# Computer Networks - *Xarxes de Computadors*

## Outline

- Course Syllabus
- Unit 1: Introduction
- **Unit 2. Network applications**
- Unit 3. IP Networks
- Unit 4. TCP
- Unit 5. LANs

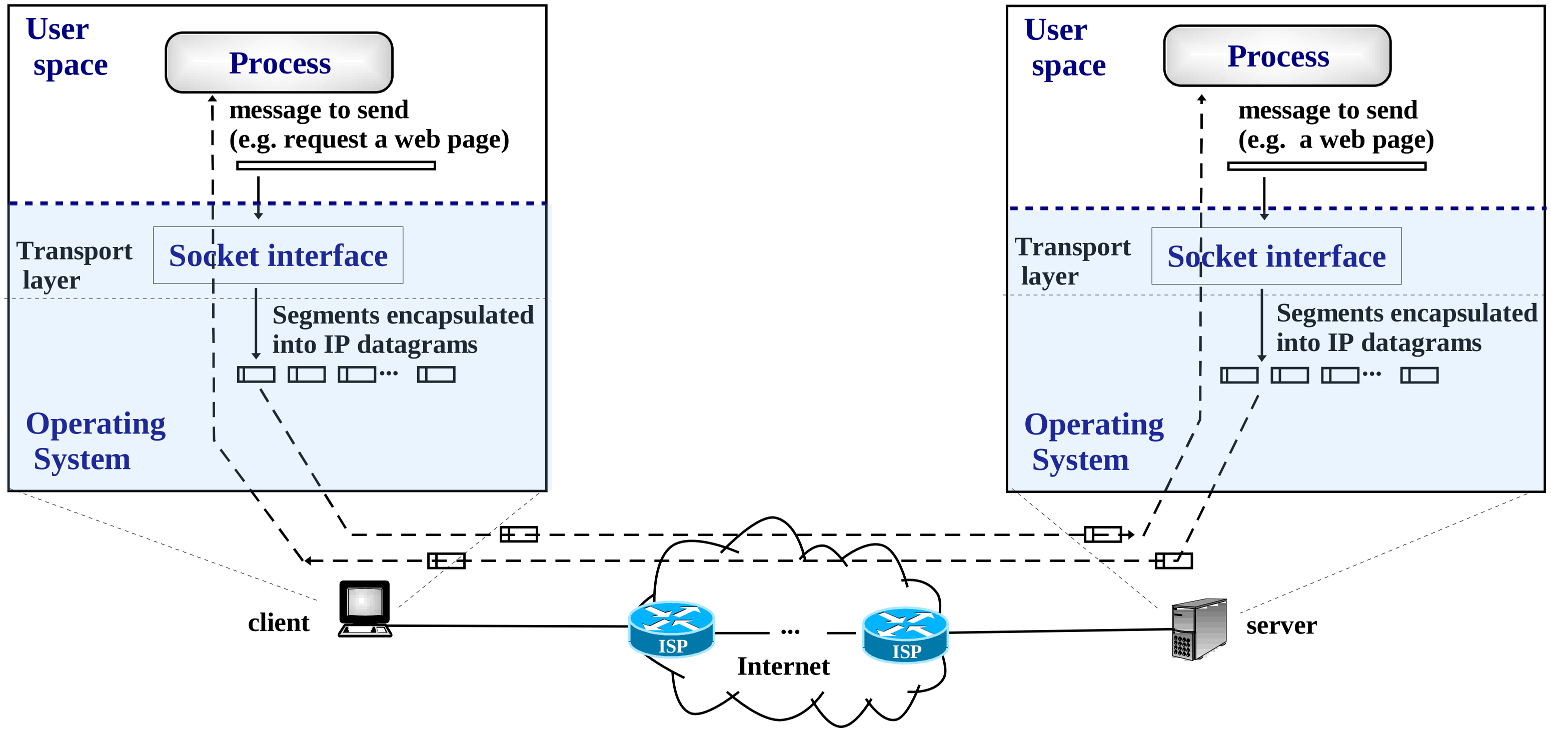
# Unit 2. Network applications

## Outline

- **Client-Server Paradigm**
- DNS
- Email
- Web
- HTML & XML

# Unit 2. Network applications

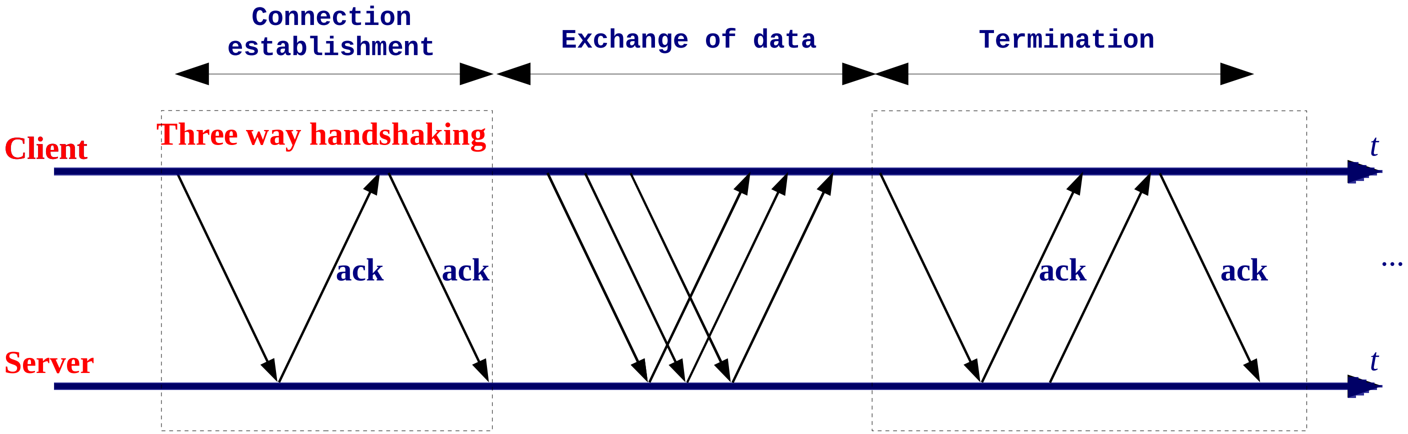
## Client Server Paradigm: Processes, messages, sockets segments and IP datagrams



# Unit 2. Network applications

## Client Server Paradigm: The Internet Transport Layer

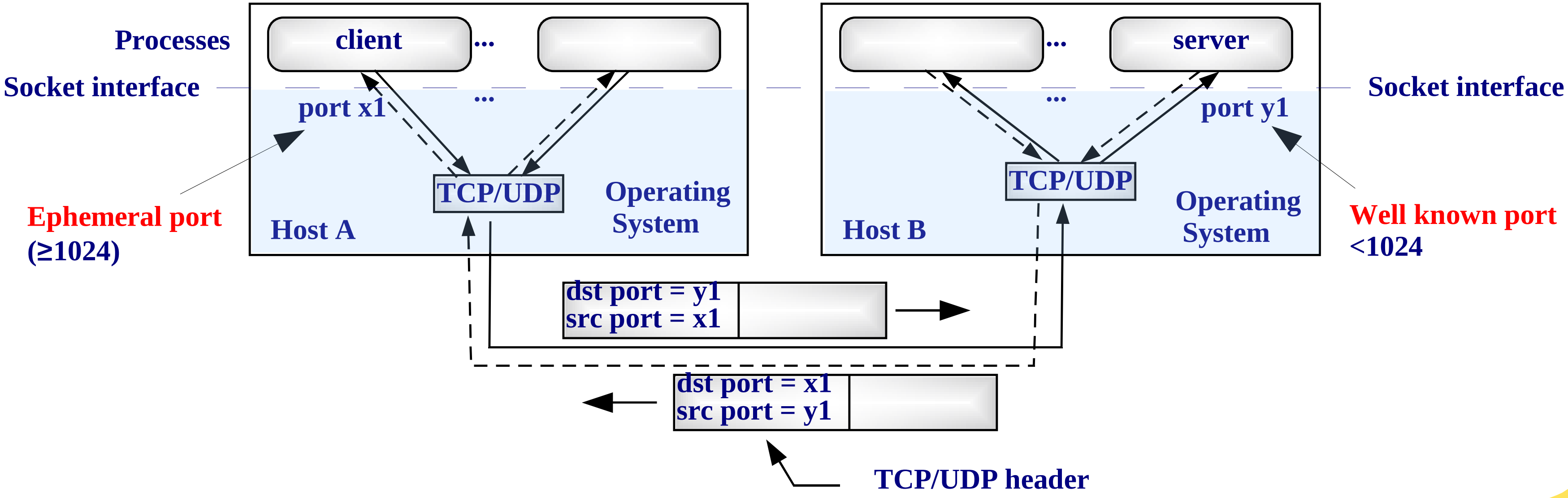
- Two protocols are used at the TCP/IP transport layer: User Datagram Protocol (UDP) and Transmission Control Protocol (TCP).
- UDP offers a *datagram service* (non reliable). It is connectionless.
- TCP offers a reliable service (correct segments are acknowledged, ack, lost segments are retransmitted). It is connection oriented (covered in detail in Unit 4).
- TCP connection:



# Unit 2. Network applications

## Client Server Paradigm

- How connection is established among processes?
- The client always initiates the connection towards a known IP address, in the IP header, and a *well known* port (< 1024), in the TCP/UDP header.
- Well known ports are standardized by IANA in RFC-1700 (Assigned Numbers). In a unix machine can be found in /etc/services.
- The server is a *daemon* waiting for client requests.



# Unit 2. Network applications

## Client Server Paradigm – UNIX /etc/services File

- Enables server and client programs to convert service names to well known ports.

```
linux> cat /etc/services
# Network services, Internet style
# Note that it is presently the policy of IANA to assign a single well-known
# port number for both TCP and UDP; hence, most entries here have two entries
# even if the protocol doesn't support UDP operations.
# This list could be found on:
#            http://www.iana.org/assignments/port-numbers
# ************************************************************************
# WELL KNOWN PORT NUMBERS
# The Well Known Ports are assigned by the IANA and on most systems can
# only be used by system (or root) processes or by programs executed by
# privileged users.
#
# Keyword  Decimal  Description
# -------  -------  -----------
echo         7/tcp  Echo
echo         7/udp  Echo
discard      9/tcp  # Discard
discard      9/udp  # Discard
daytime     13/tcp  # Daytime (RFC 867)
daytime     13/udp  # Daytime (RFC 867)
chargen     19/tcp  # Character Generator
chargen     19/udp  # Character Generator
ftp-data    20/tcp  # File Transfer [Default Data]
ftp-data    20/udp  # File Transfer [Default Data]
ftp         21/tcp  # File Transfer [Control]
ssh         22/tcp  # SSH Remote Login Protocol
ssh         22/udp  # SSH Remote Login Protocol
telnet      23/tcp  # Telnet
telnet      23/udp  # Telnet
...
```

# Unit 2. Network applications

## Client Server Paradigm – Network applications

- Remote commands
  - telnet
  - ssh
- Exchange of documents
  - ftp, sftp
  - peer-to-peer
- Web based applications
- Email
- Domain name system, DNS
- Network management
- Real time
  - Voice over IP
  - Video streaming
- ...

