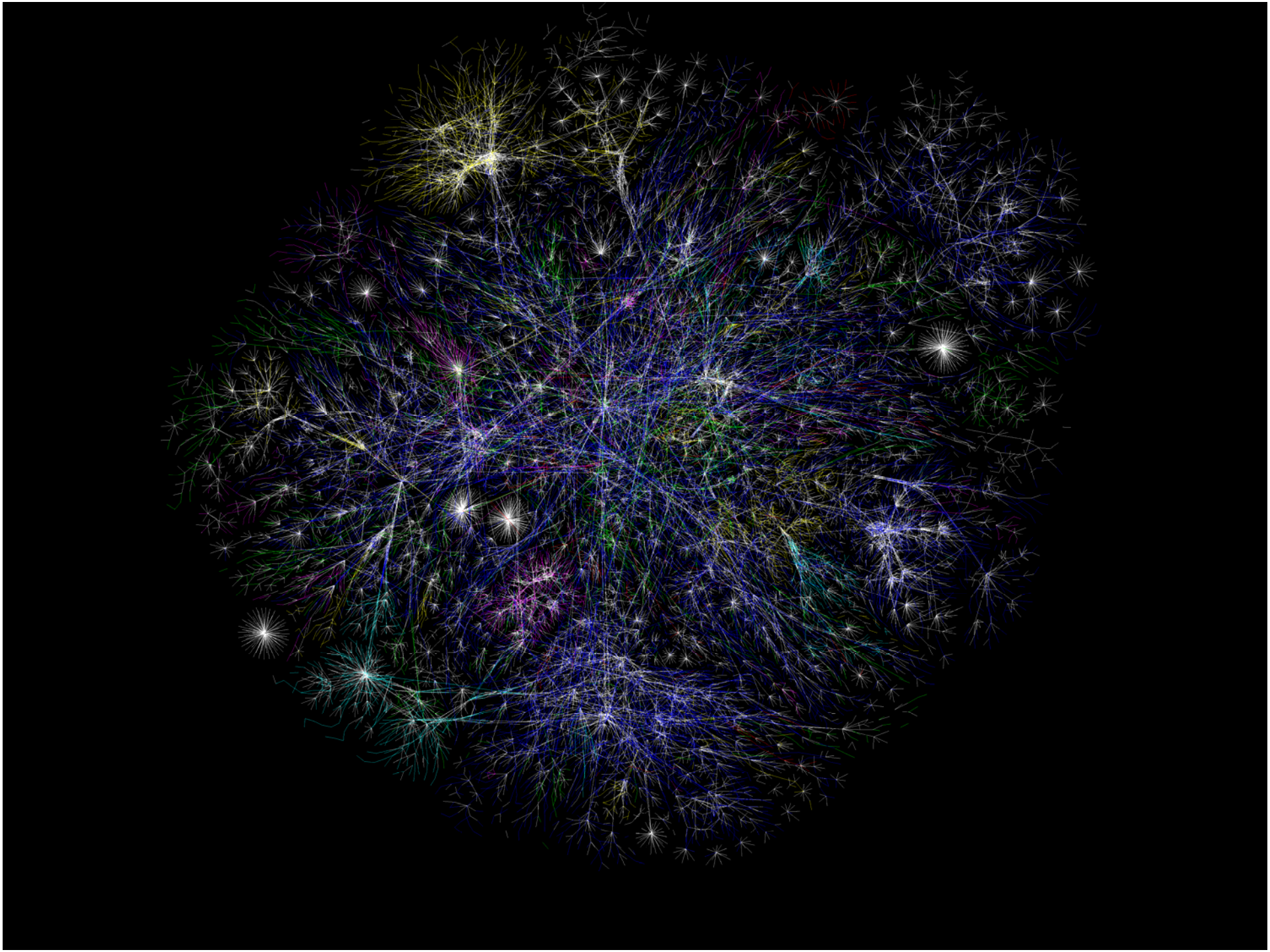
- Sharing of resources
  (printers, disks, cameras, data, applications)
- …
- …

