# Computer Networks - *Xarxes de Computadors*

## Outline

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

# Unit 5. Network applications

## Outline

- **DNS**
- Charsets
- Email
- Web
- HTML & XML

# Unit 2: IP Networks

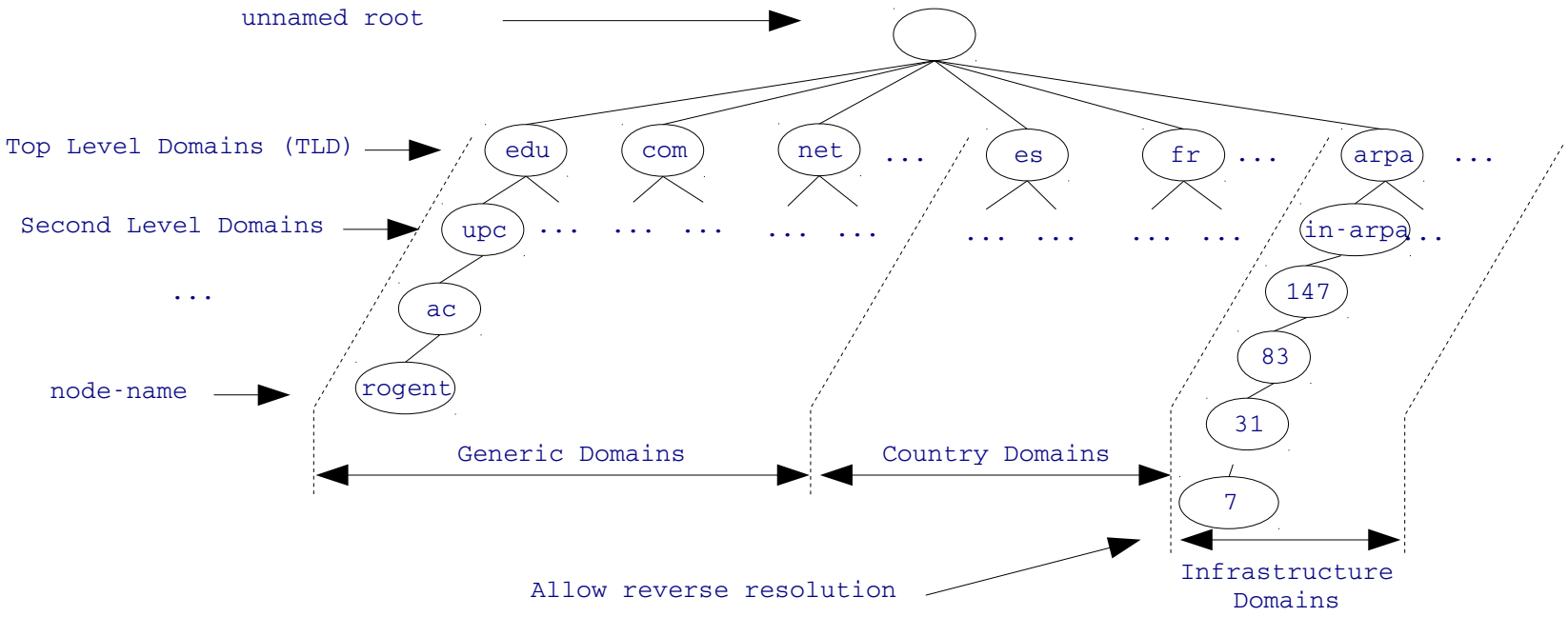
## Domain Name System DNS (RFC 1034, 1035)

- 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: IP Networks

## DNS – Domain Hierarchy

- DNS data base is organized in a tree:



# Unit 2: IP Networks

## 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.

# Unit 2: IP Networks

## 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: IP Networks

## 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: IP Networks

## 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
```

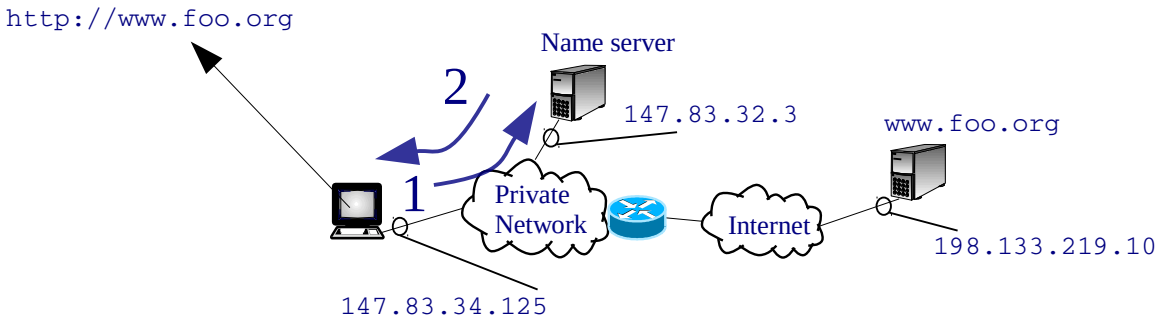
- Otherwise a *name server* is contacted using /etc/resolv.conf file:

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

# Unit 2: IP Networks

## 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)`

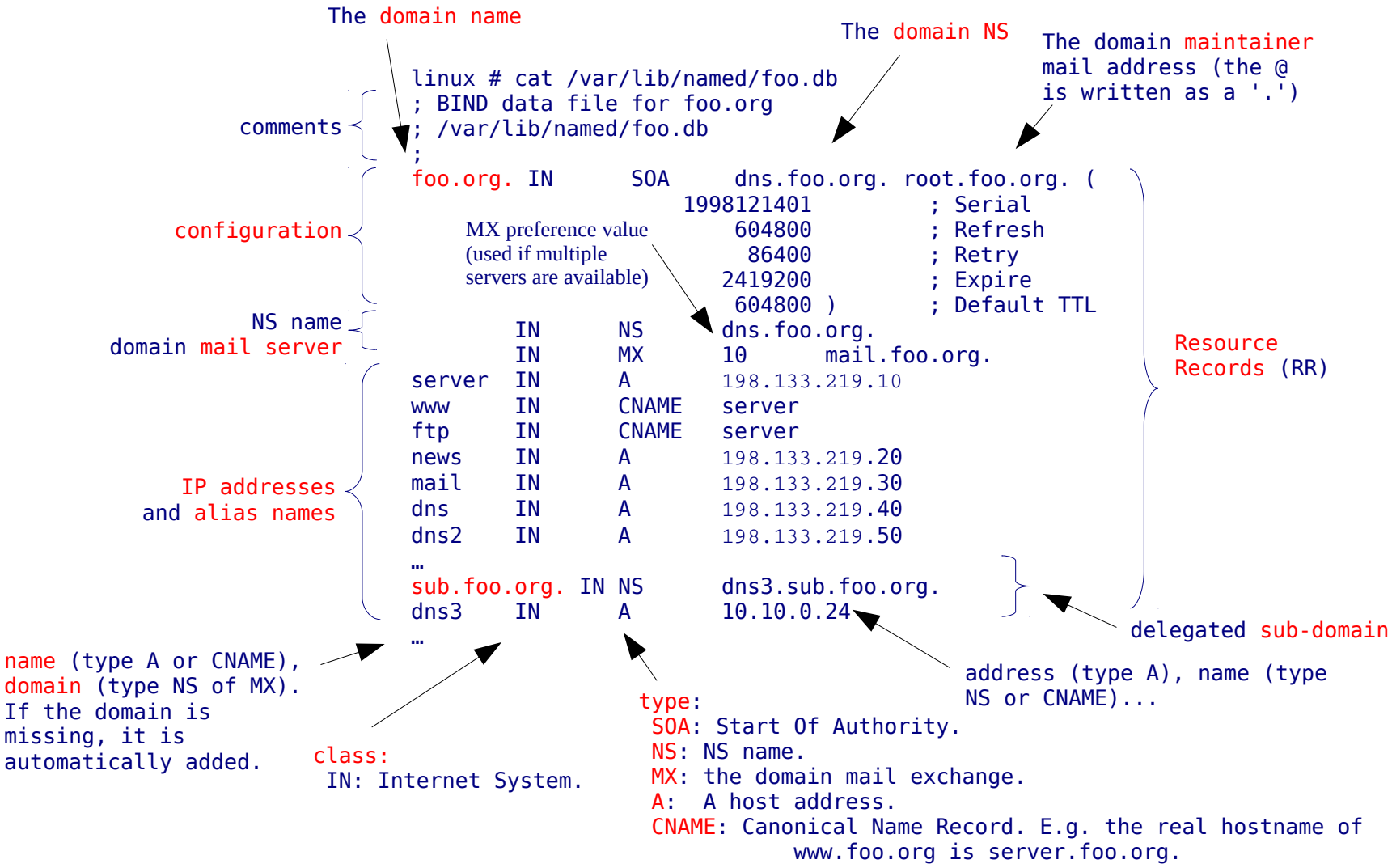
# Unit 2: IP Networks

## 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: IP Networks

## DNS – Unix example: zone file



# Unit 2: IP Networks

## 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

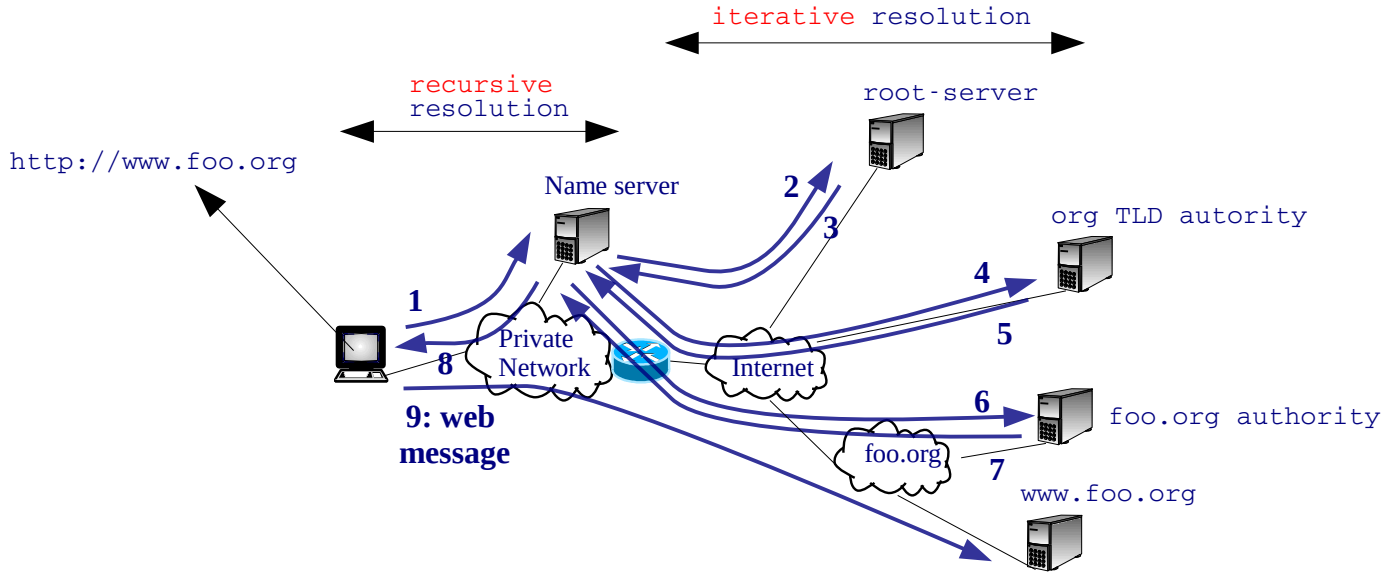
Resource Records (RR) pointing to root-servers

address of a name
NS name

# Unit 2: IP Networks

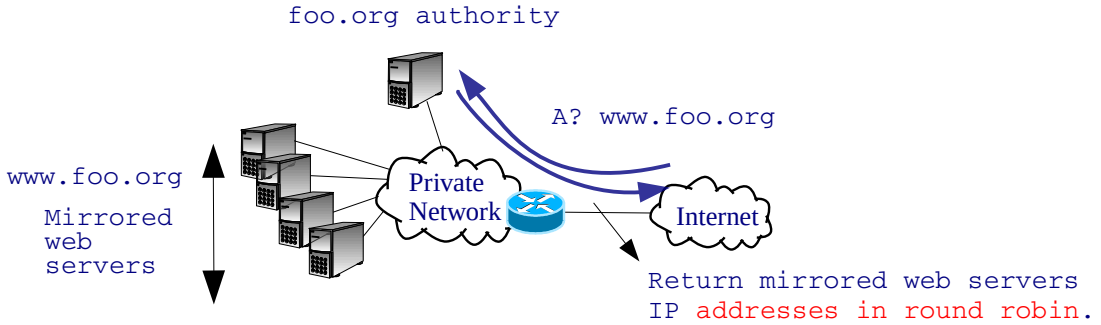
## DNS – Resolution

- 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: IP Networks

## DNS – Load balancing, example



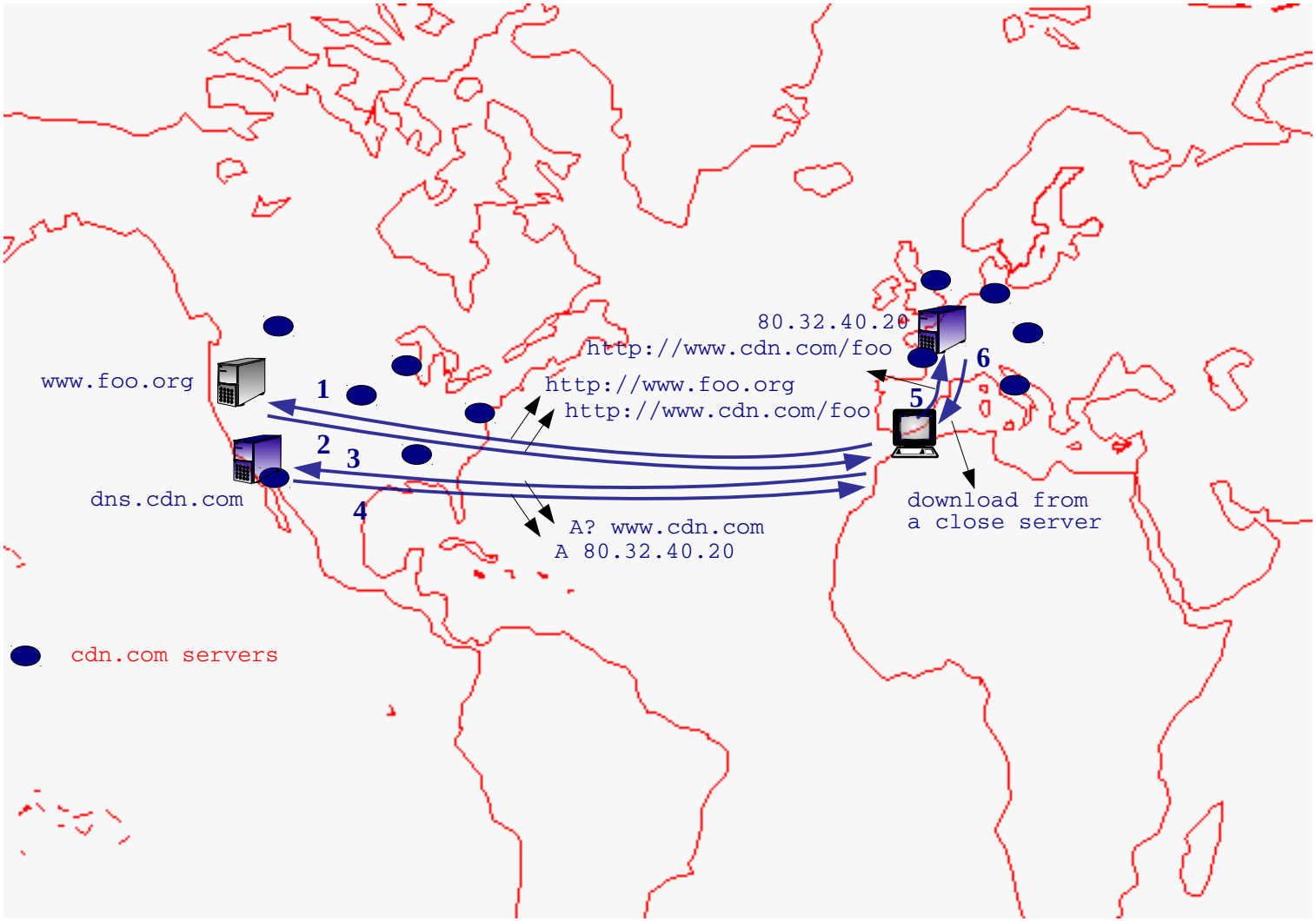

- Example using dig:

```
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: IP Networks

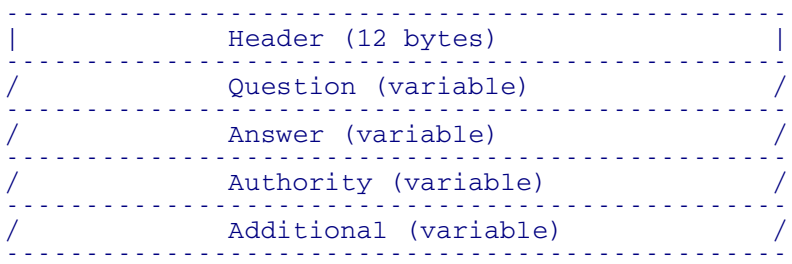
## DNS - Content Distribution Networks, example



# Unit 2: IP Networks

## DNS – Messages: Message Format

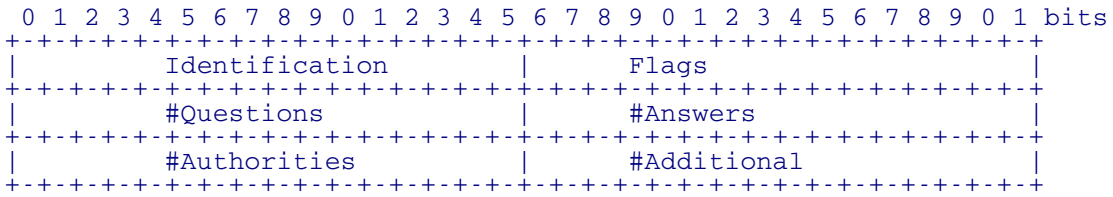
- All DNS messages have the same format:
  - Header: type of message.
  - Question: What is to be resolved.
  - Answer: Answer to question.
  - Authority: Domain authority names.
  - Additional: Typically, the authority name's addresses.