# Unit 2. Network applications

## Outline

- Client-Server Paradigm
- **DNS**
- Email
- Web
- HTML & XML

# Unit 2. Network applications

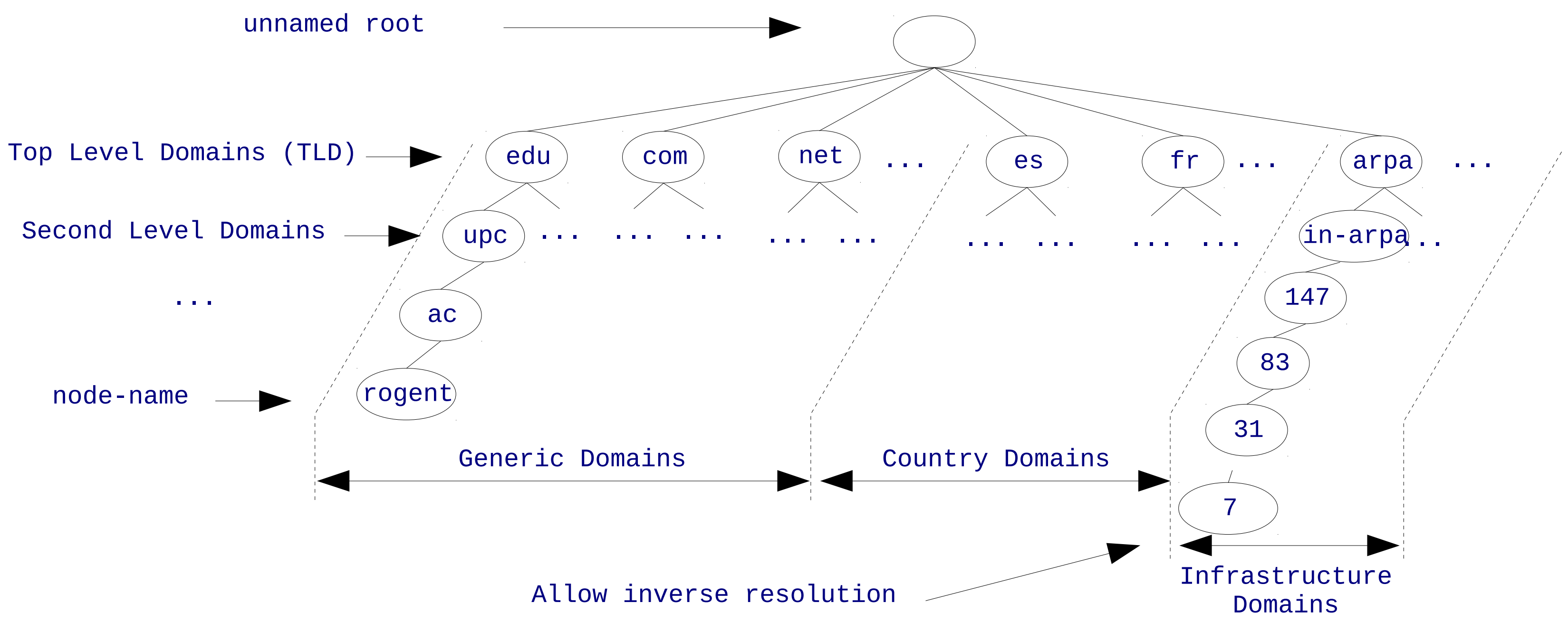
## Domain Name System DNS (RFC 1034, 1035)

- Allows users to use names instead of IP addresses: e.g. rogent.ac.upc.edu instead of 147.83.31.7, www.upc.edu instead of 147.83.194.21, etc.
- Names consists of a node-name and a domain-mane: rogent.ac.upc.edu, www.upc.edu
- DNS consists of a worldwide distributed data base.
- DNS data base entries are referred to as *Resource Records* (RR).
- The information associated with a name is composed of 1 or more RRs.
- Names are case insensitive (e.g. www.upc.edu and WWW.UPC.EDU are equivalent).

# Unit 2. Network applications

## DNS – Domain Hierarchy

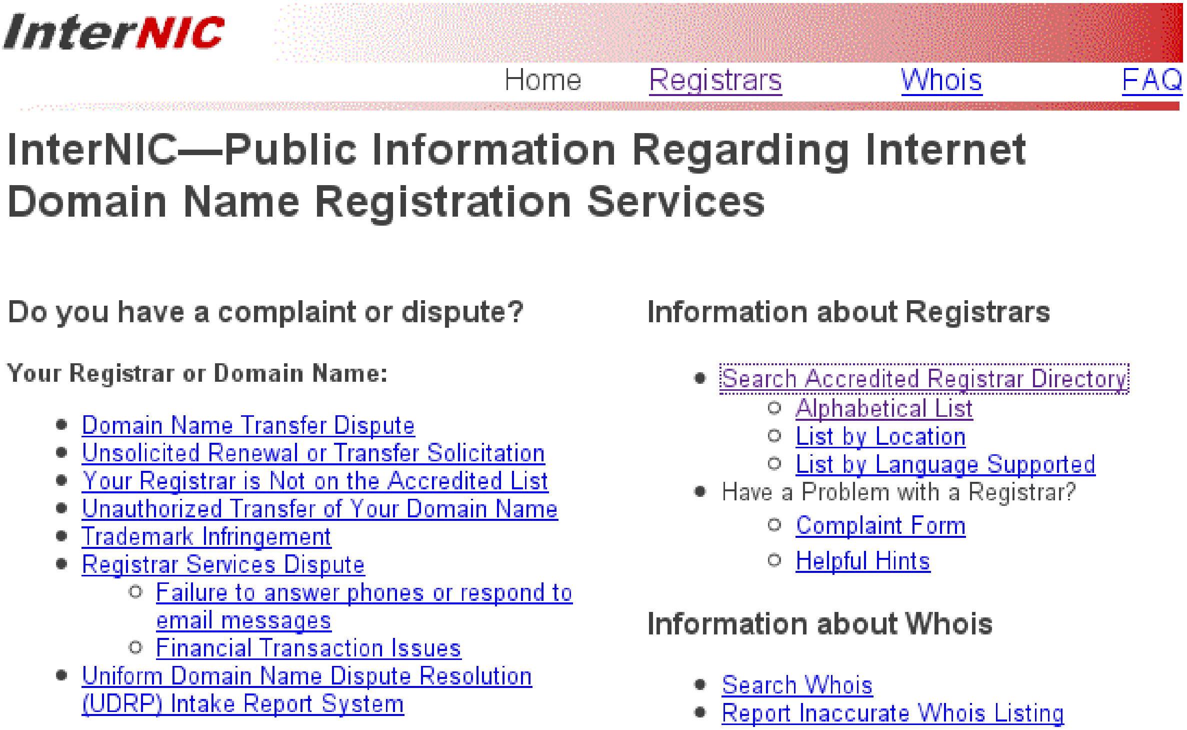
- DNS data base is organized in a tree:



# Unit 2. Network applications

## DNS – Domain Hierarchy

- The *Internet Corporation for Assigned Names and Numbers* (ICANN) is responsible for managing and coordinating the DNS.
- ICANN delegates Top Level Domains (TLD) administration to registrars: http://www.internic.net
- Domains delegate the administration of subdomains.



### InterNIC—Public Information Regarding Internet Domain Name Registration Services

**Do you have a complaint or dispute?**

**Your Registrar or Domain Name:**

- Domain Name Transfer Dispute
- Unsolicited Renewal or Transfer Solicitation
- Your Registrar is Not on the Accredited List
- Unauthorized Transfer of Your Domain Name
- Trademark Infringement
- Registrar Services Dispute
  - Failure to answer phones or respond to email messages
  - Financial Transaction Issues
- Uniform Domain Name Dispute Resolution (UDRP) Intake Report System

**Information about Registrars**

- Search Accredited Registrar Directory
  - Alphabetical List
  - List by Location
  - List by Language Supported
- Have a Problem with a Registrar?
  - Complaint Form
  - Helpful Hints

