Internet Protocols

Robin Sharp

Informatics and Mathematical Modelling
Technical University of Denmark
Phone: (+45) 4525 3749
e-mail: robin@imm.dtu.dk

Internet Protocols

- Just to remind you:

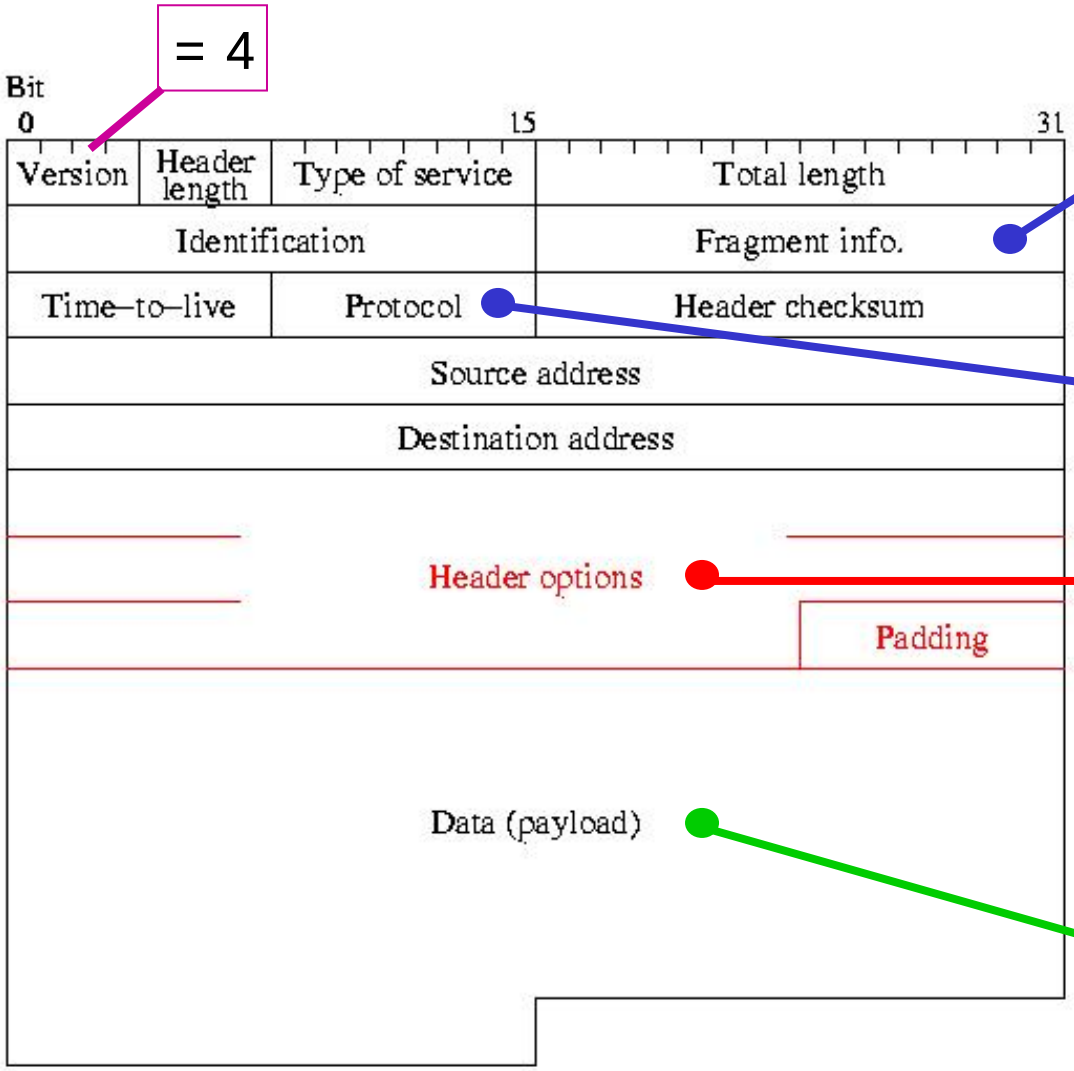
Application	Direct support to application processes FTP, SMTP, HTTP, POP, NNTP,...
Transport	End-to-end transfer of data TCP, UDP,...
Network	Transfer of data between arbitrary systems IP
Data Link	Transfer of data between directly connected systems
Physical	Physical signalling on the medium (wire or fibre)

TCP / IP

Internet Protocol (IP)

- Implements a connectionless-mode, full duplex, stream service for data transfer between arbitrary systems.
- Can offer point-to-point or multicast service.
- Available in two versions:
  - **Version 4 (IPv4),** described in Internet RFC791.
    - Identifies systems by 32-bit addresses.
  - **Version 6 (IPv6),** described in RFC1883 and RFC2373.
    - Identifies systems by 128-bit addresses.
    - Offers improved facilities for dealing with different traffic classes, security, etc.

IPv4 PDU



Is segmentation used?
Offset of this segment?