Implications

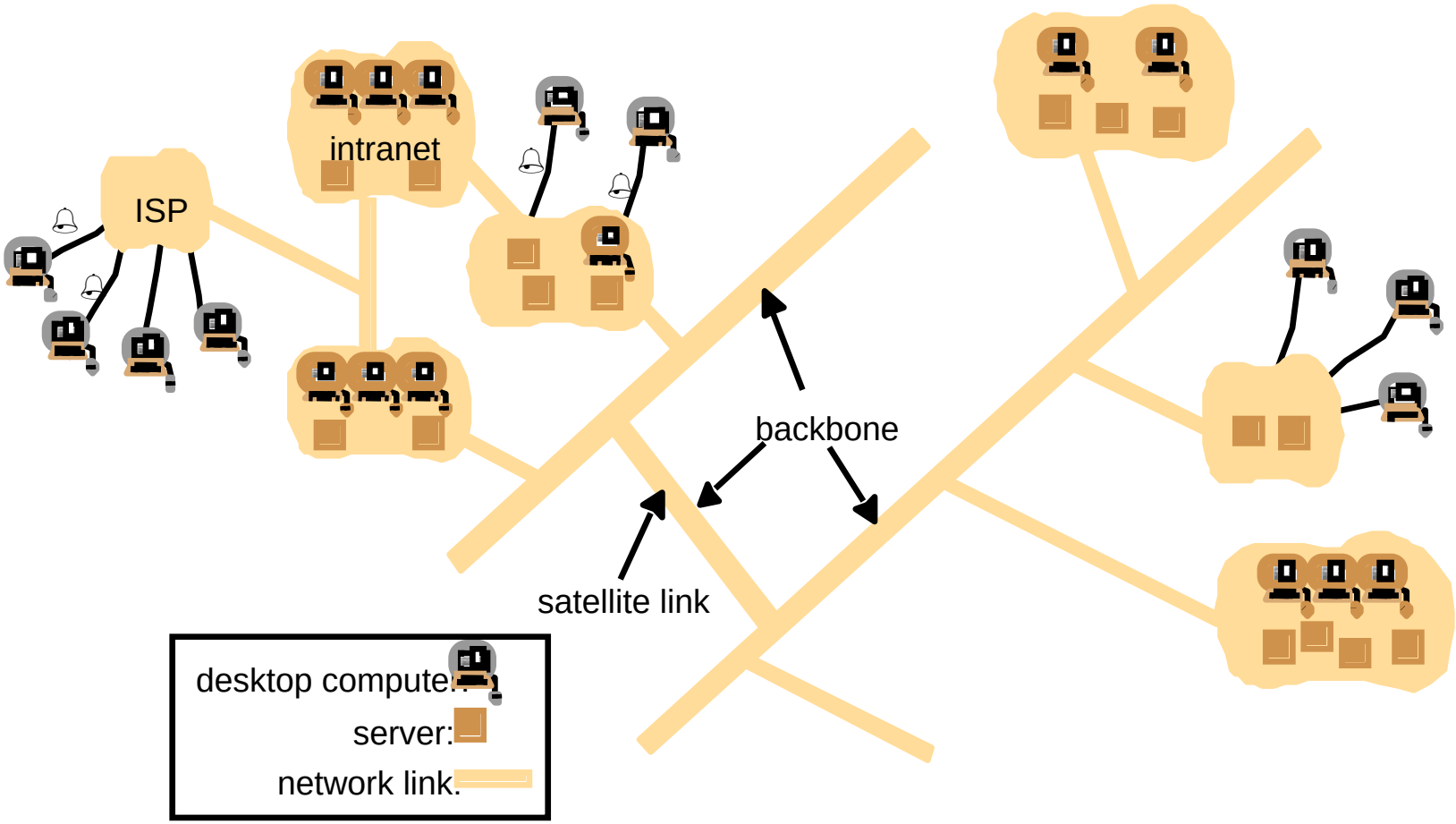
- Programs are executed concurrently
- There is no global time
- Components can fail independently (isolation, crash)

Examples

1. The Internet
2. Intranets
3. Mobile and Ubiquitous Computing
4. The Word Wide Web



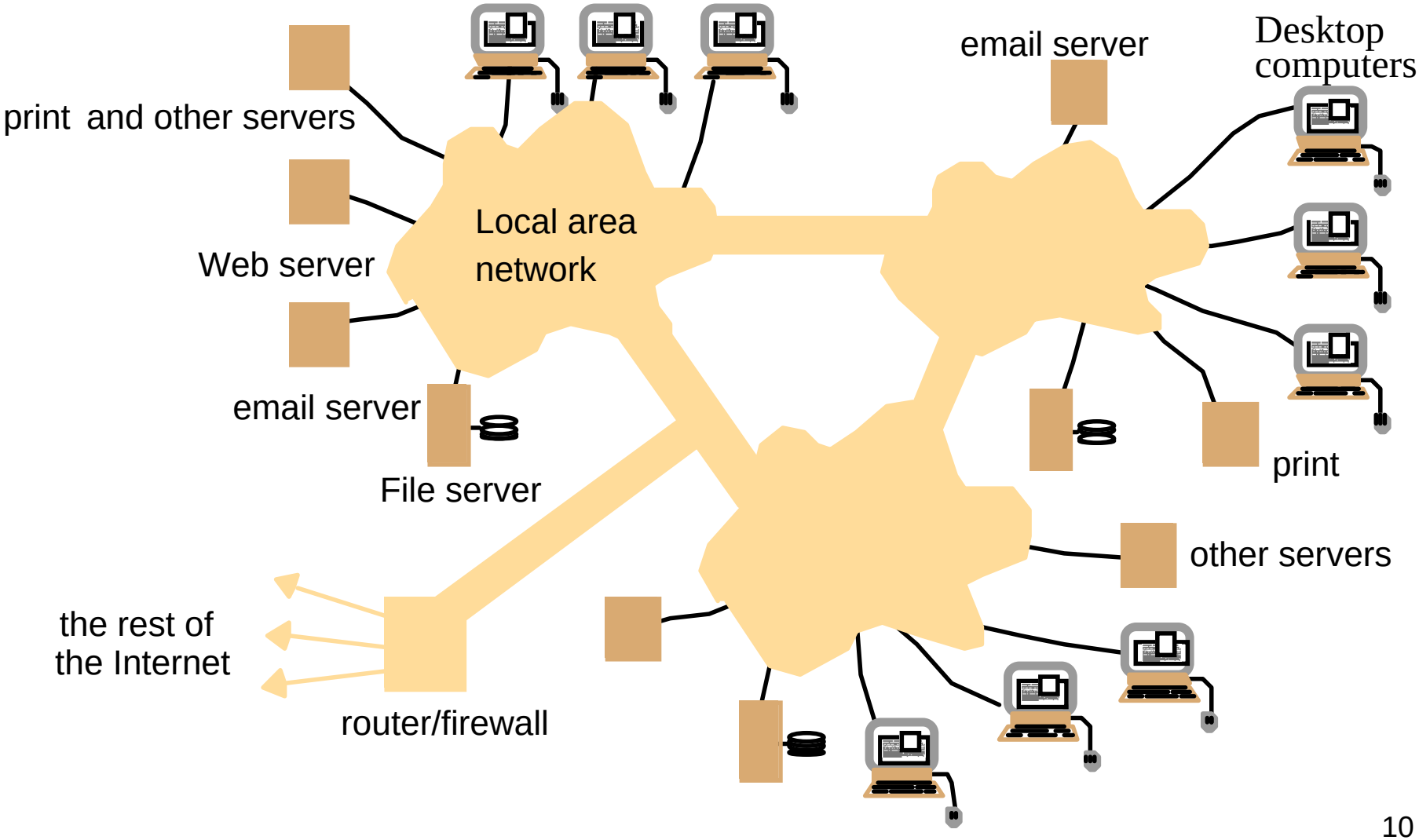
A Portion of the Internet



The Internet

- Collection of computer networks
- Enables programs to communicate over arbitrary distance
- Makes available services
  - mail, file transfer, documents, telephony, ...
- Communication via message passing according to Internet protocols
  - (IP, UDP, TCP, ICMP, SMTP, FTP, ...)
- Infrastructure: backbones, routing, naming
- Extensible (new services, new protocols)
- Open communication channels (security!)
- Technology applicable to other distributed systems