**Information about Whois**

- Search Whois
- Report Inaccurate Whois Listing

# Unit 2. Network applications

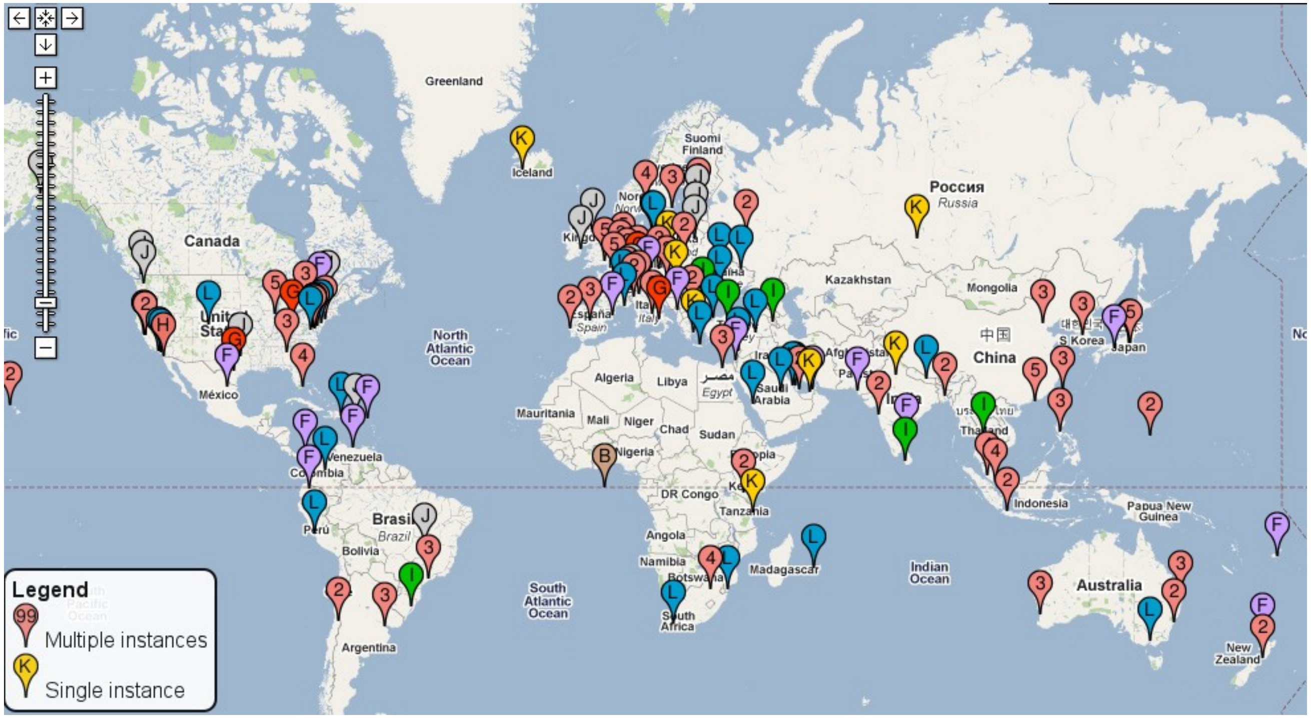
## DNS – Data Base Organization

- Access to DNS data base is done using *Name Servers* (NS).
- NSs may hold permanent and cached RRs. Cached RRs are removed after a timeout.
- Each subdomain has an *authority* which consists of a primary and backup NSs.
- In this context, subdomains are referred to as *zones*, and delegated subdomains *subzones*.
- An authority has the complete information of a zone:
  - Names and addresses of all nodes within the zone.
  - Names and addresses of all subzone authorities.

# Unit 2. Network applications

## DNS – Data Base Organization

- Root Servers are the entry point to the domain hierarchy.
- Root Servers are distributed around the world and have the TLD addresses: http://www.root-servers.org
- Root server addresses are needed in a NS configuration.



Source: http://www.root-servers.org

# Unit 2. Network applications

## DNS - Unix example: The resolver

- The applications use the calls (*resolver* library):

```
struct hostent *gethostbyname(const char *name) ;
struct hostent *gethostbyaddr(const void *addr, int len, int type);
```

- The resolver first looks the /etc/hosts file:

```
# hosts         This file describes a number of hostname-to-address
#               mappings for the TCP/IP subsystem.  It is mostly
#               used at boot time, when no name servers are running.
#               On small systems, this file can be used instead of a
#               "named" name server.
# Syntax:
# IP-Address  Full-Qualified-Hostname  Short-Hostname
127.0.0.1       localhost
10.0.1.1        massanella.ac.upc.edu massanella
```

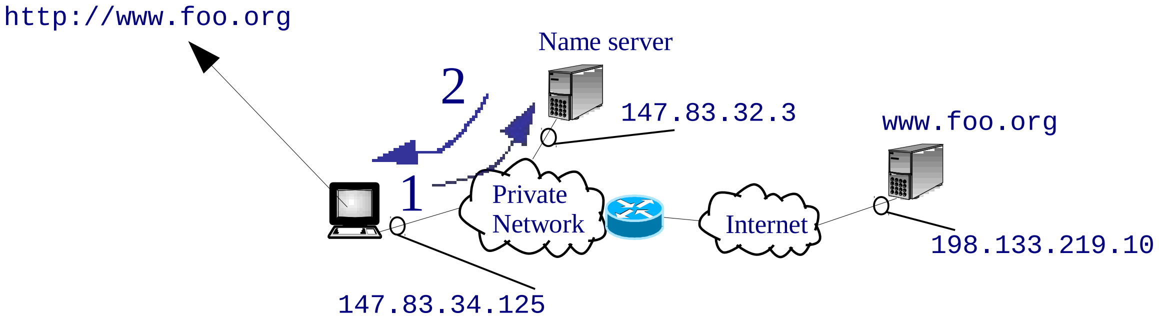
- Otherwise a *name server* is contacted using /etc/resolv.conf file:

```
search ac.upc.edu
nameserver 147.83.32.3
nameserver 147.83.33.4
```

# Unit 2. Network applications

## DNS - Protocol

- Client-server paradigm
- UDP/TCP. Short messages uses UDP.
- well-known port: 53



1
```
18:36:00.322370 IP (proto: UDP) 147.83.34.125.1333 >
       147.83.32.3.53:  53040+ A? www.foo.org. (31)
```

2
```
18:36:00.323080 IP (proto: UDP) 147.83.32.3.53 > 147.83.34.125.1333:
       53040 1/2/2 www.foo.org. A 198.133.219.10 (115)
```

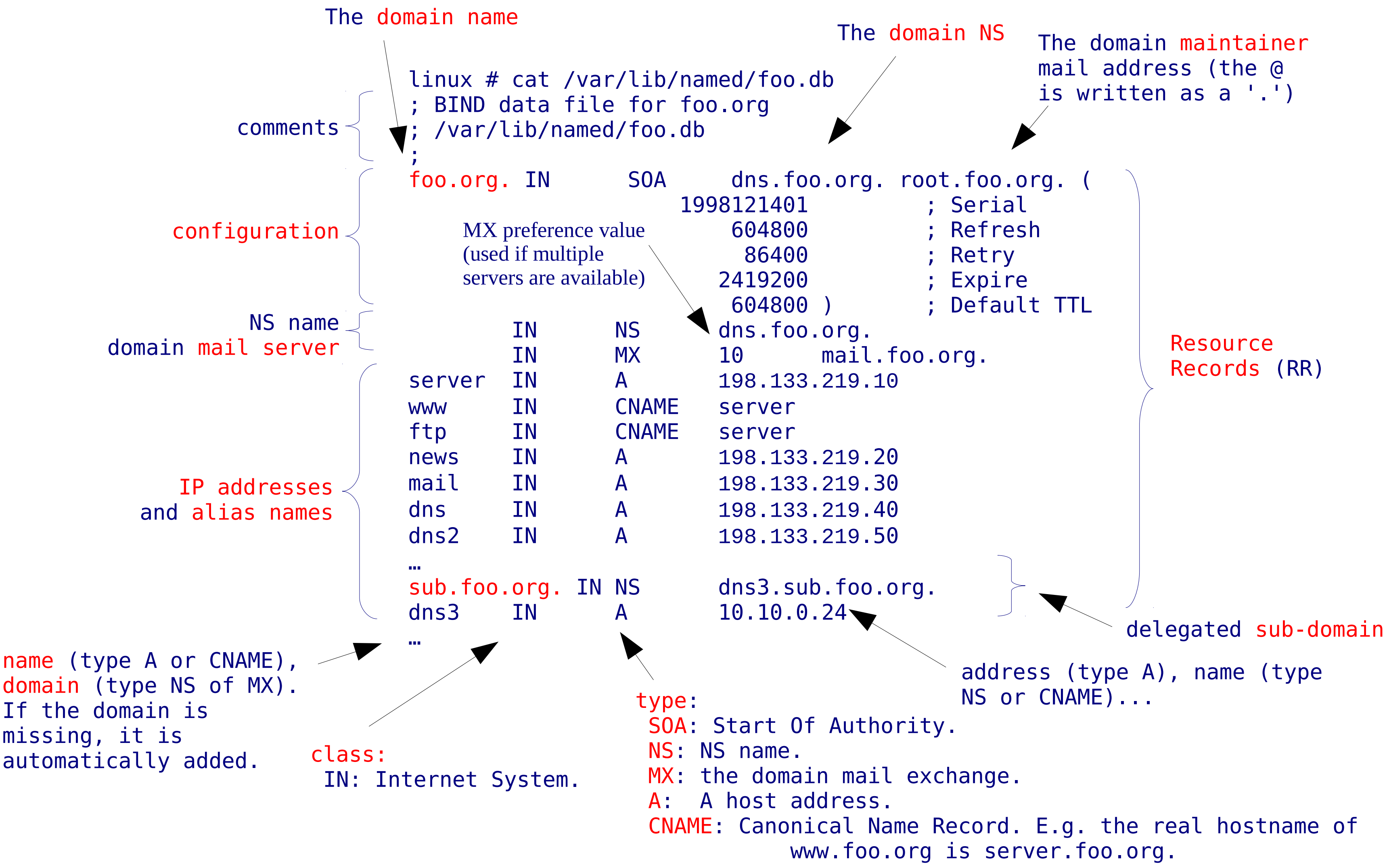
# Unit 2. Network applications

## DNS – Unix example: Basic NS configuration

- Unix NS implementation is BIND (Berkeley Internet Name Domain), http://www.isc.org.
- named is the BIND NS daemon.
- BIND basic configuration files:
  - `/etc/named.conf` global configuration
  - `/var/lib/named/root.hint` root servers addresses
  - `/var/lib/named/*.db` zone files

# Unit 2. Network applications

## DNS – Unix example: zone file



```
linux # cat /var/lib/named/foo.db
; BIND data file for foo.org
; /var/lib/named/foo.db
;
foo.org. IN      SOA     dns.foo.org. root.foo.org. (
                     1998121401         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Default TTL
        IN      NS      dns.foo.org.
        IN      MX      10      mail.foo.org.
server  IN      A       198.133.219.10
www     IN      CNAME   server
ftp     IN      CNAME   server
news    IN      A       198.133.219.20
mail    IN      A       198.133.219.30
dns     IN      A       198.133.219.40
dns2    IN      A       198.133.219.50
…
sub.foo.org. IN NS      dns3.sub.foo.org.
dns3    IN      A       10.10.0.24
…
```

The domain name

The domain NS

The domain maintainer mail address (the @ is written as a '.')

comments

configuration

MX preference value (used if multiple servers are available)

NS name

domain mail server

Resource Records (RR)

IP addresses and alias names

delegated sub-domain

name (type A or CNAME), domain (type NS of MX). If the domain is missing, it is automatically added.

class:
IN: Internet System.

type:
SOA: Start Of Authority.
NS: NS name.
MX: the domain mail exchange.
A: A host address.
CNAME: Canonical Name Record. E.g. the real hostname of www.foo.org is server.foo.org.

address (type A), name (type NS or CNAME)...