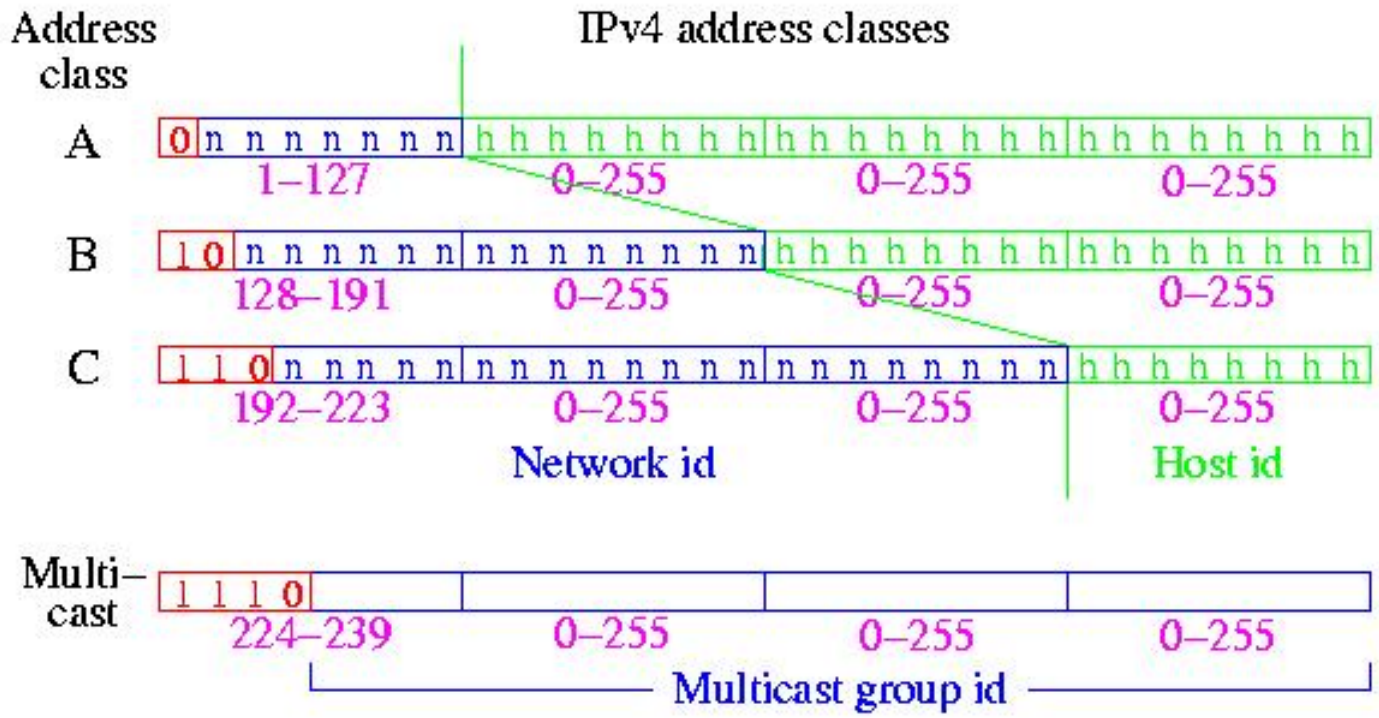
PDU carried as data?

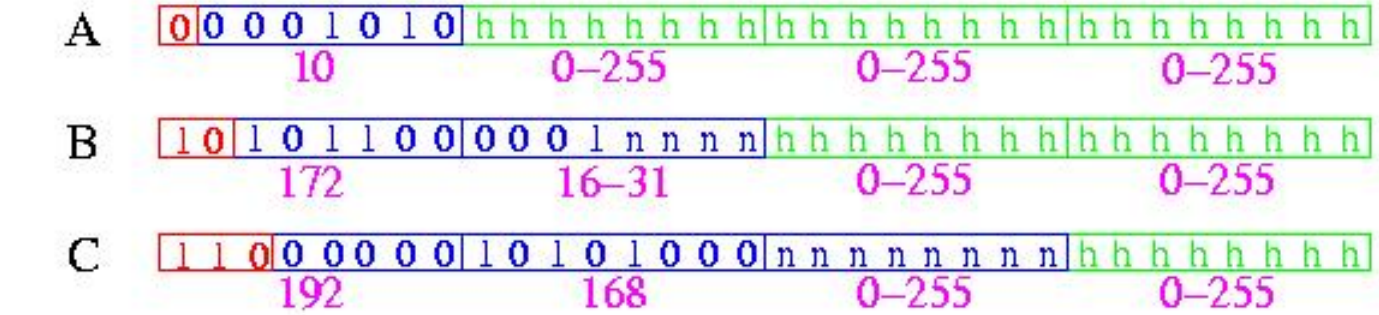
Pre-selected route?
Recorded route?
Security parameters?
Timestamps?

Typically a PDU for a higher level protocol (TCP, UDP,...)