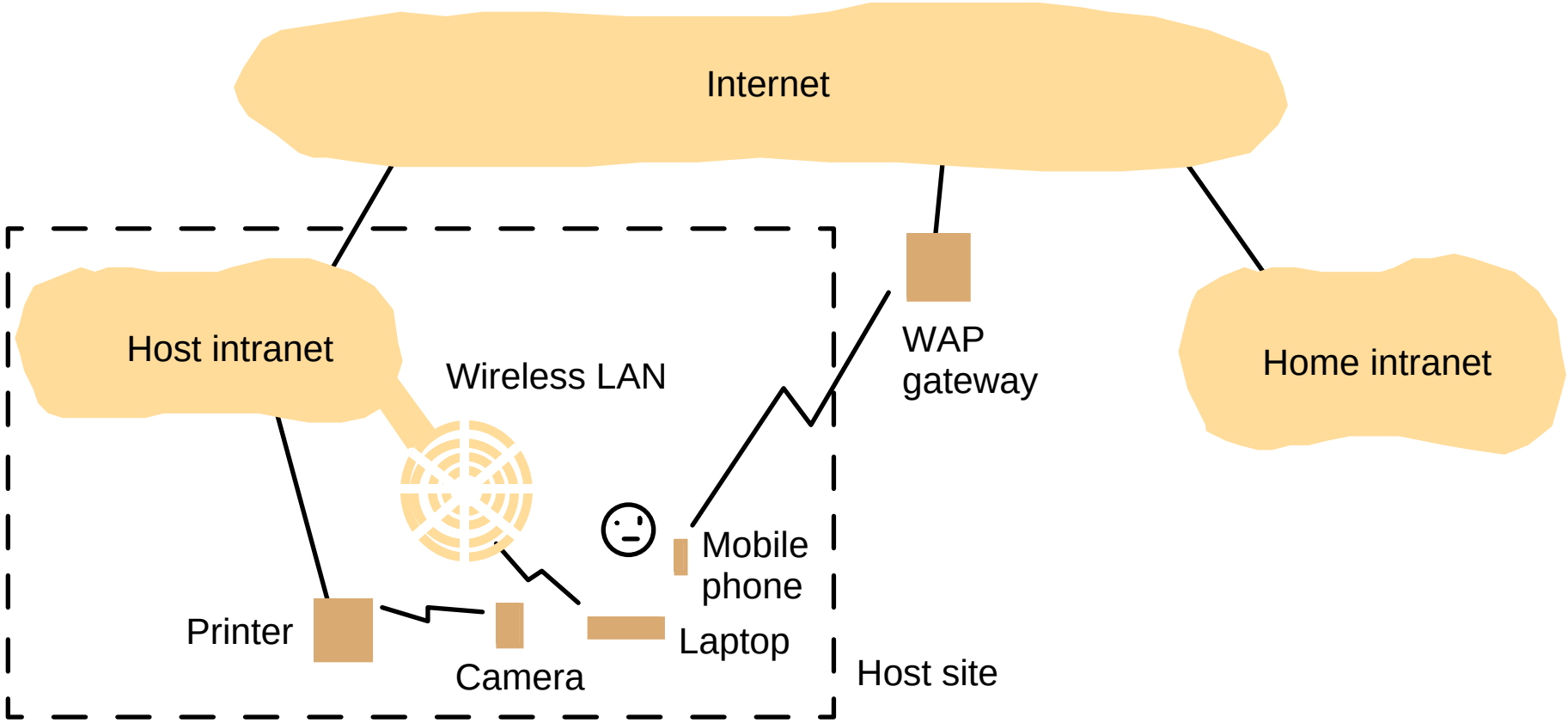
An Intranet


print and other servers
Web server
email server
File server
Local area network
email server
Desktop computers
print
other servers
the rest of the Internet
router/firewall

Intranets

- Autonomous networks running Internet protocols
  - independent administration
  - boundary, where security policies are enforced
  - access via router/firewall
- Consists of one or more LANs
- Firewall
  - filters incoming and outgoing messages
  - ... sometimes too many
- File services
- Other servers

Mobile and Handheld Devices



Note: WAP is obsolete these days!

Mobile and Ubiquitous Computing

- Mobile: computing devices are being carried around
- Ubiquitous: little computing devices are all over the place

➔ Having computers everywhere makes sense only when they can communicate

- Issues:
  - how organise physical communication
  - discovery of resources
  - eliminate need to reconfigure the device in a new environment
  - cope with limited connectivity
  - privacy and security