# Unit 2. Network applications

## DNS – Unix example: root servers addresses

```
linux # cat /var/lib/named/root.hint

;       This file holds the information on root name servers needed to
;       initialize cache of Internet domain name servers
;       (e.g. reference this file in the "cache  .  <file>"
;       configuration file of BIND domain name servers).
;
;       This file is made available by InterNIC
;       under anonymous FTP as
;           file                /domain/named.root
;           on server           FTP.INTERNIC.NET
;       -OR-                    RS.INTERNIC.NET
.                        3600000  IN  NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.      3600000  IN  A     198.41.0.4
.                        3600000  IN  NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.      3600000  IN  A     192.228.79.201
.                        3600000  IN  NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.      3600000  IN  A     192.33.4.12

...
.                        3600000  IN  NS    M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET.      3600000  IN  A     202.12.27.33
```

comments

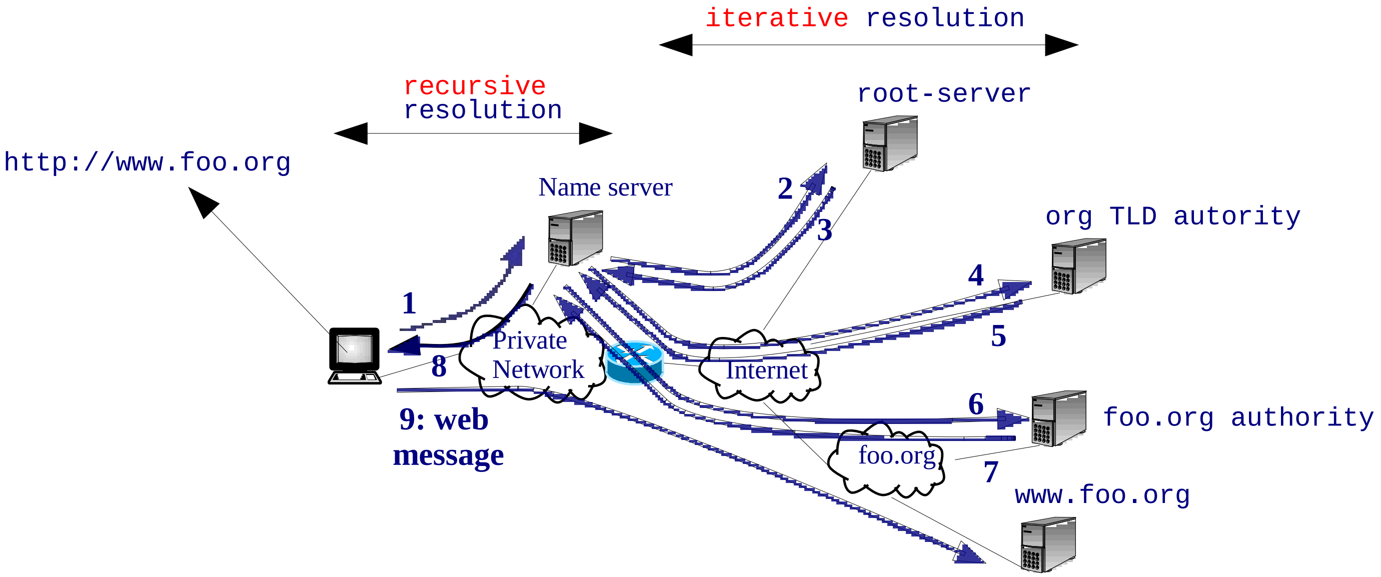
Resource Records (RR)
pointing to root-servers

address of a name
NS name

# Unit 2. Network applications

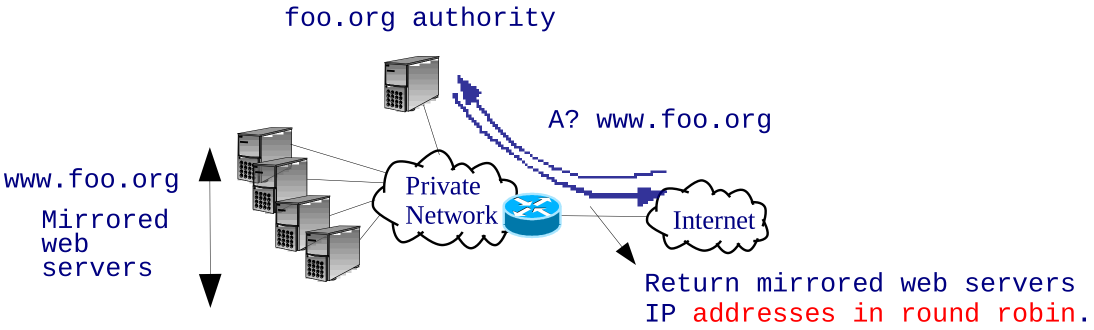
## DNS – Resolution

- NSs cache name resolutions.
- A cached RR is returned without looking for in the NS authority.
- The same name may be associated with several IP addresses (e.g. load balancing).
- The addresses of a common domain may not belong to the same IP network (e.g. Content Distribution Networks).



# Unit 2. Network applications

## DNS – Load balancing, example



foo.org authority

A? www.foo.org

www.foo.org

Mirrored
web
servers

Private
Network

Internet

Return mirrored web servers
IP addresses in round robin.

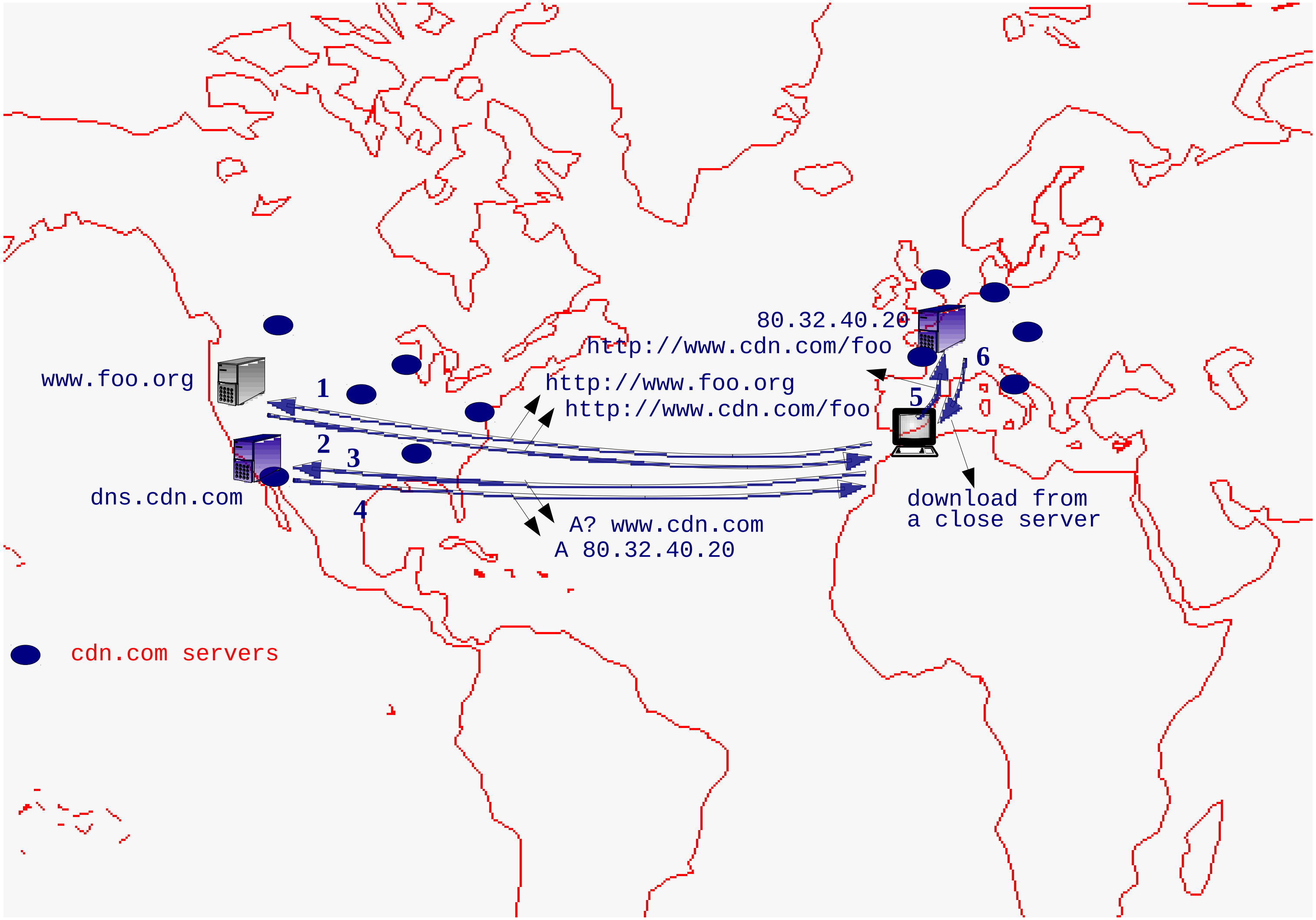
- Example using dig:

```
linux ~> dig www.microsoft.com

; <<>> DiG 9.3.2 <<>> www.microsoft.com
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 31808
;; flags: qr rd ra; QUERY: 1, ANSWER: 9, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;www.microsoft.com.             IN      A

;; ANSWER SECTION:
www.microsoft.com.      3135    IN      CNAME   toggle.www.ms.akadns.net.
toggle.www.ms.akadns.net. 181   IN      CNAME   g.www.ms.akadns.net.
g.www.ms.akadns.net.    181     IN      CNAME   lb1.www.ms.akadns.net.
lb1.www.ms.akadns.net.  181     IN      A       207.46.19.60
lb1.www.ms.akadns.net.  181     IN      A       207.46.18.30
lb1.www.ms.akadns.net.  181     IN      A       207.46.20.60
lb1.www.ms.akadns.net.  181     IN      A       207.46.19.30
lb1.www.ms.akadns.net.  181     IN      A       207.46.198.30
lb1.www.ms.akadns.net.  181     IN      A       207.46.225.60

;; Query time: 42 msec
;; SERVER: 192.168.1.1#53(192.168.1.1)
;; WHEN: Sun Mar 11 10:48:11 2007
;; MSG SIZE  rcvd: 203
```

```
linux ~> dig www.microsoft.com

; <<>> DiG 9.3.2 <<>> www.microsoft.com
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 17923
;; flags: qr rd ra; QUERY: 1, ANSWER: 9, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;www.microsoft.com.             IN      A

;; ANSWER SECTION:
www.microsoft.com.      3469    IN      CNAME   toggle.www.ms.akadns.net.
toggle.www.ms.akadns.net. 215   IN      CNAME   g.www.ms.akadns.net.
g.www.ms.akadns.net.    215     IN      CNAME   lb1.www.ms.akadns.net.
lb1.www.ms.akadns.net.  215     IN      A       207.46.198.30
lb1.www.ms.akadns.net.  215     IN      A       207.46.199.30
lb1.www.ms.akadns.net.  215     IN      A       207.46.18.30
lb1.www.ms.akadns.net.  215     IN      A       207.46.19.60
lb1.www.ms.akadns.net.  215     IN      A       207.46.198.60
lb1.www.ms.akadns.net.  215     IN      A       207.46.20.60

;; Query time: 43 msec
;; SERVER: 192.168.1.1#53(192.168.1.1)
;; WHEN: Sun Mar 11 10:42:38 2007
;; MSG SIZE  rcvd: 203
```