IP Addresses

- An IP address identifies a system (network interface)
- Can be:
  - **Static** -- allocated by system manager during system setup.
  - **Dynamic** -- obtained by operating system from DHCP server during system boot.
  - **Global** -- uniquely identify system within entire Internet.
  - **Private** -- for use in a local subnet only.
  - **Unicast** -- refers to a single system.
  - **Multicast** -- refers to a group of systems.
- **IPv4**: 32 bits, written as 4 dec. numbers, each representing 8 bits, e.g.: `130.225.76.44`
- **IPv6**: 128 bits, written as 8 hex. numbers, each repres. 16 bits: `ff:aec1:0:0:0:ffff:fffe:1`


Address class
IPv4 address classes
A
0 n n n n n n n h h h h h h h h h h h h h h h h h h h h h h h h
1–127
0–255
0–255
0–255
B
1 0 n n n n n n n n n n n n n n h h h h h h h h h h h h h h h h
128–191
0–255
0–255
0–255
C
1 1 0 n n n n n n n n n n n n n n n n n n n n n n h h h h h h h h
192–223
0–255
0–255
0–255
Network id
Host id
Multi–cast
1 1 1 0
224–239
0–255
0–255
0–255
Multicast group id


Private Network addresses
A
0 0 0 0 1 0 1 0 h h h h h h h h h h h h h h h h h h h h h h h h
10
0–255
0–255
0–255
B
1 0 1 0 1 1 0 0 0 0 0 1 n n n n h h h h h h h h h h h h h h h h
172
16–31
0–255
0–255
C
1 1 0 0 0 0 0 0 1 0 1 0 1 0 0 0 n n n n n n n n h h h h h h h h
192
168
0–255
0–255

Internet Names

- A name identifies a system (network interface), *independently of its geographical position*.
- No fixed length, structure reflects the administrative domains responsible for allocating the name:

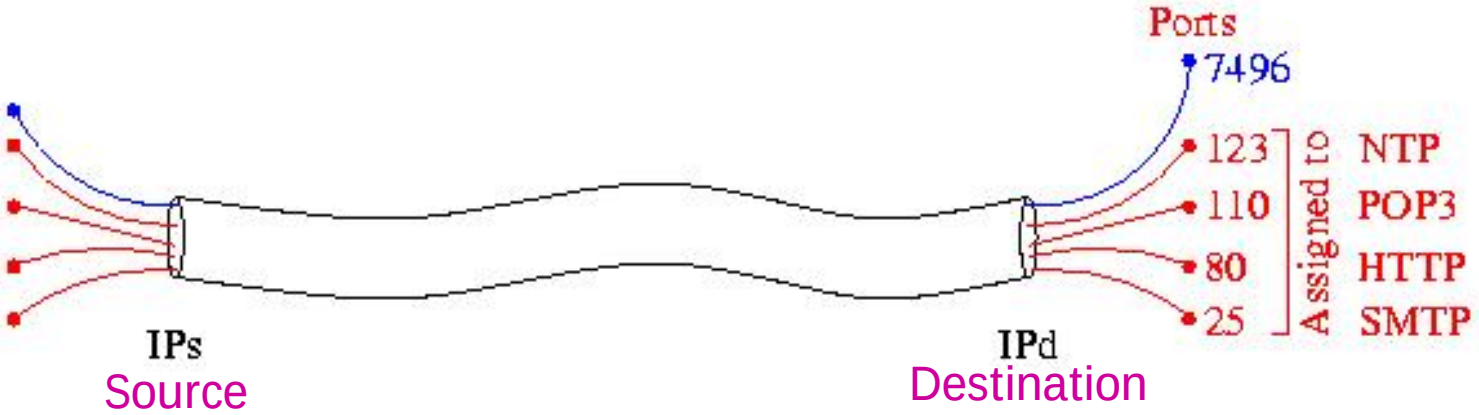
Least significant ← → Most significant

www.rfc-editor.org
hobbits.middle-earth.net
esmeralda.imm.dtu.dk
stop.it

- Mapping between names and addresses maintained by **Domain Name System (DNS).**
- Information inserted into and retrieved from DNS by using DNS A-layer protocol to contact **DNS server**.

Transmission Control Protocol

- TCP implements a connection-mode, full duplex, stream service for point-to-point data transfer between arbitrary processes.
- Described in Internet RFC793.
- Multiple flows of data between given source and dest. IP addresses distinguished by **port numbers.**



TCP Ports

- Port numbers lie in range [0..65535].
- Subintervals of range are used for specific purposes:
  - [0..1023] ***Assigned*** for use by servers for standard Internet applications, e.g.:
    - 25: SMTP
    - 53: DNS
    - 80: HTTP.