# Unit 2: IP Networks

## DNS – Messages: Header

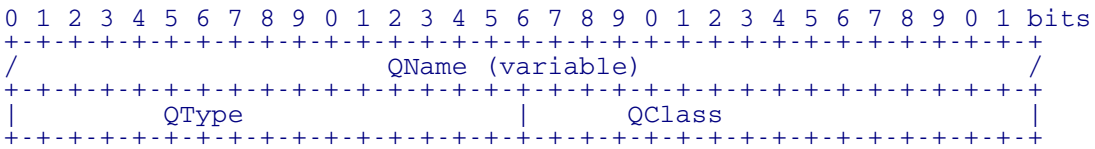
- Identification: 16 random bits used to match query/response
- Flags. Some of them:
  - Query-Response, QR: 0 for query, 1 for response.
  - Authoritative Answer, AA: When set, indicates an authoritative answer.
  - Recursion Desired, RD: When set, indicates that recursion is desired.
- The other fields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

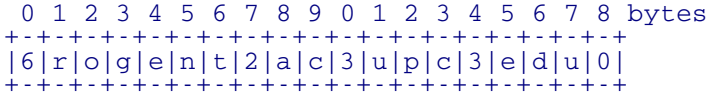


# Unit 2: IP Networks

## 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.



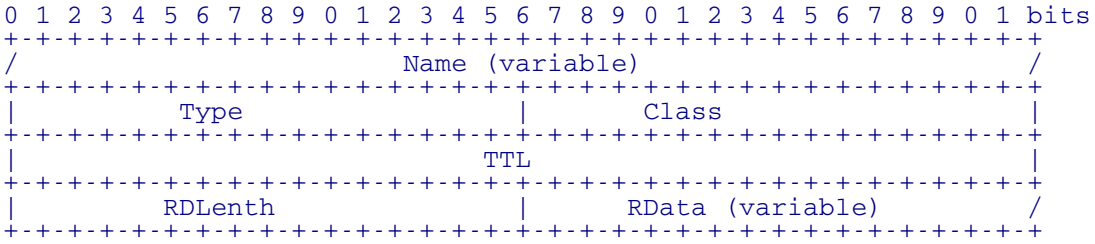




Codification example of `rogent.ac.upc.edu`

# Unit 2: IP Networks

## 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'.



# Unit 2: IP Networks

## 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 5. Network applications

## Outline

- DNS
- **Charsets**
- Email
- Web
- HTML & XML

# Languages, cultures, alphabets

7400 million people (2016)

22% speak Chinese, 11% English, 7% Spanish, 0,1% Catalan

Apart from languages, there are cultures and alphabets

- Language with several cultures: es_ES, es_CO ("locale")
- Alphabet shared by several languages (e.g. català & français)

Culture:

- Messages, character sets, transliteration, ordering, search in strings, hours and dates, numbers and currency, pronunciation, …

Interaction between agents in different languages and cultures: alphabets and character sets

# Languages, cultures, alphabets

Internacionalization (i18n), Localization (l10n)

Alphabets

- "base": ascii
- National: e.g.: latin-1 (includes ascii), kanji
- International: e.g.: unicode (includes latin-1 and “all” languages)

Expression or language negotiation (in HTTP):

**Accept-Language**: es, ca, en-gb, en
**Accept-Charset**: iso-8859-15, unicode-9-0
. . .



English is the default …



**Content-Language**: ca
**Content-Type**: text/html; charset=utf-8
. . .

# Character sets

Characters are encoded following several conventions:

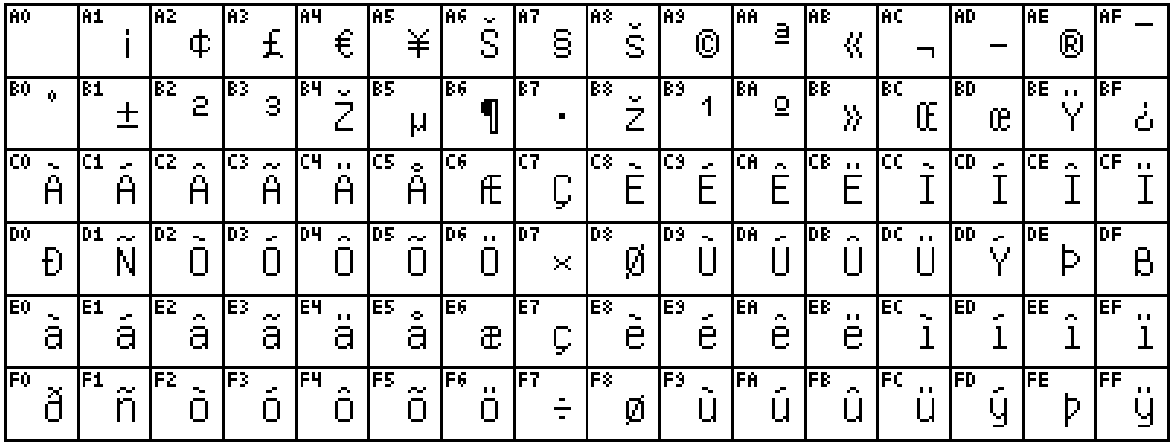
- **repertoire**: a set of characters (name and representation (glyph))
- **code**: correspondence between repertoire and natural numbers.
- **encoding**: method (algorithm) to convert code numbers into a sequence of octets (> 256 characters)
- US-ASCII: 95 characters + control=128: 7 bits (1 octet sent)

USASCII code chart

| b7 b6 b5 → | 0 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1 0 0 | 1 0 1 | 1 1 0 | 1 1 1 |
|---|---|---|---|---|---|---|---|---|
| b4 b3 b2 b1 ↓ Row | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 0 0 0 0 | NUL | DLE | SP | 0 | @ | P | ` | p |
| 0 0 0 1 1 | SOH | DC1 | ! | 1 | A | Q | a | q |
| 0 0 1 0 2 | STX | DC2 | " | 2 | B | R | b | r |
| 0 0 1 1 3 | ETX | DC3 | # | 3 | C | S | c | s |
| 0 1 0 0 4 | EOT | DC4 | $ | 4 | D | T | d | t |
| 0 1 0 1 5 | ENQ | NAK | % | 5 | E | U | e | u |
| 0 1 1 0 6 | ACK | SYN | & | 6 | F | V | f | v |
| 0 1 1 1 7 | BEL | ETB | ' | 7 | G | W | g | w |
| 1 0 0 0 8 | BS | CAN | ( | 8 | H | X | h | x |
| 1 0 0 1 9 | HT | EM | ) | 9 | I | Y | i | y |
| 1 0 1 0 10 | LF | SUB | * | : | J | Z | j | z |
| 1 0 1 1 11 | VT | ESC | + | ; | K | [ | k | { |
| 1 1 0 0 12 | FF | FS | , | < | L | \ | l | \| |
| 1 1 0 1 13 | CR | GS | - | = | M | ] | m | } |
| 1 1 1 0 14 | SO | RS | . | > | N | ^ | n | ~ |
| 1 1 1 1 15 | SI | US | / | ? | O | _ | o | DEL |

# ISO 8859

- ISO 8859-1 (ISO Latin 1): 190 + control = 256: 1 octet Western European, default for HTTP



- More variants

ISO 8859-15 extends -1 + Ÿ, €

ISO 8859-2 (Central European)

ISO 8859-4 (North European)

ISO 8859-5 (Cyrillic)

ISO 8859-6 (Arabic) — Most common Arabic glyphs

ISO 8859-7 (Greek)

ISO 8859-8 (Hebrew) — modern Hebrew.