The World Wide Web

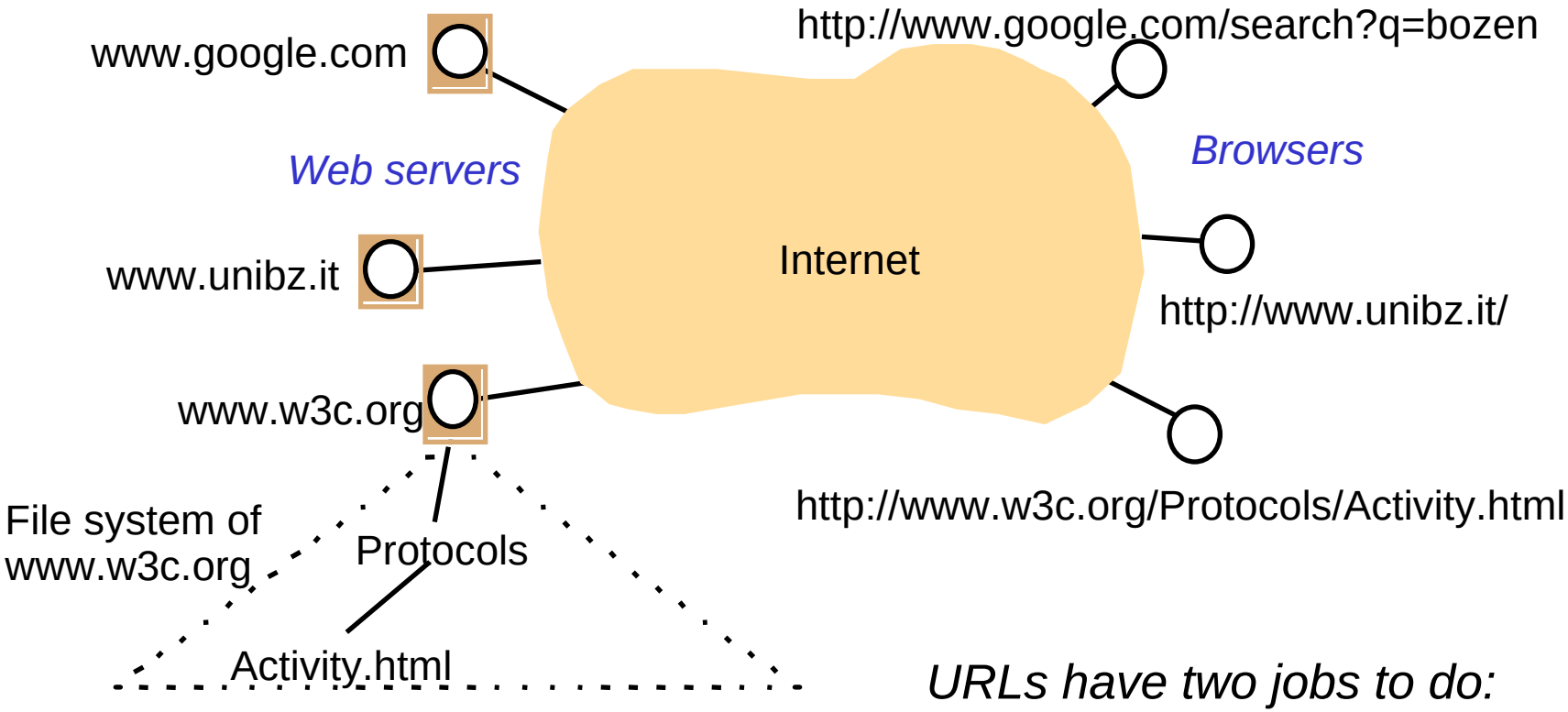
System for publishing and accessing resources across the internet

Open system

- The WWW can be reimplemented and extended in many ways
  - many browsers, web servers with increasing capabilities
  - on all kinds of platforms
- Increasing number of resource types can be published
  - data
  - services

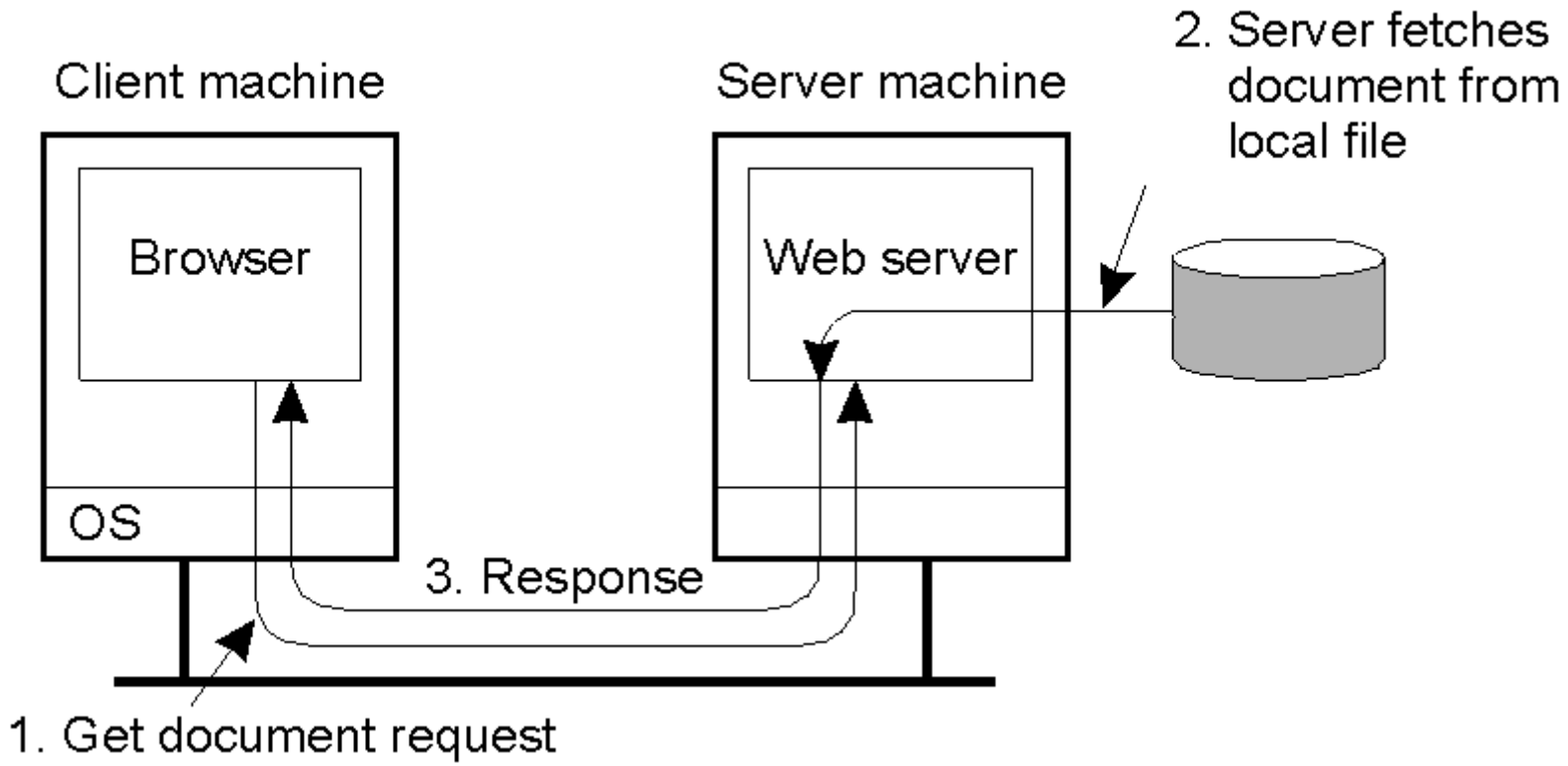
- Basis: Document Model: HTML (XML, MIME types)
  Resource Location: URLs
  Interaction: HTTP