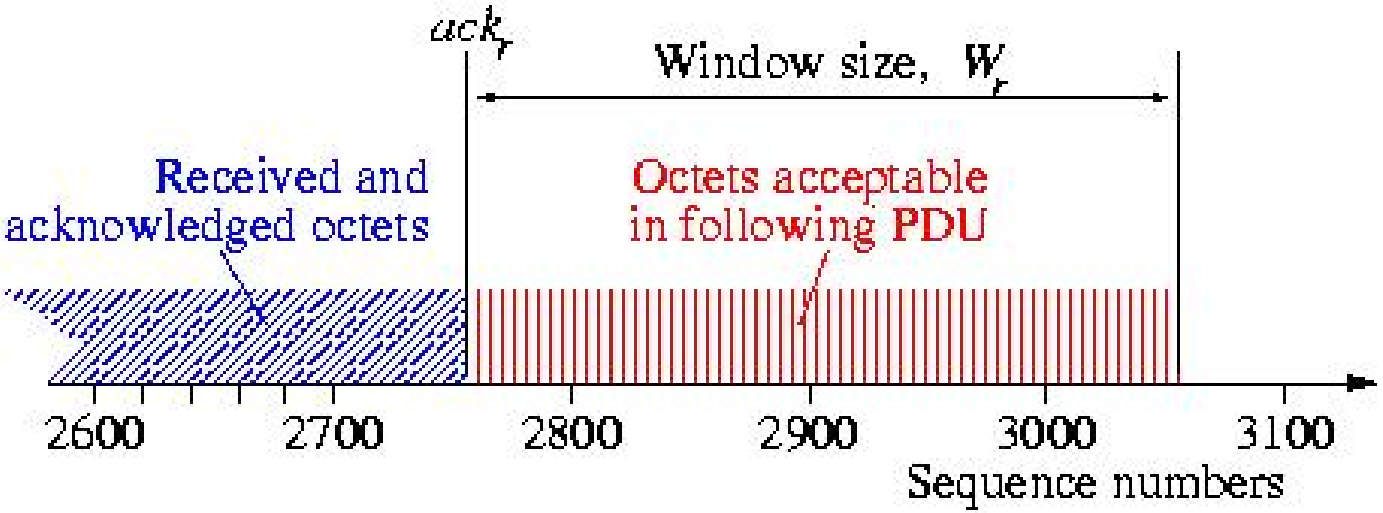
    Only use assigned ports for their assigned purpose!
  - [1024..49151] Can be ***registered*** with Internet Application Naming Authority (IANA) for use with specific applications. See **`www.iana.org`** for details.
  - [49152..65535] **Freely available** for use, for example when ports have to be dynamically allocated.

TCP operation

- Data in each direction is considered as a potentially unlimited **stream** of octets.
- Position of a given octet in stream is given by a **sequence number**.
- Each PDU from A to B contains:
  - **Seq.no.** n_s (modulo 2^{32}) of first octet of data in PDU.
  - **Acknowledgment** ack_r, expressed as seq.no. modulo 2^{32} of next octet expected from B. (This acknowledges receipt of all octets up to $(ack_r - 1)$ from B.)
  - **Credit value** W_r, giving no. of octets which A is willing to receive from B. W_r is often known as **receive window size**.