# Unit 2. Network applications

## DNS - Content Distribution Networks, example



# Unit 2. Network applications

## DNS – Messages: Message Format

- All DNS messages have the same format:
  - Header: type of message.
  - Question: What is to be resolved.
  - Answer: Answer to question.
  - Authority: Domain authority names.
  - Additional: Typically, the authority name's addresses.

```
-------------------------------------------------
|             Header (12 bytes)                 |
-------------------------------------------------
/             Question (variable)               /
-------------------------------------------------
/             Answer (variable)                 /
-------------------------------------------------
/             Authority (variable)              /
-------------------------------------------------
/             Additional (variable)             /
-------------------------------------------------
```

# Unit 2. Network applications

## DNS – Messages: Header

- Identification: 16 random bits used to match query/response
- Flags. Some of them:
  - Query-Response, QR: 0 for query, 1 for response.
  - Authoritative Answer, AA: When set, indicates an authoritative answer.
  - Recursion Desired, RD: When set, indicates that recursion is desired.
- The other fields indicate the number of Questions, Answer, Authority and Additional fields of the message.

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 bits
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Identification        |      Flags                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         #Questions            |      #Answers                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         #Authorities          |      #Additional              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

# Unit 2. Network applications

## DNS – Messages: Question

- QName: Indicates the name to be resolved.
- QType: Indicates the question type:
  - Address, A.
  - Name Server, NS.
  - Pointer, PTR: For an inverse resolution.
  - Mail Exchange, MX: Domain Mail Server address.
- Qclass: For Internet addresses is 1.

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 bits
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                       QName (variable)                        /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         QType                 |      QClass                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 bytes
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|6|r|o|g|e|n|t|2|a|c|3|u|p|c|3|e|d|u|0|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Codification example of `rogent.ac.upc.edu`

# Unit 2. Network applications

## DNS – Messages: Resource Records (RRs)

- The fields Answer, Authority and  Additional are composed of RRs:
  - Name, Type, Class: The same as in the Question field.
  - TTL (Time To Live): Number of seconds the RR can be cached.
  - RDLenth: RR size in bytes.
  - Rdata: E.g. An IP address if the Type is 'A', or a name if the Type is 'NS', 'MX' or 'CNAME'.

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 bits
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                        Name (variable)                         /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Type                 |       Class                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             TTL                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         RDLenth               |      RData (variable)         /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

# Unit 2. Network applications

## DNS – Messages: Example

```
# tcpdump -s1500 -vvpni eth0 port 53
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 200 bytes
11:17:30.769328 IP (UDP, length: 55) 147.83.30.137.1042 > 147.83.30.70.53: 36388+ A? ns.uu.net. (27)
11:17:30.771324 IP (UDP, length: 145) 147.83.30.70.53 > 147.83.30.137.1042: 36388
                 q: A? ns.uu.net. 1/2/2 ns.uu.net. A 137.39.1.3
                 ns: ns.uu.net. NS auth00.ns.uu.net., ns.uu.net. NS auth60.ns.uu.net.
                 ar: auth00.ns.uu.net. A 198.6.1.65, auth60.ns.uu.net. A 198.6.1.181 (117)
```

Query message:

- 36388: Identifier.
- +: Recursion-Desired is set.
- A?: Qtype = A.
- ns.uu.net.: Name to resolve.

Response message:

- 36388: Identifier.
- q: A? ns.uu.net.: Repeat the Question field.
- 1/2/2: 1 Answers, 2 Authorities, 2 Additional follows.
- ns.uu.net. A 137.39.1.3: The answer (RR of type A, address: 137.39.1.3).
- ns: ns.uu.net. NS auth00.ns.uu.net., ns.uu.net. NS auth60.ns.uu.net.: 2 Authorities (RRs of type NS: the domain ns.uu.net. authorities are auth00.ns.uu.net. and auth60.ns.uu.net).
- ar: auth00.ns.uu.net. A 198.6.1.65, auth60.ns.uu.net. A 198.6.1.181: 2 Additional (RRs of type A: authorities IP addresses).

# Unit 2. Network applications

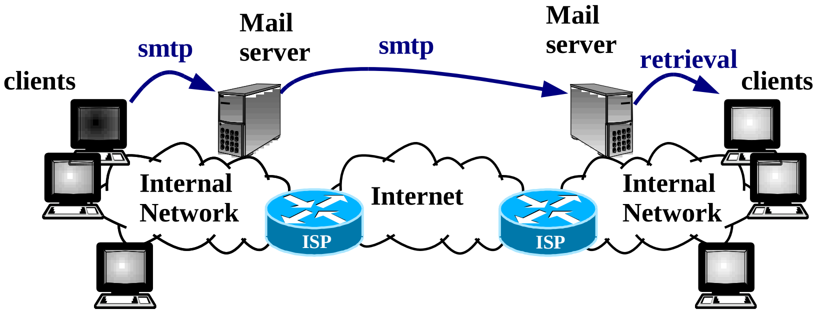
## Outline

- Client-Server Paradigm
- DNS
- **Email**
- Web
- HTML & XML

# Unit 2. Network applications

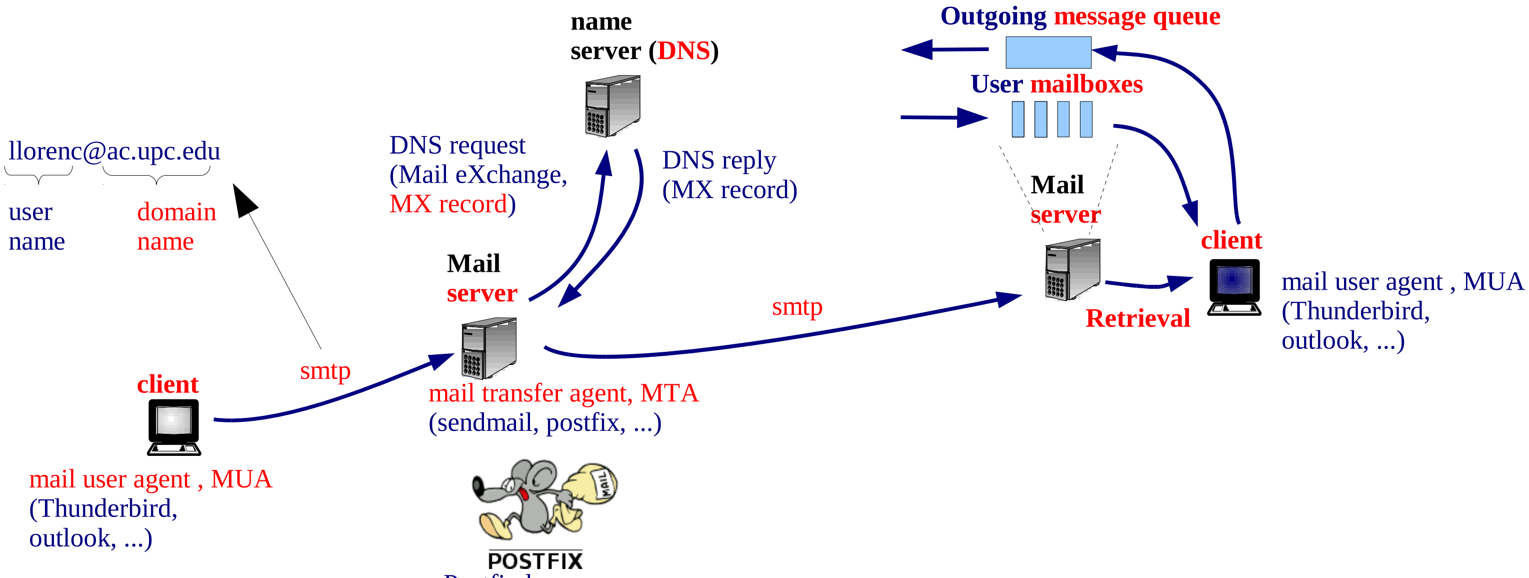
## Email

- Electronic mail (email): One of the first applications used in the Internet to electronic messaging.
- Components:
  - Transport layer: TCP, well-known port: 25.
  - Application layer protocol: Simple Mail Transfer Protocol (SMTP). First defined by RFC-821 and last updated by RFC-5321.
  - Retrieval protocols (IMAP, POP, HTTP).



# Unit 2. Network applications

## Email - SMTP processing model



llorenc@ac.upc.edu

user name

domain name

client

mail user agent , MUA (Thunderbird, outlook, ...)

smtp

Mail server

mail transfer agent, MTA (sendmail, postfix, ...)

name server (DNS)

DNS request (Mail eXchange, MX record)

DNS reply (MX record)

smtp

Outgoing message queue

User mailboxes

Mail server

Retrieval

client

mail user agent , MUA (Thunderbird, outlook, ...)

POSTFIX

Postfix logo
http://www.postfix.org
(UNIX, free and open-source)

# Unit 2. Network applications

## Email - SMTP protocol

- Designed as a simple (few commands) and text-based protocol (ASCII).
  - Client basic commands: HELO (identify SMTP client), MAIL FROM: (identify sender mailbox), RCPT TO: (identify recipient mailbox), DATA (mail message), QUIT (close transaction).
  - Server replies: Three digit number (identify what state the client to enter next), and a human understandable message.
- Example: Manually send an email using telnet to port 25.