Web Servers and Web Browsers



URLs have two jobs to do:
- identify web server
- identify resource

The Basic Web Mechanism



Refinements exist both at the client and the server side

HTML Document with Head

```
<!doctype html public "-//w3c//dtd html 4.0 transitional//en">
<html>
<head>
   <meta HTTP-EQUIV="Content-Type" CONTENT="text/html; CHARSET=iso-8859-1">
   <meta NAME="GENERATOR" CONTENT="Mozilla/4.7 (Macintosh; I; PPC) [Netscape]">
   <title>Werner Nutt</title>
</head>

<body TEXT="#000000" BGCOLOR="#FFFFFF" LINK="#3333FF"
      VLINK="#3366FF"ALINK="#CC3232">
<center><a href="index.html"><img SRC="Pictures/WernerNutt.jpg"
      ALT="WN" BORDER=0 HEIGHT=100 WIDTH=270></a>
<br> 
<br> 
<br> 
<a href="http://www.provincia.bz.it/wetter/suedtirol.htm">
<img SRC="http://www.provincia.bz.it/wetter/images/m_suedtirol_deutsch.jpg"
     ALT="Weather" BORDER=0></a>
</center>

</body>
</html>
```

Document Model: HTML

HTML is a markup language for hypertext documents

Mobile code can be embedded as JavaScript or Java Applet

```
<HTML>                                          <!- Start of HTML document    -->
<BODY>                                          <!- Start of the main body      -->
<H1>Hello World/H1>                             <!- Basic text to be displayed  -->
<P>                                             <!- Start of a new paragraph   -->
<SCRIPT type = "text/javascript">               // Identify scripting language
alert ('Hello World');                          // Create Alert Window
</SCRIPT>                                       <!- End of scripting section    -->
</P>                                            <!- End of paragraph section -->
</BODY>                                         <!- End of main body           -->
</HTML>                                         <!- End of HTML section        -->
```

Document Model: XML (1)

In XML, new tags can be defined by document type definitions (DTDs) or XML schema declarations

```
(1)     <!ELEMENT article (title, author+,journal)>
(2)     <!ELEMENT title (#PCDATA)>
(3)     <!ELEMENT author (name, affiliation?)>
(4)     <!ELEMENT name (#PCDATA)>
(5)     <!ELEMENT affiliation (#PCDATA)>
(6)     <!ELEMENT journal (jname, volume, number?, month? pages, year)>
(7)     <!ELEMENT jname (#PCDATA)>
(8)     <!ELEMENT volume (#PCDATA)>
(9)     <!ELEMENT number (#PCDATA)>
(10)    <!ELEMENT month (#PCDATA)>
(11)    <!ELEMENT pages (#PCDATA)>
(12)    <!ELEMENT year (#PCDATA)>
```

Document Model: XML (2)