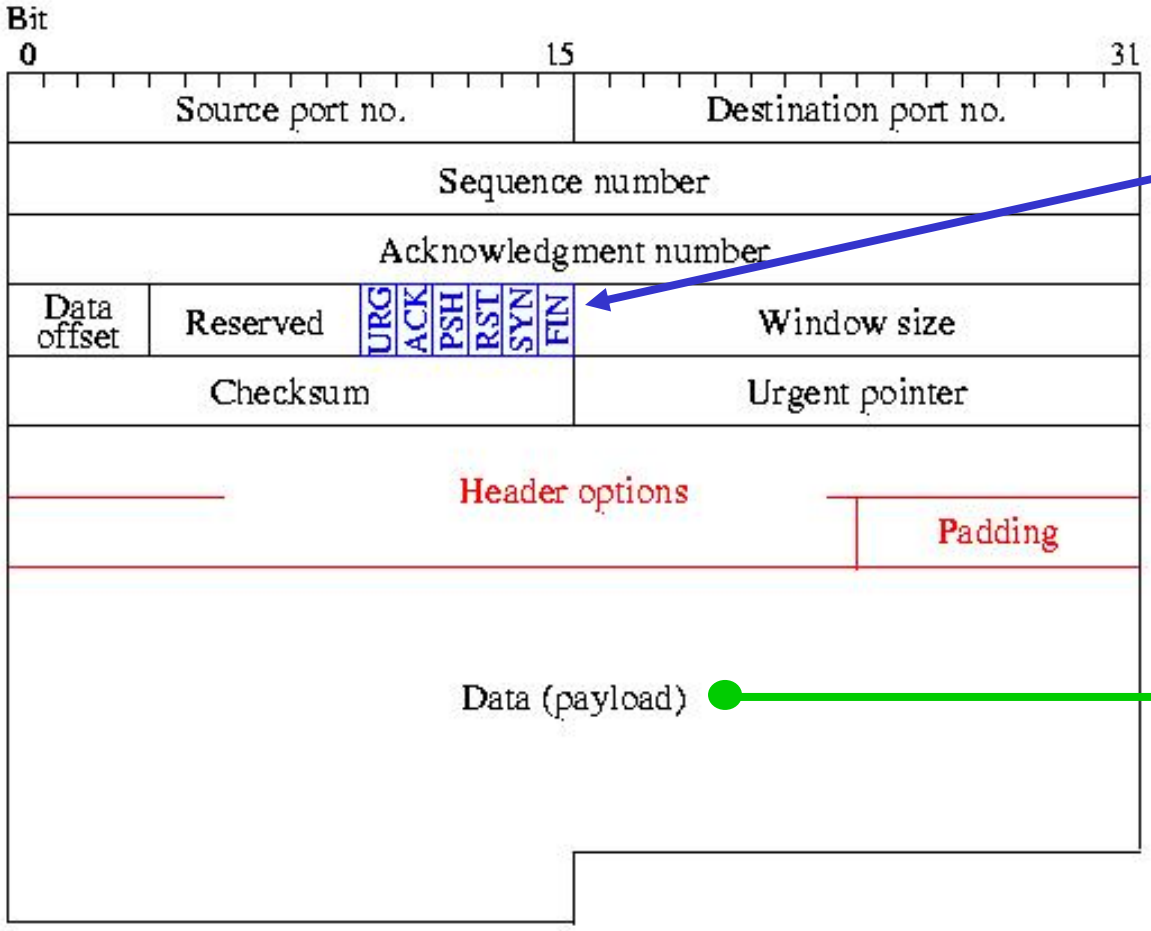
TCP receive window

- Operation of receive window: A sends ack_r and W_r to B, so that B will be informed of A's state:



- A has received octets up to and including $(ack_r - 1)$
- A is willing to receive octets from ack_r to $(ack_r + W_r - 1)$

TCP PDU



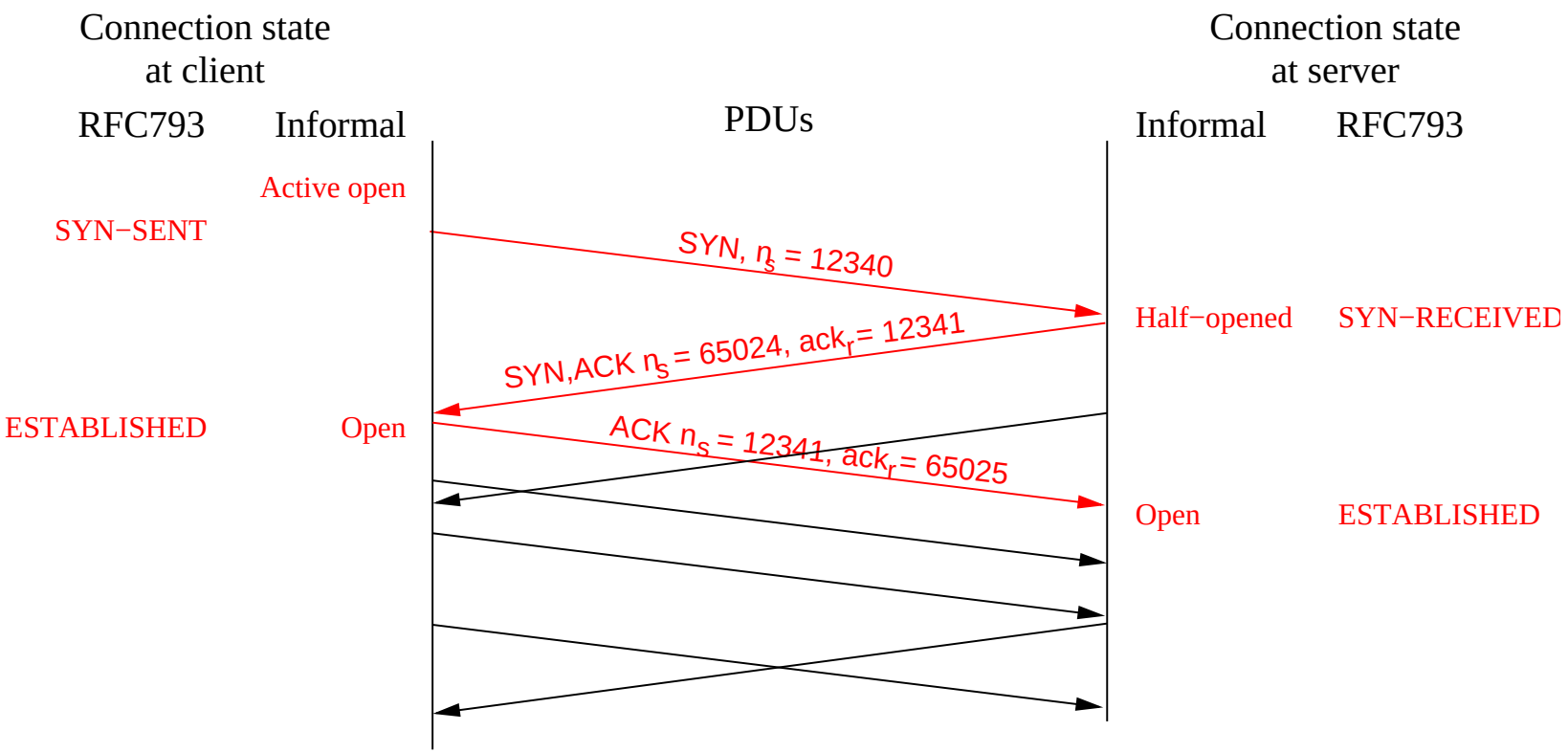
Only one format, **flags** are used to distinguish types of PDU.

Typically a PDU from an Application layer protocol.