CLIENT

SERVER

COMMANDS

SMTP transaction

```
pcmassanella ~> telnet relay.upc.edu 25
Trying 147.83.2.12...
Connected to relay.upc.edu.
Escape character is '^]'.
220 dash.upc.es ESMTP Sendmail 8.14.1/8.13.1; Fri, 4 Feb 2011 14:57:15 +0100
HELO pcmassanella.ac.upc.edu
250 dash.upc.es Hello pcmassanella.ac.upc.edu [147.83.34.125], pleased to meet you
MAIL FROM: <llorenc@ac.upc.edu>
250 2.1.0 <llorenc@ac.upc.edu>... Sender ok
RCPT TO: <llorenc@ac.upc.edu>
250 2.1.5 <llorenc@ac.upc.edu>... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself

Hello world
.
250 2.0.0 p14DvFOQ008320 Message accepted for delivery
QUIT
221 2.0.0 dash.upc.es closing connection
Connection closed by foreign host.
pcmassanella ~>
```

# Unit 2. Network applications

## Email – message formats

- Format described in RFC-5322 Internet Message Format (originally RFC-822)
- Example (extracted from the RFC):

```
From: John Doe <jdoe@machine.example>
To: Mary Smith <mary@example.net>
Subject: Saying Hello
Date: Fri, 21 Nov 1997 09:55:06 -0600
Message-ID: <1234@local.machine.example>

This is a message just to say hello.
So, "Hello".
```

**Header: gives information about the message. Fields defined in RFC-5322**

**Empty line**

**Body**

# Unit 2. Network applications

## Email – Extended message formats

- Multipurpose Internet Mail Extensions (MIME), RFC-2045, 2046, 2049
  - Inclussion of non-ASCI data (e.g. files, images, audio, video)
  - Multipart messages (extracted from the RFC-2049):

```
MIME-Version: 1.0
From: Nathaniel Borenstein <nsb@nsb.fv.com>
To: Ned Freed <ned@innosoft.com>
Date: Fri, 07 Oct 1994 16:15:05 -0700 (PDT)
Subject: A multipart example
Content-Type: multipart/mixed; boundary=unique-boundary-1

--unique-boundary-1
Content-type: text/plain; charset=US-ASCII

  ... Some text appears here ...

--unique-boundary-1
Content-Type: audio/basic
Content-Transfer-Encoding: base64

  ... base64-encoded 8000 Hz single-channel audio data goes here ...

--unique-boundary-1
Content-Type: image/jpeg
Content-Transfer-Encoding: base64

  ... base64-encoded image data goes here ...

--unique-boundary-1
Content-type: text/enriched

This is <bold><italic>enriched.</italic></bold> <smaller>as defined in RFC 1896</smaller>

--unique-boundary-1--
```

# Unit 2. Network applications

## Email - retrieval protocols

- Post Office Protocol (POP), RFC-1939:
  - POP server listens on well-known port 110
  - User normally deletes messages upon retrieval.
- Internet Message Access Protocol (IMAP) RFC-3501:
  - IMAP server listens on well-known port 143
  - Messages remain on the server until the user explicitly deletes them.
  - Provide commands to create folders, move messages, download only parts of the messages (e.g. only the headers)
- Web based Email (HTTP)
  - A web server handles users mailboxes. User agent is a web browser, thus, using HTTP to send and retrieve email messages.

# Unit 2. Network applications

## Outline

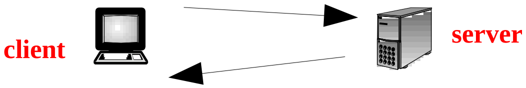
- Client-Server Paradigm
- DNS
- Email
- **Web**
- HTML & XML

# Unit 2. Network applications

## Web

- World Wide Web, www: was started by Tim John Berners-Lee in 1989 and developed in the 90s to provide an easy access to information in the Internet.
- Components:
  - Transport layer: TCP, well-known port: 80.
  - Application layer protocol: HyperText Transfer Protocol (HTTP). RFC1945 (HTTP-1.0), RFC2616 (HTTP-1.1).
  - HyperText Markup Language (HTML): Language used to format web documents.

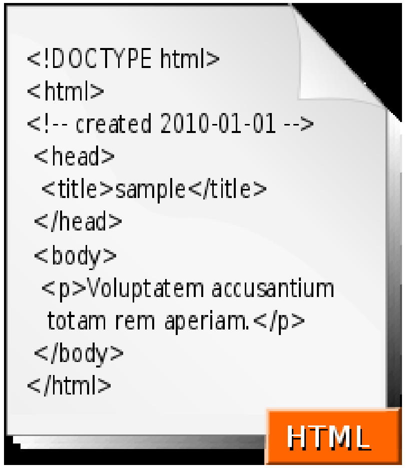
HTTP request



Web browser:
•firefox
•explorer
•...

HTTP response
(HTML page)

Web server:
•apache
•...

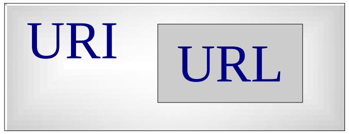


```
<!DOCTYPE html>
<html>
<!-- created 2010-01-01 -->
 <head>
  <title>sample</title>
 </head>
 <body>
  <p>Voluptatem accusantium
   totam rem aperiam.</p>
 </body>
</html>
```

Source: wikipedia

# Unit 2. Network applications

## Web – links



- Uniform Resource Identifier (URI) RFC3986
  - Generic syntax to identify a resource.
- Uniform Resource Locator (URL) RFC1738
  - Subset of URIs identifying the locating a resource in the Internet.
- The URL general syntax is

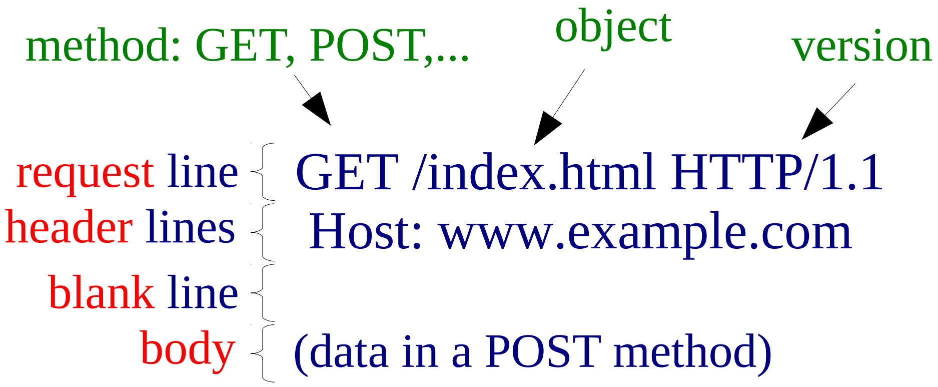
**scheme://username:password@domain:port/path?query_string#fragment_id**

  - scheme: Purpose, and the syntax of the remaining part. http, gopher, file, ftp...
  - domain name or IP address gives the destination location. The port is optional.
  - query_string: contains data to be passed to the server.
  - fragment_id: specifies a position in the html page.
  - Examples:
    - http://tools.ietf.org/html/rfc1738
    - http://147.83.2.135
    - http://studies.ac.upc.edu/FIB/grau/XC/#Practs
    - file:///home/llorenc/gestio/2010/cd/autors.html
    - http://www.amazon.com/product/03879/refs9?pf_ra=ATVPD&pf_rd=07HR2

# Unit 2. Network applications

## Web – HTTP Messages, RFC2616

- Client (HTTP request):



method: GET, POST,...

object

version

request line GET /index.html HTTP/1.1

header lines Host: www.example.com

blank line

body (data in a POST method)

- Methods:
  - GET: Typical command. Requests an object.
  - POST: Request an object qualified by the data in the body. This data is the contents of the HTML form fields, provided by the client.
  - ...
- Header: Allows the client to give additional information about the request and the client itself.

# Unit 2. Network applications

## Web – HTTP Messages, RFC2616

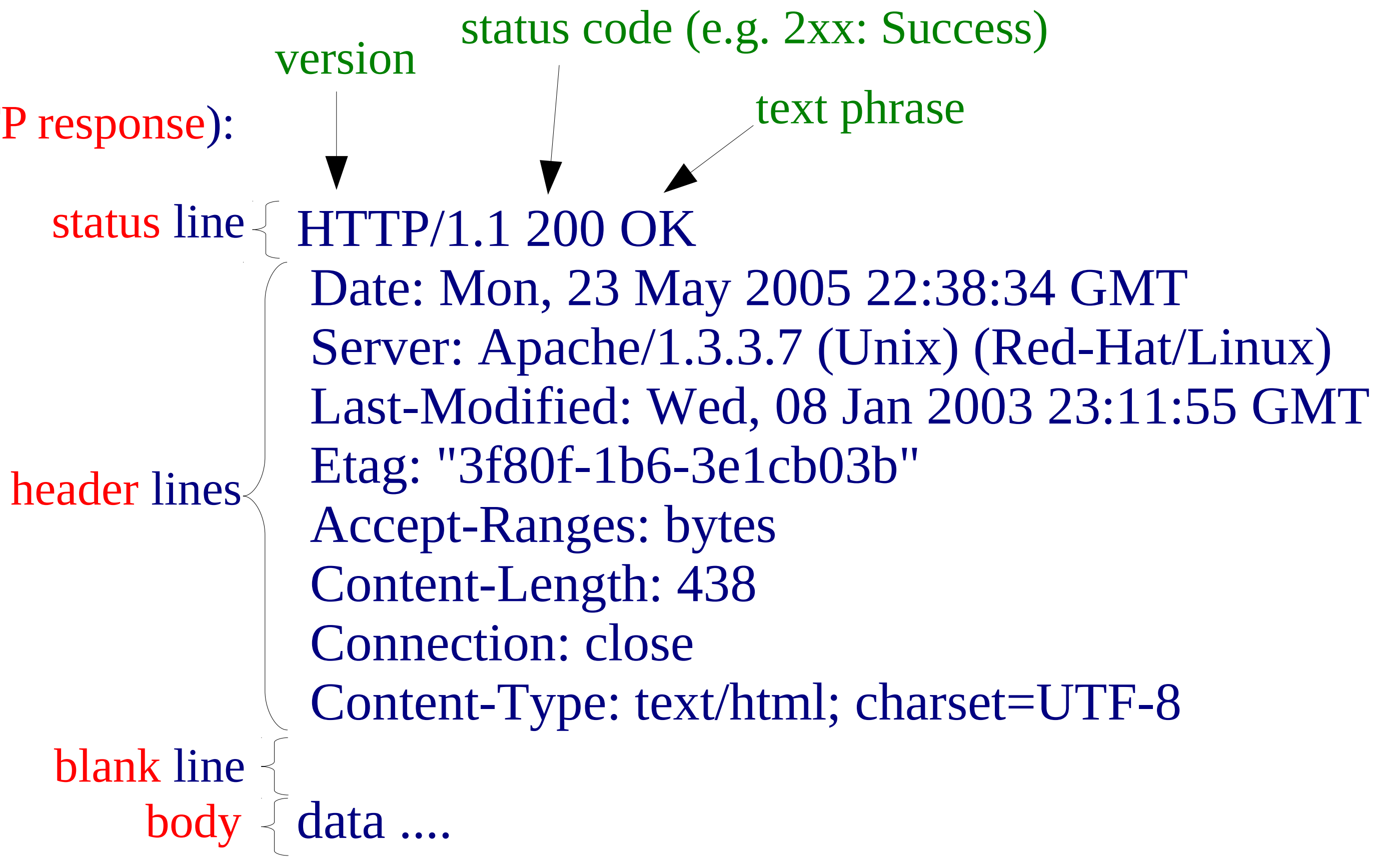
- POST uses MIME types: application/octet-stream, to send raw binary data, and application/x-www-form-urlencoded, to send name-value pairs. Example:

| | |
|---|---|
| request line | POST /login.jsp HTTP/1.1 |
| header lines | Host: www.mysite.com<br>User-Agent: Mozilla/4.0<br>Content-Length: 27<br>Content-Type: application/x-www-form-urlencoded |
| blank line | |
| body | userid=llorenc&password=mypassword |

# Unit 2. Network applications

## Web – HTTP Messages, RFC2616

- Server (HTTP response):



version

status code (e.g. 2xx: Success)

text phrase

status line

header lines

blank line

body

```
HTTP/1.1 200 OK
 Date: Mon, 23 May 2005 22:38:34 GMT
 Server: Apache/1.3.3.7 (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

data ....
```

# Unit 2. Network applications

## Web – Persistent/non Persistent connections

- Non persistent: The server close the TCP connection after every object. E.g, for an html page with 10 jpeg images, 11 TCP connections are sequentially opened.
- Persistent: The server maintains the TCP connection opened until an inactivity time. All 11 objects would be sent over the same TCP connection.
- Persistent connections with pipelining: The client issues new requests has soon as it encounter new references, even if the objects have been not completely downloaded.

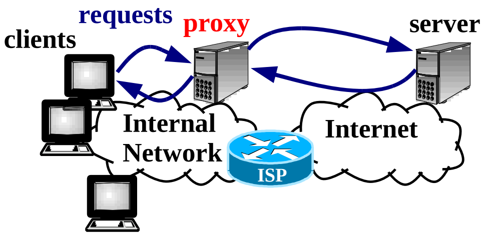
# Unit 2. Network applications

## Web – Caching and Proxies

- Caching: The client stores downloaded pages in a local cache. Conditional GET requests are used to download pages if necessary. It can use the Date and/or Etag:

```
GET /index.html HTTP/1.1
Host: www.example.com
If-Modified-Since: October 21, 2002 4:57 PM
If-None-Match: "686897696a7c876b7e"
```

- Proxy server: Acts as an intermediary for requests from clients.
    - Advantages:
        - Security (the proxy may reject the access to unauthorized servers)
        - Logs
        - Caching
        - Save public IP addresses (only the proxy may have access to the Internet)
        - ...



Source: wikipedia

# Unit 2. Network applications

## Web – web based applications

- Components:
  - Presentation: A web browser (client side).
  - Engine generating “on the fly” HTML pages (server side).
    - Languages:
      - Java.
      - Hypertext Preprocessor (PHP): Embedded program language and HTML code (http://www.php.net).
      - Other: ASP, CGI, ColdFusion, Perl, Python...
  - Storage: a database (e.g. mysql).

- Benefits:
  - Fast to deploy and upgrade (only server side).
  - Only a compatible browser is required at the client side.
  - Provide cross-platform compatibility (i.e., Windows, Mac, Linux, etc.)

Source: wikipedia

# Unit 2. Network applications

## Outline

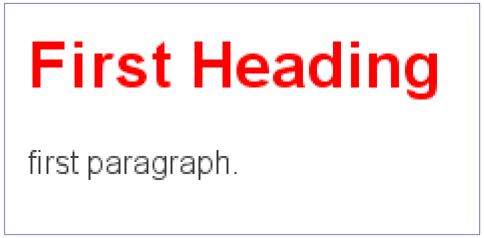
- Client-Server Paradigm
- DNS
- Email
- Web
- **HTML & XML**

# Unit 2. Network applications

## HTML & XML – Hyper-Text Markup Language, HTML

- In 1986 ISO standardized the Standard Generalized Markup Language (SGML). SGML introduced the <> syntax, and has been used in large documentation projects.
- Tim Berners-Lee defined HTML in 1989 inspired in SGML. HTML design mail goal was displaying formated text documents with hyperlinks (including links to other documents) in web browsers.
- Based on tags e.g. <head> data </head>
- Example:

```
<html>
<head>
 <title>Basic html document</title>
</head>
<body>
  <h1><font color="red">First Heading</font></h1>
  <p>first paragraph.</p>
</body>
</html>
```



# First Heading

first paragraph.

Terminology:

•element

•attribute

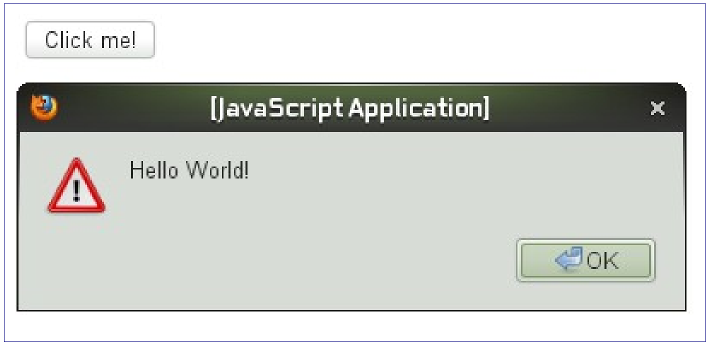
•text

# Unit 2. Network applications

## HTML & XML – Hyper-Text Markup Language, HTML

- HTML features (1):
  - Forms: The document accept user inputs that are sent to the server
  - Scripting: Allow adding programs. The program executes on the client's machine when the document loads, or at some other time such as when a link is activated.
  - javascript example:

```
<html>
<head>
<script type="text/javascript">
 function displaymessage() {
   alert("Hello World!");
 }
</script>
</head>
<body>
 <form>
  <input type="button"
   value="Click me!" onclick="displaymessage()" />
 </form>
</body>
</html>
```

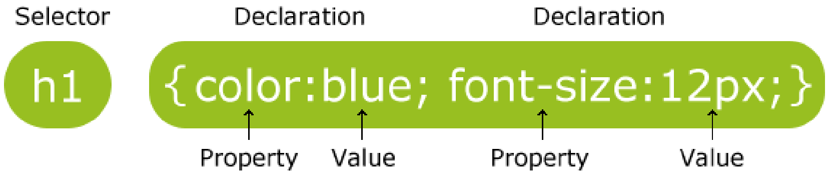


Hello World!

# Unit 2. Network applications

## HTML & XML – Hyper-Text Markup Language, HTML

- HTML features (2):
  - Cascading Style Sheets, CSS: Allows describing the *physical layout* in a separate document. E.g. thousand of HTML pages can use the same CSS. If the style must be changed, only the CSS need to be updated.
  - CSS Syntax

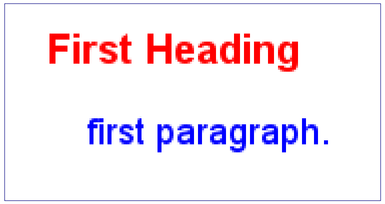


Source: http://www.w3schools.com/xml/

- CSS example
  - Content of the file "mystyle.css":

```
h1 {color:red; font-size:20px;}
p {margin-left:20px; color:blue; font-size:18px;}
```

```
<html>
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css" />
</head>
<body>
  <h1>First Heading</h1>
  <p>first paragraph.</p>
</body>
</html>
```



**First Heading**

first paragraph.

# Unit 2. Network applications

## HTML & XML – Extensible Markup Language, XML

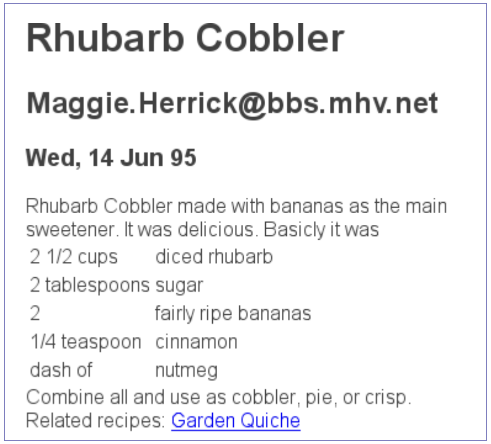
- History and Motivation
  - Due to tremendous success of web, World Wide Web Consortium (W3C) was created in 1994 to produce web standards.
  - Web evolution has increasingly involved towards the exchange of structured information, making HTML inadequate for many web projects.
  - XML is being developed in W3C to cope with transport and store of structured information. In 1998  XML 1.0 was the first W3C recommendation.
  - XML is not a replacement of HTML, but a framework for defining markup languages.
  - XML does not do anything: Someone must write an application (possibly a web application) to send, receive or display it.