An XML document using the tags defined in the DTD

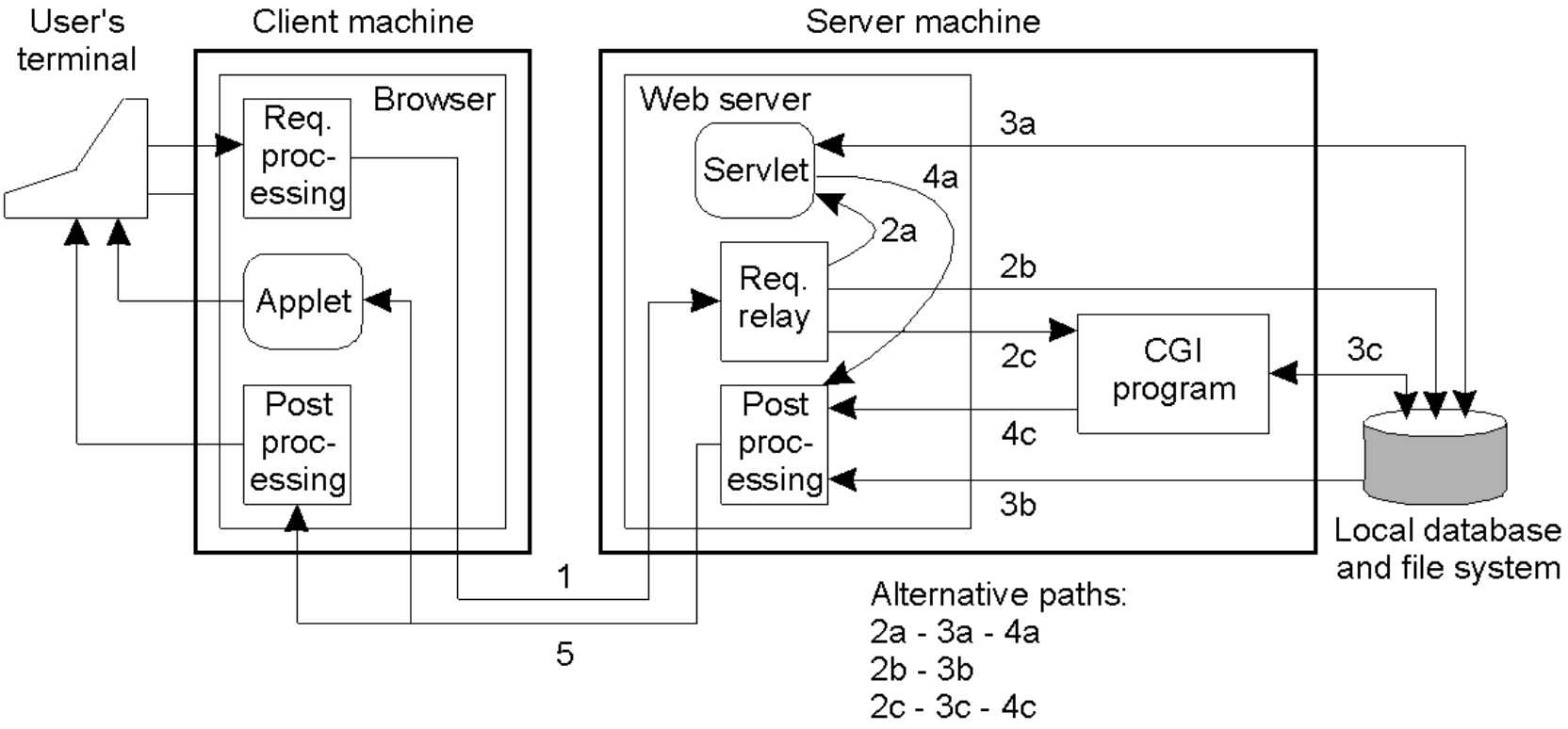
```
(1)   <?xml = version "1.0">
(2)   <!DOCTYPE article SYSTEM "article.dtd">
(3)   <article>
(4)      <title> Prudent Engineering Practice for Cryptographic Protocols</title>
(5)      <author><name>M. Abadi</name></author>
(6)      <author><name>R. Needham</name></author>
(7)      <journal>
(8)            <jname>IEEE Transactions on Software Engineering</jname>
(9)            <volume>22</volume>
(10)           <number>12</number>
(11)           <month>January</month>
(12)           <pages>6 – 15</pages>
(13)           <year>1996</year>
(14)     </journal>
(15)  </article>
```

Document Model: MIME Types

Type	Subtype	Description
Text	Plain	Unformatted text
	HTML	Text including HTML markup commands
	XML	Text including XML markup commands
Image	GIF	Still image in GIF format
	JPEG	Still image in JPEG format
Audio	Basic	Audio, 8-bit PCM sampled at 8000 Hz
	Tone	A specific audible tone
Video	MPEG	Movie in MPEG format
	Pointer	Representation of a pointer device for presentations
Application	Octet-stream	An uninterrupted byte sequence
	Postscript	A printable document in Postscript
	PDF	A printable document in PDF
Multipart	Mixed	Independent parts in the specified order
	Parallel	Parts must be viewed simultaneously

Six top-level MIME types (= internet media types)
and some subtypes

The Web Mechanism: Details



URLs

- URL = *Uniform Resource Locator*

  standard mechanism to identify resources

  exists for several access schemes (protocols)

- Syntax:

  <scheme> : <scheme dependent info>

- Examples:

  http://www.isoc.org/internet/history/

  mailto:rossi@inf.unibz.it

  ftp://mcnutt:blabla@izanami.macs.hw.ac.uk

URLs

- Hyper Text Transfer Protocol (HTTP) for Web

  http:// *host*[:*port*][/*path*][?*arguments*]

- File Transfer Protocol (FTP)

  ftp://[*user*[:*password*@]]*host*[:*port*][*path*]

- OS accessible files (file)

  file://*host*/*path*

URLs

- Email (mailto)
  mailto:*account@site*
- Telnet (telnet)
  telnet://*user@host*
- Others:
  news and NNTP (newsgroups),
  jar (Java jar files), imap (mail servers)

Browser

- generates hypertext view
- or calls helper application

HTTP

- Request-reply protocol for transferring documents
- Documents
  - may be of different types (MIME types)
  - may have embedded documents
- Several types of request messages
- One resource per request

  one page may imply *many requests*,

  e.g., one for the html and one for every image

APIs in various programming languages, e.g.,
class **`HttpURLConnection`** in **`java.net`**