ISO 8859-9 (Turkish, Kurdish)

ISO 8859-11 (Thai) — Contains most glyphs needed

# Universal Coded Character Set
# Unicode

All characters from all written languages + math + emoticons +
+=Universal Character set (ucs)

Encoding: UCS-4 bytes (fixed length)

Proportional spacing, language independent

Unicode consortium: synchronized with ISO,

- Unicode 9.0.0 (7/2016): 128,172 symbols
- U+hex code: U+0020 = ' '

Character Encodings: Universal Transformation Format (UTF)

- Difficulty or impossibility to transport 8 o 16 bits data in Internet protocols:
- UTF-7, **UTF-8**, UTF-16, UTF-32 (variable length)

http://www.unicode.org

# Variable length encodings

- UTF-8 (8 bits) (rfc2044)

```
Content-Type: text/plain; charset=UTF-8
Content-Transfer-Encoding: 8bit
CatalÃ , FranÃ§ais, TÃ¤mÃ¤ on testi.
```

- UTF-7 (7 bits) (for smtp …)

```
Content-Type: text/plain; charset=UTF-7
Content-Transfer-Encoding: 7bit
Catal+-AOA-, Fran+-AOc-ais, T+-AOQ-m+-AOQ- on testi.
```

# Unit 5. Network applications

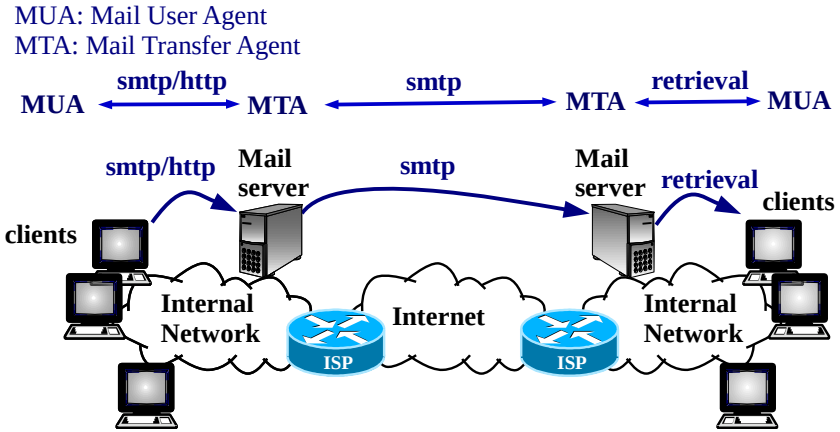
## Outline

- DNS
- Charsets
- **Email**
- Web
- HTML & XML

# Unit 5. Network applications

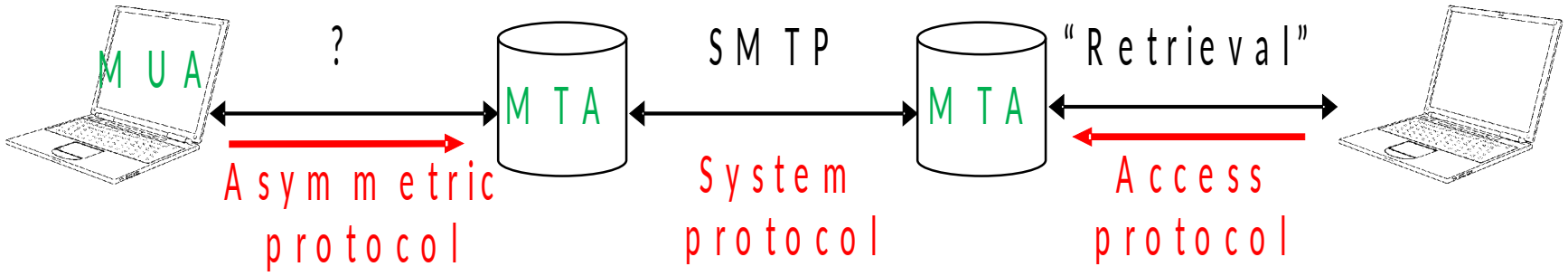
## Email

- 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 5. Network applications

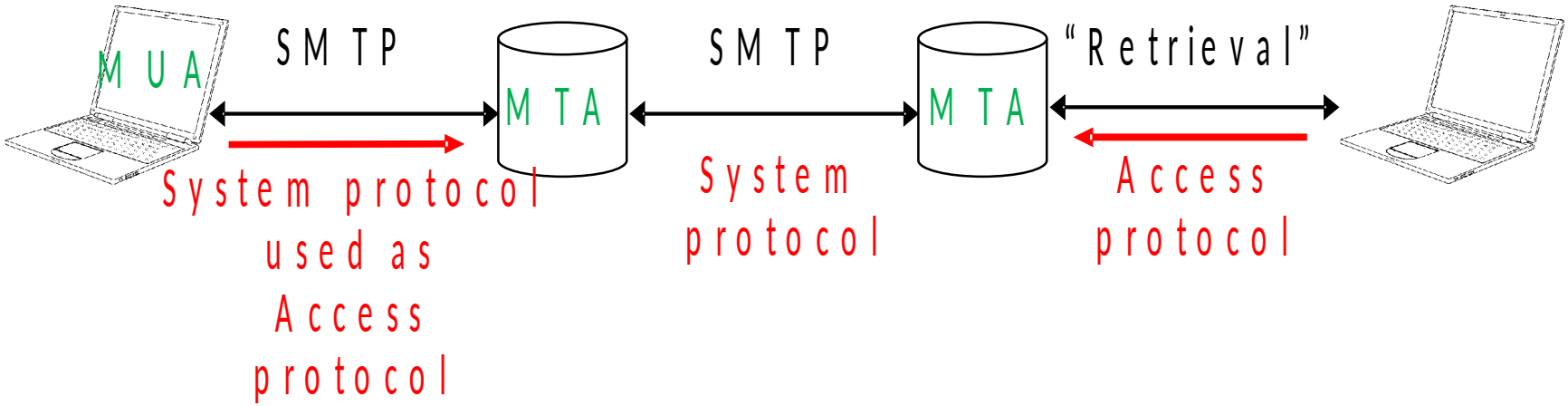
## Email - Architecture




- MUA: Mail User Agent
- MTA: Mail Transfer Agent
- SMTP: Simple Mail Transfer Protocol