# Unit 2. Network applications

## HTML & XML – Extensible Markup Language, XML

- Limitations of HTML
  - Consider a web site publishing recipes. A recipe could be as:

```
<h1>Rhubarb Cobbler</h1>
<h2>Maggie.Herrick@bbs.mhv.net</h2>
<h3>Wed, 14 Jun 95</h3>
Rhubarb Cobbler made with bananas as the main sweetener.
It was delicious.  Basicly it was
  <table>
  <tr><td> 2 1/2 cups <td> diced rhubarb
  <tr><td> 2 tablespoons <td> sugar
  <tr><td> 2 <td> fairly ripe bananas
  <tr><td> 1/4 teaspoon <td> cinnamon
  <tr><td> dash of <td> nutmeg
  </table>
Combine all and use as cobbler, pie, or crisp.
Related recipes: <a href="#GardenQuiche">Garden Quiche</a>
```



**Rhubarb Cobbler**

**Maggie.Herrick@bbs.mhv.net**

**Wed, 14 Jun 95**

Rhubarb Cobbler made with bananas as the main sweetener. It was delicious. Basicly it was

| | |
|---|---|
| 2 1/2 cups | diced rhubarb |
| 2 tablespoons | sugar |
| 2 | fairly ripe bananas |
| 1/4 teaspoon | cinnamon |
| dash of | nutmeg |

Combine all and use as cobbler, pie, or crisp.
Related recipes: Garden Quiche

Source: http://www.brics.dk/~amoeller/XML/xml/htmlvsxml.html

- Problems:
  - How to check that a recipe is introduced correctly? (ingredients amounts...)
  - How to identify the fields of the recipe? (author, ingredients...)
  - What if we want to display the fields in a different order?
  - …
- We need to define the semantics (meaning) of tags, and the syntax (what tags, and how can they be used).

# Unit 2. Network applications

## HTML & XML – Extensible Markup Language, XML

- Solution: XMLization

```
<recipe id="117" category="dessert">
  <title>Rhubarb Cobbler</title>
  <author><email>Maggie.Herrick@bbs.mhv.net</email></author>
  <date>Wed, 14 Jun 95</date>
  <description>
    Rhubarb Cobbler made with bananas as the main sweetener.
    It was delicious.
  </description>
  <ingredients>
    <item><amount>2 1/2 cups</amount><type>diced rhubarb</type></item>
    <item><amount>2 tablespoons</amount><type>sugar</type></item>
    <item><amount>2</amount><type>fairly ripe bananas</type></item>
    <item><amount>1/4 teaspoon</amount><type>cinnamon</type></item>
    <item><amount>dash of</amount><type>nutmeg</type></item>
  </ingredients>
  <preparation>
    Combine all and use as cobbler, pie, or crisp.
  </preparation>
  <related url="#GardenQuiche">Garden Quiche</related>
</recipe>
```

Source: http://www.brics.dk/~amoeller/XML/xml/htmlvsxml.html

Terminology:
•element
•attribute
•text

- XML is designed to tailor-made markup languages.
- Examples:
  - Community Network Markup Language (CNML) to describe guifi.net: http://guifi.net/es/guifi/cnml/3671
  - gnome GConf configuration system: http://projects.gnome.org/gconf
  - ...

# Unit 2. Network applications

## HTML & XML – Extensible Markup Language, XML

- A well-formed XML document satisfies a list of syntax rules provided in the specification. It is more rigid than HTML (e.g. all tags must be closed: <tag> </tag> or <tag attribute1=.. />).
- XML namespaces
  - Allow differentiating elements names defined by different developers.
  - The namespace is defined by the xmlns attribute in the start tag of an element.
  - URL are often used as an easy way to define “unique” namespaces.

```
<widget xmlns="http://www.widget.org"
        xmlns:xhtml="http://www.w3.org/TR/xhtml1"
        type="gadget">
  <head size="medium"/>
  <big><subwidget ref="gizmo"/></big>
  <info>
    <xhtml:head>
      <xhtml:title>Description of gadget</xhtml:title>
    </xhtml:head>
    <xhtml:body>
      <xhtml:h1>Gadget</xhtml:h1>
      A gadget contains a big gizmo
    </xhtml:body>
  </info>
</widget>
```

default namespace.

namespace with prefix xhtml. The prefix acts as a shortname for the namespace.

Source: http://www.brics.dk/~amoeller/XML/xml/htmlvsxml.html

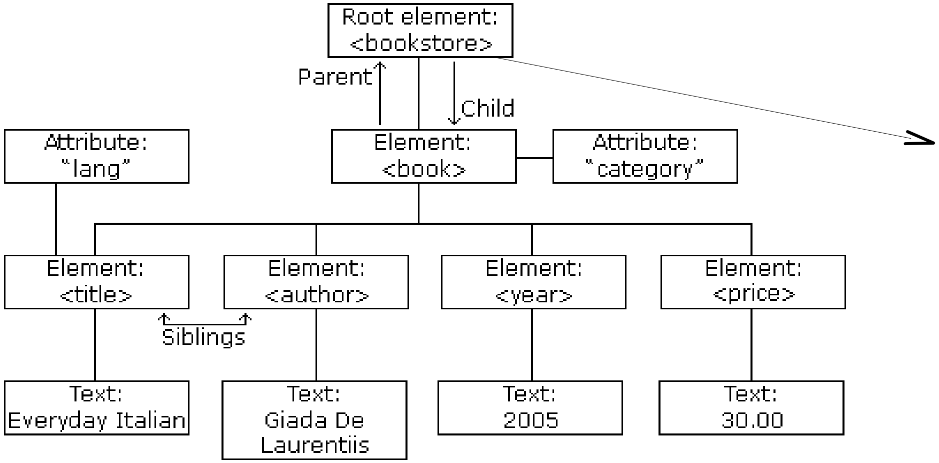
# Unit 2. Network applications

## HTML & XML – Extensible Markup Language, XML

- XML documents have a tree structure

```
<bookstore>
  <book category="COOKING">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="CHILDREN">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
...
</bookstore>
```

Terminology:
- element
- attribute
- text



… other books

Source: http://www.w3schools.com/xml/

# Unit 2. Network applications

## HTML & XML – Extensible Markup Language, XML

- XPath: Navigating XML documents
  - Syntax for selecting parts of an XML document.
  - Used e.g. by the XML transformation language XSLT (explained later).

  - Example

```
<bookstore>
  <book category="COOKING">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="CHILDREN">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
...
</bookstore>
```

Source: http://www.w3schools.com/xml/

Example of a XPath expression: title of the first book of the bookstore: /bookstore/book[1]/title

# Unit 2. Network applications

## HTML & XML – Extensible Markup Language, XML

- Validation of XML documents
- A "Valid" XML document conforms to the syntax of an XML schema.
- The syntax defines what tags and how can be used.

- Most used schema languages:
  - Document Type Definition, DTD:
    - First XML schema language.
    - Do not follows XML syntax.
  - XML Schema Definition, XSD:
    - Follows XML syntax (allows namespaces).
    - Can express more complex rules than DTD.

# Unit 2. Network applications

## HTML & XML – Extensible Markup Language, XML

- Document Type Definition, DTD
  - Content of the file “note.dtd”:

```
<!ELEMENT note (to,from,heading,body)>
<!ATTLIST note date CDATA #IMPLIED>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
```

Source: http://www.w3schools.com/xml/

element “note” contains the elements to,from,heading,body.

element “note” has attribute “date” (#IMPLIED means “not required”).

element “to” contains character data (#PCDATA).

  - Reference to the DTD defined in “note.dtd”:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE note SYSTEM "note.dtd">
<note date="10/01/2008">
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

Declaration starting an XML document.

DTD schema defined in location “note.dtd”.

  - Validation example with xmllint (http://xmlsoft.org/):

```
linux ~/> xmllint --dtdvalid note.dtd exemple-dtd.xml
exemple-dtd.xml validates
```

# Unit 2. Network applications

## HTML & XML – Extensible Markup Language, XML

- XML Schema Definition, XSD
  - Content of the file “note.xsd”:

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="note">
    <xs:complexType>
      <xs:sequence>
            <xs:element name="to" type="xs:string"/>
            <xs:element name="from" type="xs:string"/>
            <xs:element name="heading" type="xs:string"/>
            <xs:element name="body" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

http://www.w3schools.com/xml/

namespace where the schema is defined, the namespace should be prefixed xs.

root element

complexType: contains other elements

sequence: child elements must appear in the same order

- Reference to the XSD defined in “note.xsd”:

```
<?xml version="1.0"?>
<note xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="note.xsd">
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

XSD schema defined in location “note.xsd”

- An XML file using XSD can also be validated with xmllint.

# Unit 2. Network applications

## HTML & XML – Extensible Markup Language, XML

- Extensible Stylesheet Language, XSL
  - Extend the CSS idea of HTML.
  - The main component is the XSL Transformations, XSLT.
  - XSLT is a programming language for specifying transformations between XML and a particular target language (e.g. HTML).
  - All major browsers support XML/XSLT: mozilla, explorer, google chrome...

  - An XSL style sheet consists of one or more rules called templates.
  - Templates are applied when a specified node is matched.

# Unit 2. Network applications

## HTML & XML – Extensible Markup Language, XML

- XSL Transformations, XSLT

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl" href="cdcatalog.xsl"?>
<catalog>
  <cd>
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
  </cd>
…
</catalog>
```

reference an XSLT “cdcatalog.xsl” in an XML document

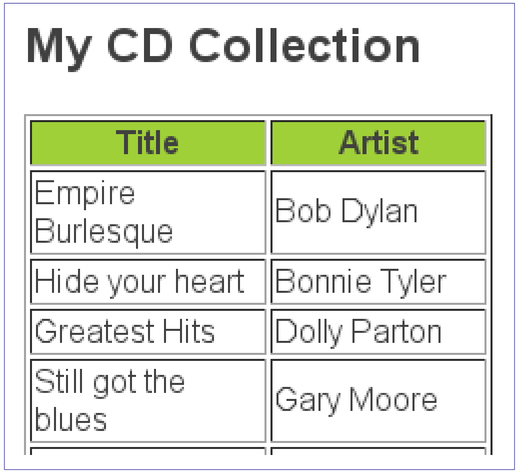
- Content of the file “cdcatalog.xsl”:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
  <html><body>
  <h2>My CD Collection</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
      <th>Title</th>
      <th>Artist</th>
    </tr>
    <xsl:for-each select="catalog/cd">
    <tr>
      <td><xsl:value-of select="title"/></td>
      <td><xsl:value-of select="artist"/></td>
    </tr>
    </xsl:for-each>
  </table>
  </body></html>
</xsl:template>
</xsl:stylesheet>
```

defines a template. Attibute match specifies the nodes using XPath

select every element of a node-set

extract the value of an XML element



My CD Collection

| Title | Artist |
|---|---|
| Empire Burlesque | Bob Dylan |
| Hide your heart | Bonnie Tyler |
| Greatest Hits | Dolly Parton |
| Still got the blues | Gary Moore |

Source: http://www.w3schools.com/xml/