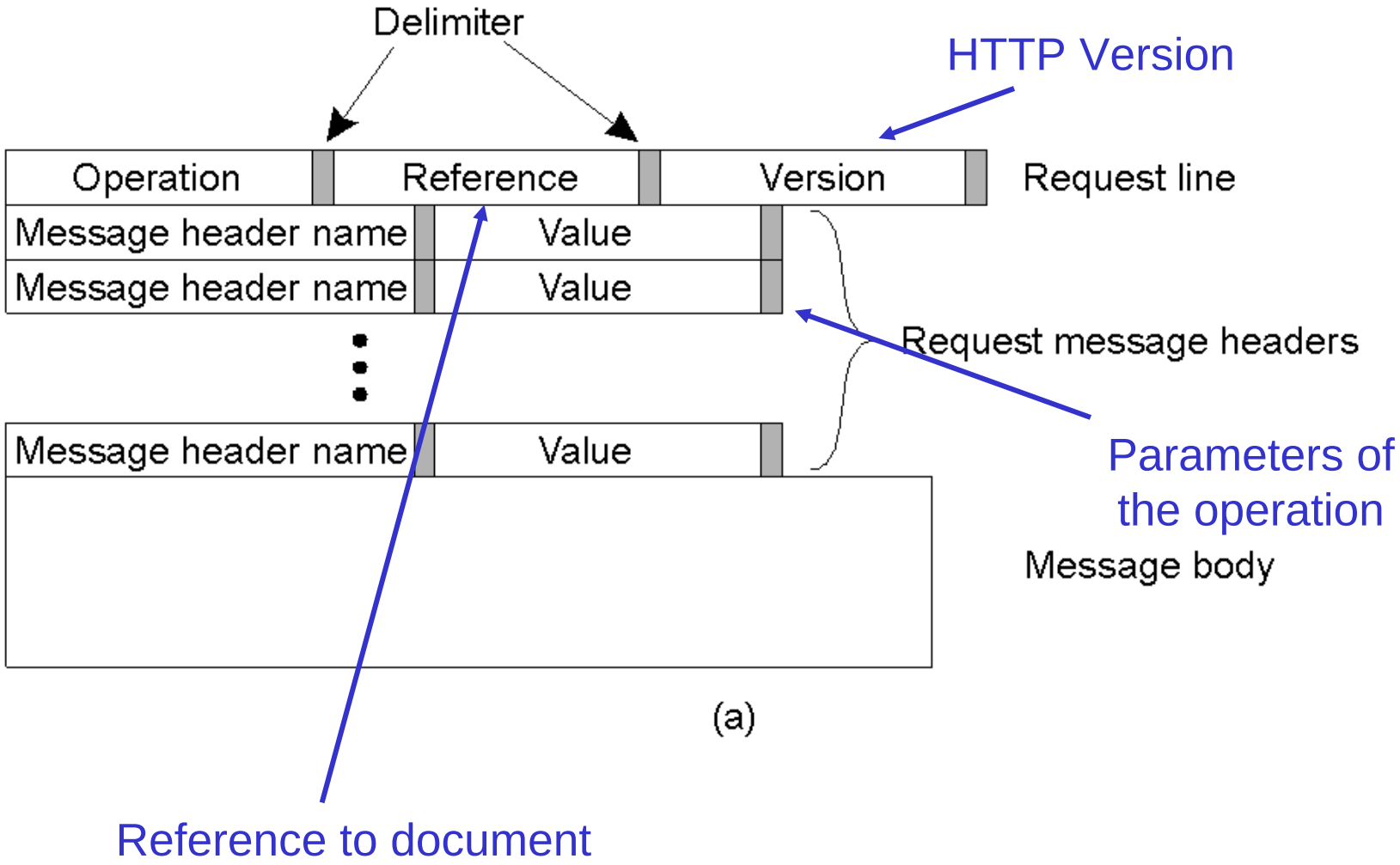
HTTP Operations

Commonly used HTTP requests

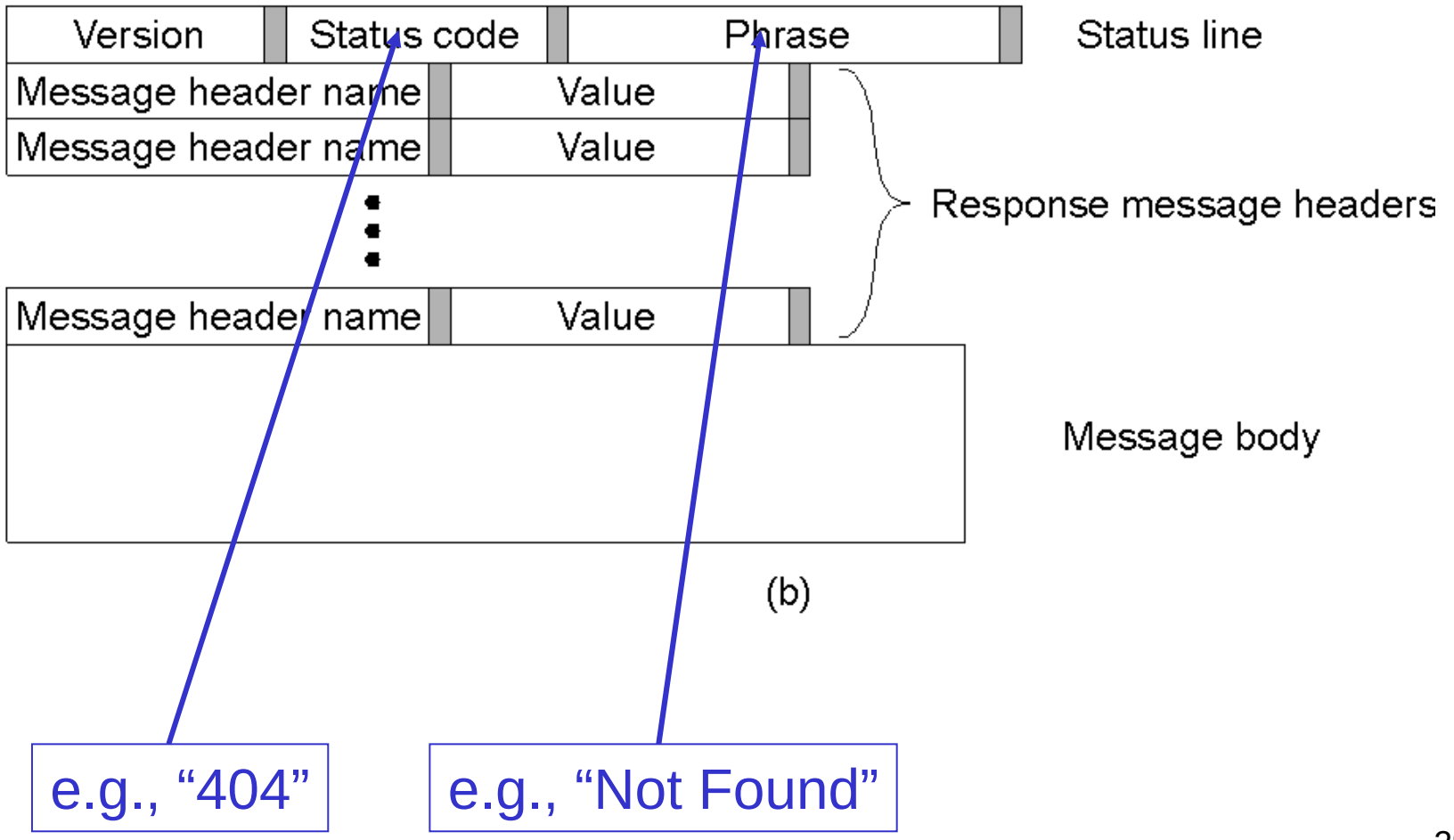
Operation	Description
Head	Request to return the header of a document
Get	Request to return a document to the client
Put	Request to store a document
Post	Provide data that is to be added to a document (collection)
Delete	Request to delete a document