# Unit 5. Network applications

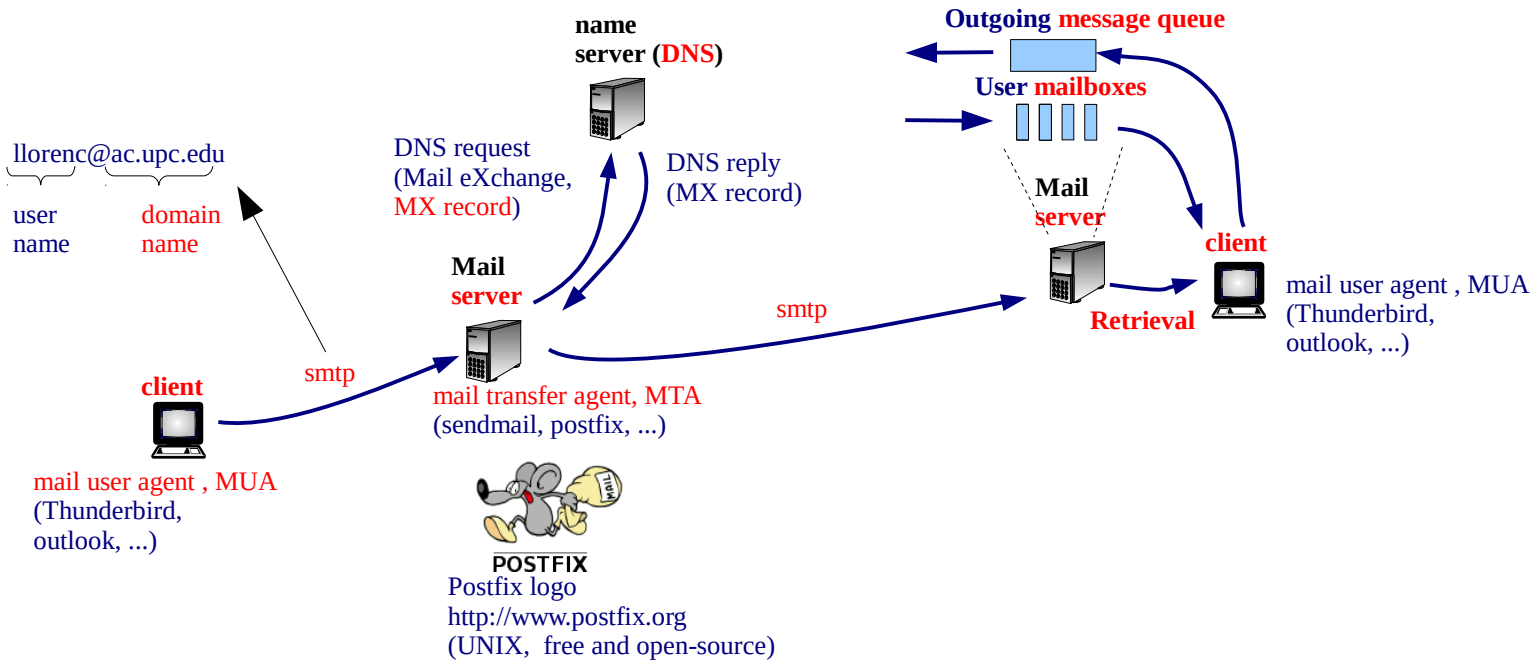
## Email - Protocols




- "Retrieval" protocols (mailbox access):
  - POP3 (Post Office Protocol)
  - IMAP (Internet Message Access Protocol)
- SMTP: Simple Mail Transfer Protocol

# Unit 5. Network applications

## Email - SMTP processing model



# Unit 5. Network applications

## Email - SMTP protocol (RFC-821, last update RFC-5321)

- 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.

SMTP transaction

```
CLIENT  linux ~> telnet relay.upc.edu 25
        Trying 147.83.2.12...
        Connected to relay.upc.edu.
        Escape character is '^]'.
SERVER  220 dash.upc.es ESMTP Sendmail 8.14.1/8.13.1; Fri, 4 Feb 2011 14:57:15 +0100
COMMANDS HELO linux.ac.upc.edu
        250 dash.upc.es Hello linux.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: <albert@ac.upc.edu>
        250 2.1.5 <albert@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.
        linux ~>
```

# Multipurpose Internet Mail Extensions: MIME

- Used in mail, web, etc
- Specification for “Transport” of composite multimedia objects
  - Transport type information (receiver can automatically present)
  - Encoding to enable/facilitate the transfer
- The internal format becomes invisible to users
- Include one or more objects, text in diverse alphabets, large objects (fragments, refs), alternatives, etc.

# MIME: examples

```
From: Nathaniel Borenstein <nsb@thumper.bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: Plain old email

This is a plain old email message.
It contains ASCII text, nothing more.
```

---

```
From: Nathaniel Borenstein <nsb@thumper.bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: Plain text mail
Content-type: text/plain; charset=us-ascii

This is plain text mail.
```

---

```
...Subject: French mail
Content-type: text/plain; charset=iso-8859-1
Content-transfer-encoding: quoted-printable

Le courrier =E9lectronique =E0 la fran=E7aise ...
```

---

```
...Content-type: image/gif
Content-Transfer-Encoding: base64

R0lGODdhSgGgAfUAAENDQ01NTTw8PEVF...
```

# MIME: example multipart

```
From: Nathaniel Borenstein <nsb@bellcore.com>
To: Ned Freed <ned@innosoft.com>
Subject: A multipart example
Content-Type: multipart/mixed; boundary=CUT_HERE

--CUT_HERE
Content-type: text/plain

Hey, Ned, look at this neat picture:
--CUT_HERE
Content-type: image/gif
Content-Transfer-Encoding: base64

5WVlZ6enqqqqr....
--CUT_HERE
Content-type: text/plain

Wasn't that neat?
--CUT_HERE--
```

# MIME: content type

- Text: …
  - Attribute: charset=iso-8859-1
  - text/plain (simple text), text/html ...
- Image: image/gif, image/jpeg, image/png ...
- Audio: sound, voice, music …
- Application: application specific content
  - Application/octet-stream: data without any associated application
  - Application/organization-product
- Multipart: a set of objects
  - Mixed: a combination of several objects
  - Alternative: an object in several formats to select one (text/html/rtf)
  - Parallel: several objs for simultaneous presentation (e.g. audio+video)
  - Digest: collection of messages
  - Related: set of objects part of a single object (web page)
- Message:
  - RFC822: a complete message (eg. resent message)
  - Partial: a fragment …
  - External-Body: a reference to an external object