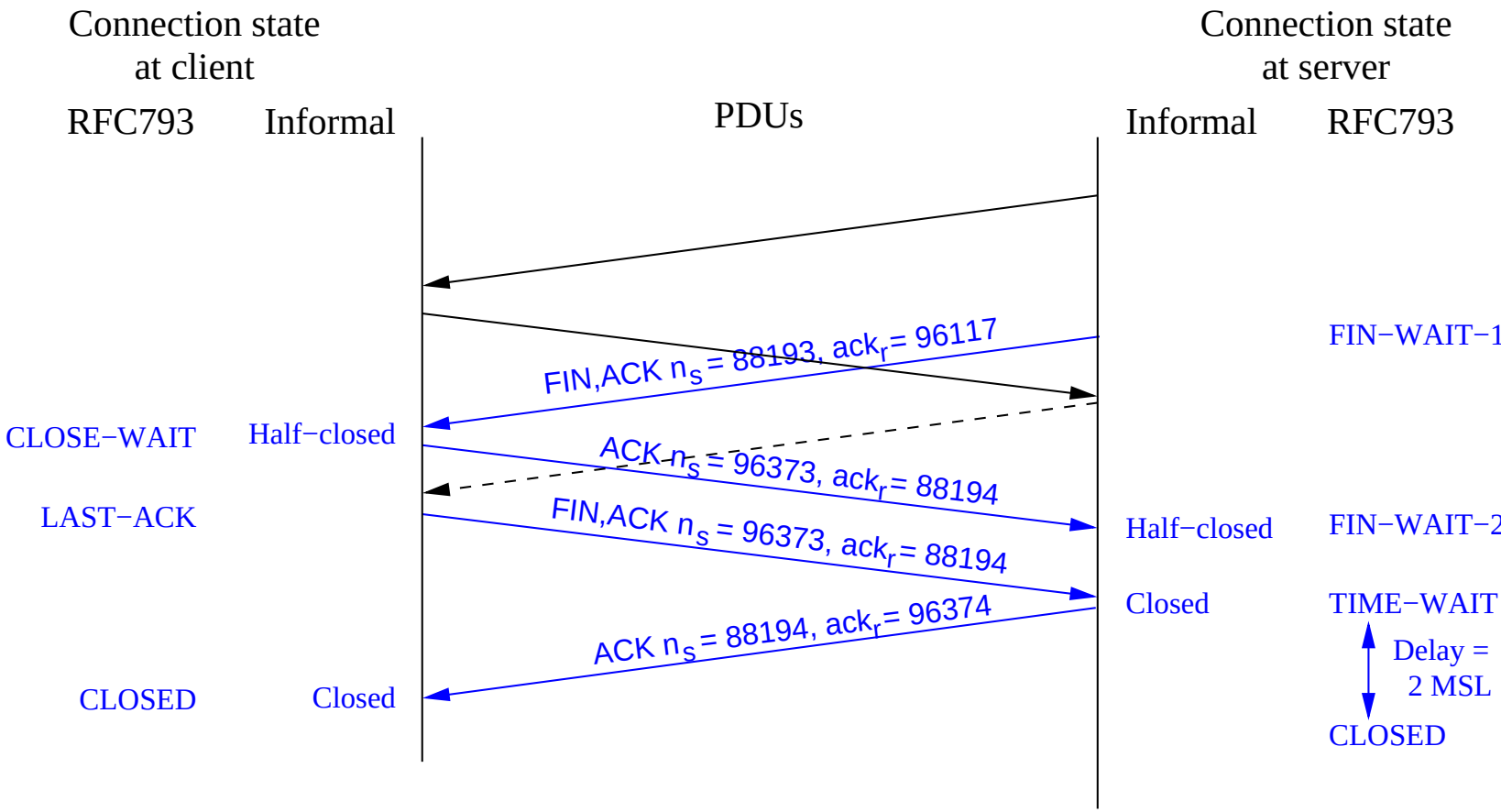
TCP principles of operation

- **Connection establishment:**
  - ”Three-way handshake” for two parties to agree on initial sequence numbers for data transfer in each direction.
- **Data transfer phase:**
  - Received data must be acknowledged. Sender retransmits data if no ACK received for these data within timeout period.
  - Suitable timeout period determined dynamically to reflect current conditions in network.
- **Closing the connection:**
  - Each party sends PDU with FIN(+ACK) to the other, waits for ACK in reply.
  - Each party closes its end independently of the other.
  - PDU with RST flag causes abnormal closing of connection.

TCP connection establishment



TCP closing connection



TCP implementation choices

- TCP protocol definition leaves many choices open.
- Implementation choices can affect the efficiency of the protocol. For example:
  - **When to send ACK:**
    - Send dataless ACK as soon as possible?
    - Wait (as long as possible) for user data, so ACK can be piggybacked on PDU with data?
  - **When to send data:**
    - Send data as soon as user provides some?

      Inefficient for character input.
    - Wait until a reasonable amount is provided?

      Principle of **Nagle's algorithm**.

      Can give long response times and odd behaviour.
- Some of these choices may be under program control.