Requests are part of messages

HTTP Request Message



HTTP Response Message



Some HTTP Message Headers

Header	Source	Contents
Accept	Client	The type of documents the client can handle
Accept-Language	Client	The natural language the client can handle
WWW-Authenticate	Server	Security challenge the client should respond to
Date	Both	Date and time the message was sent
Expires	Server	The time how long the response remains valid
Host	Client	The TCP address of the document's server
If-Modified-Since	Client	Tells the server to return a document only if it has been modified since the specified time
Last-Modified	Server	The time the returned document was last modified
Location	Server	A document reference to which the client should redirect its request
Referer	Client	Refers to client's most recently requested document
Upgrade	Both	The application protocol the sender wants to switch to
Warning	Both	Information about the status of the data in the message

HTTP Conversation: Request

Client:

```
GET /index.html HTTP/1.1
Host: www.example.com
```

HTTP Conversation: Reply

Server:

```
HTTP/1.1 200 OK
Date: Mon, 23 May 2005 22:38:34 GMT
Server: Apache/1.3.27 (Unix)  (Red-Hat/Linux)
Last-Modified: Wed, 08 Jan 2003 23:11:55 GMT
Etag: "3f80f-1b6-3e1cb03b"
Accept-Ranges: bytes
Content-Length: 438
Connection: close
Content-Type: text/html; charset=UTF-8
```

[HTML Document]

Distributed Systems Challenges

Developers of distributed systems have to cope with challenges:

1. Heterogeneity
2. Openness
3. Security
4. Scalability
5. Failure handling
6. Concurrency
7. Transparency

Heterogeneity

- Heterogeneity appers at several **levels**:
  - Network (Ethernet, token ring, ISDN,...)
  - Computing hardware (data representation)
  - Operating systems (different APIs to protocols)
  - Programming languages (data structures, APIs)
  - Applications by different developers

    (data exchange standards)

- **Middleware**:

  Software layer which abstracts from the above providing a uniform computational model (CORBA, Java RMI, ODBC, Web Services...)

Openness

- The degree to which a computer system can be extended and re-implemented

Measures:

- Publication of key interfaces
- Uniform communication mechanisms: protocols

Principle:

- Heterogeneous hardware and software, but
- Conformance to published standards

Openness (cntd.)

Examples of standardization organisations:

- ISO = International Standards Organisation
- ITU-T = International Telecommunication Union - Telecommunication Standardization Sector

  e.g., G.992.1 ADSL DMT
- IETF = Internet Engineering Task force

  e.g., RFC 791 specifies IPv4 (RFC = Request for Comments)
- IEEE = Institute of Electrical and Electronic Engineers)

  e.g., IEEE 802.11 WLAN, IEEE 802.3 Ethernet
- W3C = World Wide Web Consortium

  e.g., HTML Recommendations

Security

Security Aspects and violations:

- **Confidentiality**: disclosure of the contents of a message to a party different from the intended receiver
  Example: packet sniffing
- **Integrity**: corruption of the transmitted contents by a third party
  Example: man in the middle attack
  (➔ Encryption and Authentication)
- **Availability**: interference with a communication
  Example: denial of service attack
- **Security of Mobile Code:** harmful actions

Scalability

- A distributed system is scalable if
  - it operates effectively and efficiently independently from the number of resources and users connected to it

- Challenges:
  - Keep costs of physical resources proportional to growth
  - Minimize performance loss
  - Prevent software resources from running out
    (e.g., IP addresses)
  - Avoid performance bottlenecks
    (e.g., domain name service)

Failure Handling

Failures in distributed systems are partial!

- Failure Detection
  Example: message checksum
- Failure Masking
  Example: email retransmission
- Fault Tolerance

  Example: array of servers

Measures

- *Detected* failures may be recovered and *masked*
- Redundancy may improve *fault tolerance*

Concurrency

Becomes a problem when

- two or more parties
- access a the same resources

Approach:

- Control schemes under which concurrent operations behave *as if* they were isolated

Transparency

"Let system appears as a whole rather than a collection of independent components"

ISO [1992] has defined eight forms of transparency for distributed systems

ISO's Eight Forms of Transparency

- Access Transparency: Local and remote resources are accessed using identical operations.
- Location Transparency: Resources can be accessed without knowing their location.
- Concurrency Transparency: Several processes can operate concurrently using shared resources without interference between them.
- Replication Transparency: Multiple instances of resources can be used to increase reliability and performance without knowledge of the replicas by users or application programmers.

ISO's Eight Forms of Transparency

- Failure Transparency: Faults are being concealed, allowing users and application programs to complete their tasks despite the failure of hardware or software components.
- Mobility Transparency: Resources and clients are moved within a system without affecting the operation of users or programs.
- Performance Transparency: The system is being reconfigured to improve performance as loads vary.
- Scaling Transparency: System and applications expand in scale without change to the system structure or the application algorithms.

References

In preparing the lectures I have used several sources.
The main ones are the following:

Books:

- Coulouris, Dollimore, Kindberg. Distributed Systems – Concepts and Design (CDK)
- Tannenbaum, van Steen. Distributed Systems – Principles and Paradigms

Slides:

- Marco Aiello, course on Distributed Systems at the Free University of Bozen-Bolzano
- Andrew Tanenbaum, Slides from his website
- CDK Website

URI

URI = *Uniform Resource Identifier*

- Standard mechanism to identify electronic resources independently from their host location
- Union of
  - URL
  - URN (= *Uniform Resource Name*)
  - URC (= *Uniform Resource Characteristic*)

… only URLs are currently being resolved by name services