Registration scheme Type/subtype: mantained by IANA

# MIME: transfer encoding

Ways to encode content: (to “get through” a 7 bit transport)

- Quoted-Printable:
  - The majority of text is 7 bits, transform some characters € → =E4
  - The result “almost" legible without decoding. Depends on table (charset)
- Base64:
  - 3 bytes (24 bits) <=> 4 ASCII (32 bits)
  - A-Za-z0-9+/=
  - '=' as padding, other are ignored (\r, \n, …)
- Binary: No encoding: any character and lines of any length
- 7Bit: No character encoding (all 7 bits) and lines of appropriate length
- 8Bit: No character encoding (8 bits) and lines of appropriate length
- In the heading:

```
MIME-Version: 1.0
Subject: =?iso-8859-1?Q?acentuaci=F3n=20t=EDpica?=
```

# Unit 5. Network applications

## Email - retrieval protocols

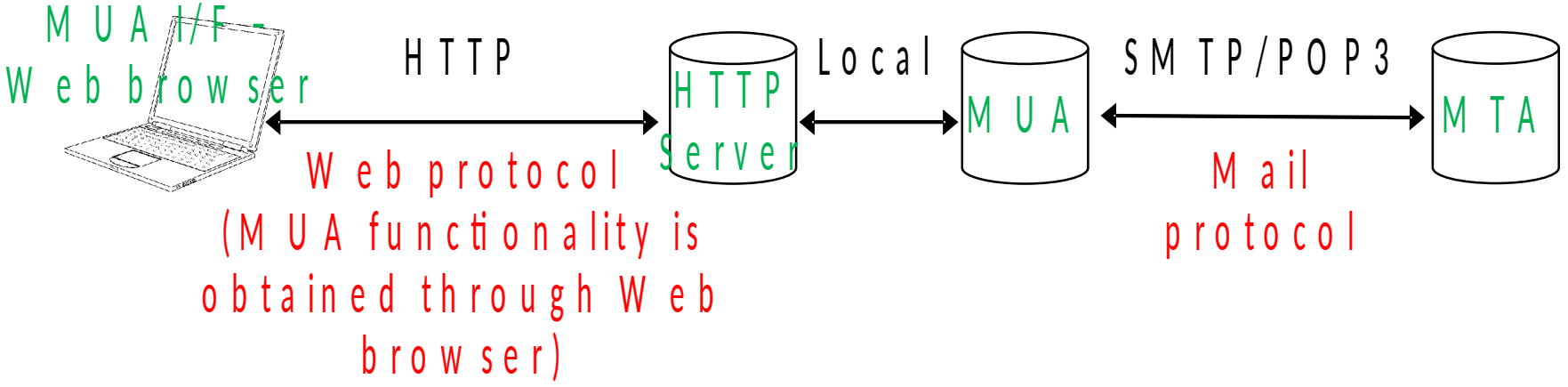
- Post Office Protocol (POP), RFC-1939:
  - POP server listens on well-known port 110
  - User normally deletes messages upon retrieval.

- 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 5. Network applications

## Email - Webmail




- Web front-end for mail services. The MUA is a web browser.
- Real protocol to access the services: HTTP (web).
- The HTTP server machine uses SMTP or POP3, as required.

# Unit 5. Network applications

## Outline

- DNS
- Charsets
- Email
- **Web**
- HTML & XML

# Unit 5. 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

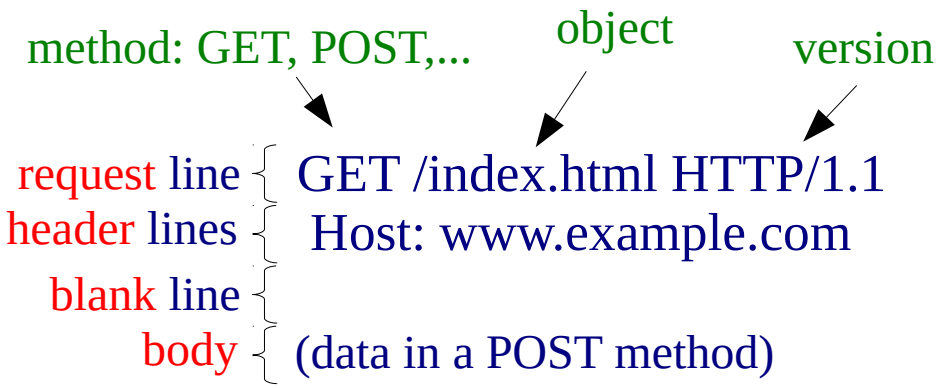
  **scheme://username:password@domain:port/path?query_string#fragment_id**

  - 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 5. Network applications

## Web – HTTP Messages, RFC2616

- Client (HTTP request):




- 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 5. 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
User-Agent: Mozilla/4.0
Content-Length: 27
Content-Type: application/x-www-form-urlencoded