Application Layer Protocols

Network Applications

- Typical types of application in a distributed system based on a network:
  - Data storage: Large amounts of data distributed among several (many?) computer systems.
  - Databases: Special case of data storage, where the stored data are systematically organised as a database.
  - Document handling: Mail, WWW, Workflow, Info. search.
  - Multimedia: Presentation of audio, video etc. in “real time”, pre-recorded or live (teleconferencing, teleimmersion).
  - Technical computations: Engineering (CAD/CAM/...), natural sciences, economic models etc.
  - E-commerce: E-buying/selling, e-banking, e-payment.
  - Public administration: Taxes, voting.

Application Layer protocols

- Are based on a (more or less reliable) Transport service — in the Internet, typically provided by TCP or UDP.
- May support various ways of organising an application. Common examples:
  - Peer-to-peer: Two or more participants with equal status.
  - Client/server: Two participants. One party (server) offers services to the other (client).
  - Agent-based: Several parties collaborate in an “intelligent” way.
  - Grid: (Very) large number of parties offer services, and system will find the most appropriate one.

Client/server Systems

A popular paradigm for design of distributed systems:

- ***Processes acting as Servers offer to perform services for processes acting as Clients.***

Most simple Internet applications rely on a Client/server architecture.

Why so popular?

- Client/server is distributed analogue of the Object Oriented programming paradigm: The server is an object whose methods can be invoked by the client.

Internet Client/server protocols

- Well-known examples of Internet A-layer protocols following the Client/server paradigm are:

SMTP	RFC821	Simple Mail Transfer Protocol *Transfers mail to recipient's mailbox.*
POP3	RFC1939	Post Office Protocol, version 3 *Retrieves mail from mailbox.*
NNTP	RFC977	Network News Transfer Protocol *Retrieves news from news service.*
FTP	RFC959	File Transfer Protocol *Transfers files.*
TELNET	RFC854	Virtual Terminal protocol *Uniform handling of diverse terminals.*
DNS	RFC1034, RFC1035	Domain Name Service *Registers/finds IP-addresses corresponding to names.*
HTTP	RFC2616	Hypertext Transfer Protocol *Retrieves documents from WWW.*
LDAP	RFC2251	Lightweight Directory Access Protocol *Lookup service for properties of objects..*

SMTP

Simple Mail Transfer (SMTP)

A simple A-protocol using the Client/server paradigm.

Involves a dialogue between Client and Server:

```
HELO goofy.dtu.dk
250 design.dilbert.org
MAIL FROM <bones@goofy.dtu.dk>
250 OK
RCPT TO <grass@design.dilbert.org>
250 OK
DATA
354 Start mail input; end with <CRLF>.<CRLF>
From: Alfred Bones <bones@goofy.dtu.dk>
To:   Smokey Grass <grass@design.dilbert.org>
Date: 21 Aug 2000 13:31:02 +0200
Subject: Client exploder
Here are the secret plans for the client exploder
   etc. etc.
.
250 OK
```

SMTP (2)

- A conversation between Client and Server involves a sequence of exchanges, in each of which:
  - Client sends a Command, for example:

HELO	Identifies sender
MAIL	Specifies route back to sender
RCPT	Identifies a recipient
DATA	Actual mail message
QUIT	Ends conversation

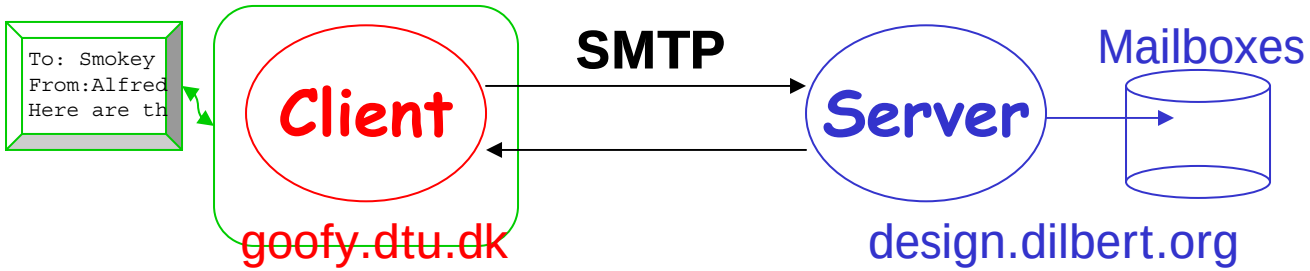
  - Server responds with a Reply, for example:

250	Requested mail action OK, completed.
354	Start mail input
500	Syntax error: Command unrecognised

  - Commands (and replies) may contain extra information, such as addresses or data.