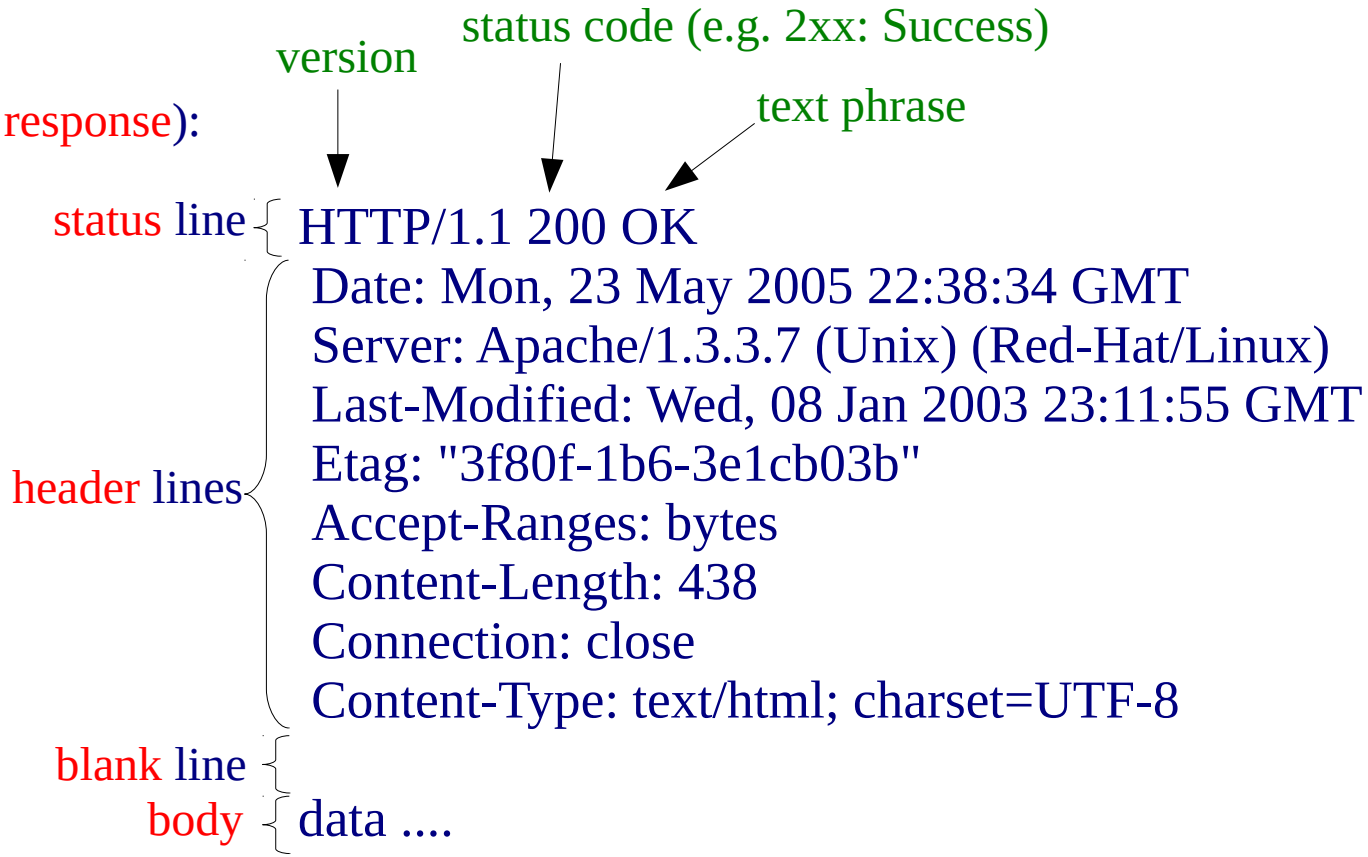
blank line

body: userid=llorenc&password=mypassword

# Unit 5. Network applications

## Web – HTTP Messages, RFC2616

- Server (HTTP response):



# Unit 5. Network applications

## Web – Persistent/non Persistent connections

- Non persistent (default in HTTP/1.0): 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 (default in HTTP/1.1) : 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 (supported only in HTTP/1.1): The client issues new requests as soon as it encounter new references, even if the objects have been not completely downloaded.

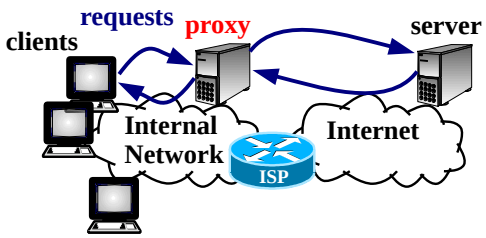
# Unit 5. Network applications

## 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)
    - ...



# Unit 5. 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.)

# Unit 5. Network applications

## Outline

- DNS
- Charsets
- Email
- Web
- **HTML & XML**

# Unit 5. Network applications

## HTML – 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 5. Network applications

## HTML – 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.

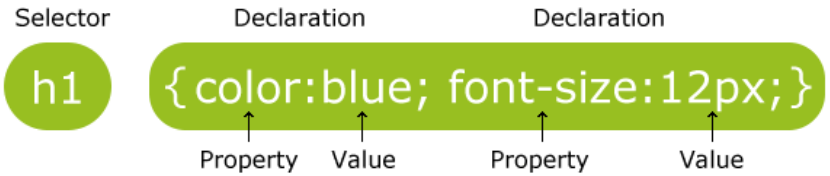
- javascript example:

```
<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>
```

# Unit 5. Network applications

## HTML – 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

## XML – eXtensible Markup Language

- Designed to transport and store data (HTML to *display* data).
- Users define their own tags to describe information structures

→ Process them automatically with applications.

- Tree structure.
- Elements, attributes & text.
- Example:

```
<book category="COOKING">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    ...
</book>
```

# Unit 2. Network applications

## XML – eXtensible Markup Language

- 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

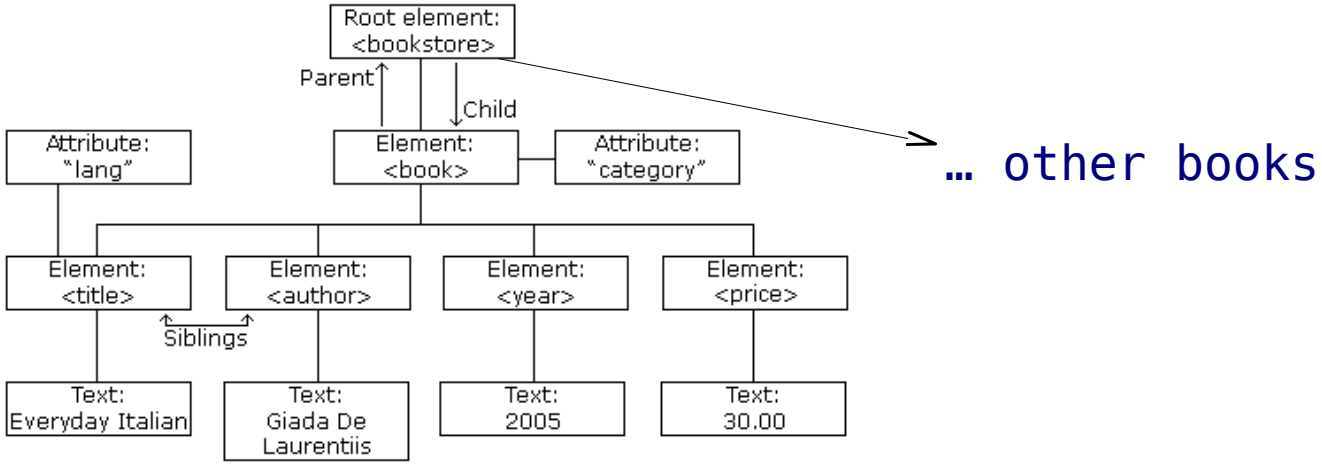
# Unit 2. Network applications

## XML – eXtensible Markup Language

- 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




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

# Unit 2. Network applications

## XML – eXtensible Markup Language

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

- Most known schema languages:
  - Document Type Definition, DTD:
    - First XML schema language (obsolete now).
    - Does not follow XML syntax.
  - XML Schema Definition, XSD:
    - Follows XML syntax (allows namespaces).
    - Can express more complex rules than DTD.