Mail Application

- That was the protocol,



- But what does the application do?
  - Offers user interface so user can formulate mail, include attachments, give delivery instructions.
  - Runs SMTP client to send mail *from* user to the recipients' mailboxes on SMTP server.
  - Runs POP or IMAP client to retrieve mail sent *to* user.
  - Offers user interface so user can read received mail, display attachments, and possibly store received mail locally.
- Often part of a web browser, or other office package

Mail Extensions

- SMTP (like many other Internet A-layer protocols) uses ASCII encoding for all messages:
  - Message types and contents are expressed as sequences of characters from the US ASCII character set.
- Simple SMTP clients and servers can only deal with a single message body in ASCII text.
- Extensions are required in order to deal with:
  - Texts in languages with non-English letters (æþêçü...)
  - Non-text documents, such as images, video or audio.
  - Multi-part documents
- Standard set of extensions: MIME
  (Multipurpose Internet Mail Extensions)

MIME Encoding

- A way of encoding chunks of data (MIME entities) which contain non-ASCII characters.
- Originally defined for use with SMTP, but now also used in other contexts.
- Each entity encoded as a header followed by a body.
- Header is made up of one or more header fields:
  - **`Content-type`** of body: **`text, image, video, audio,`** possibly with subtype and/or parameters giving more detailed specification.
  - **`Content-transfer-encoding`** describing way in which body is encoded *in addition to* what is implied by type.
  - **`Content-id`** for referring to entity.
  - **`Content-description`** of entity in plain text.

MIME Encoding (2)

- Example: Multi-part mail message.

```
From: Ebenezer Snurd <ebes@bugeyed.monster>
To: Rod Akromats <rak@tundranet.ice>
Date: Wed, 09 Aug 2000 12:34:56 +0100 (CET)
Subject: Finalised material
---------------------------------------------
MIME-Version: 1.0
Content-type: multipart/mixed;
              boundary=5c12g7YTurb19zp4UX
```

Body of multi-part message

- All in ASCII, so can be sent using SMTP!

MIME Encoding (3)

- Body of multi-part message contains several entities:

```
--5c12g7YTurb19zp4Ux
Content-type: text/plain; charset=ISO-8859-1
Content-transfer-encoding: 8bit

  Dear Rod,

    Here are some recent pictures, including
  the one I told you about from the Clones.
  Enjoy!   /Ebe.

--5c12g7YTurb19zp4Ux
Content-type: image/jpeg
Content-transfer-encoding: base64

Ap3u107+sexfyfyd66menop4RorS8hach8tf3
  ...

--5c12g7YTurb19zp4Ux
```

Transfer Encodings

- Describe the way data have been transformed, so they can be passed correctly through the network.
- Five standard MIME transfer encodings:
  - 7-bit: No transformation. Data consist of lines of not more than 998 chars in 7-bit representation, separated by CRLF.
  - 8-bit: Similarly, using an 8-bit representation.
  - binary: No transformation. Arbitrary sequences of octets.
  - quoted-printable: Transformation such that:
    - Non-graphic characters,
    - Non-ASCII graphical characters (æþêçü...)
    - Equals sign
    - Trailing white space

    are replaced by =XY, where X, Y give hexadecimal char. code
  - base64: Transformation of binary data into 64-char ASCII subset.

HTTP

HTTP, version 1.1

- A Client/server protocol for handling Web documents.
- Client is (typically) integrated into Web browser.
- Server is a Web server, somewhere in Cyberspace...
- Conversation between Client and Server consists of a sequence of exchanges, in each of which:
  - Client sends a Request which:
    - Identifies a resource, by giving a file name on the server.
    - Specifies an action ("method") to be performed on it.
    - Optionally gives details of rules to be followed in carrying out the action.
  - Server replies with a Response which gives status and possibly further explanation of what has happened.

HTTP Client/server exchanges

- Simple example:
  - Client sends Request:

```
GET http://www.w3.org/pub/WWW/xy.html HTTP/1.1
```

Method | Protocol+Server | File name | HTTP version

Uniform Resource Identifier (URI)

  - Server sends Response:

```
HTTP/1.1 200 OK
```

Status

```
<html>
<header><title>Progress on XY</title>
</header>
<body>
  …
</body>
</html>
```

Content of resource

HTTP Exchanges (2)

- Standard methods which the Client can request are:

GET	Retrieve content of resource.
PUT	Store new content in resource.
DELETE	Delete resource.
OPTIONS	Request information about resource or server.
HEAD	Get headers but not content of resource.
POST	Transfer information to an application, for example for transmission as mail, processing as a Web form, etc.
TRACE	Trace route to server via loop-back connection.

- Of course, some of these require suitable authorisation from the Server!

HTTP Exchanges (3)

- More complex forms of Request allow the Client to:
  - Define acceptable media types, character sets, languages,...
  - Request only parts of a document.
  - Control caching of the document.
  - Provide security information for authentication purposes.
- Example:

```
GET pub/WWW/xy.html HTTP/1.1
Host: www.w3.org
Accept: text/html, text/x-dvi;q=0.8
Accept-Charset: iso-8859-1, unicode-1-1;q=0.5
Accept-Encoding: gzip, identity;q=0.5, *;q=0
Accept-Language: da, en-gb;q=0.8, en;q=0
Range: bytes=500-999
Cache-Control: max-age=600
```

HTTP Exchanges (4)

- More complex forms of Response allow the Server to:
  - Give information about errors.
  - Indicate that only part of a request has been fulfilled.
  - Provide information about the capabilities of the Server, such as whether it supports selection of ranges.
  - Provide information about the specified resource, such as:
    - Which methods can be executed on it.
    - Its size.
    - Which media type(s), character set(s) etc. it uses.
    - A message digest to ensure integrity of the content.
- Simple example:

```
HTTP/1.1 405 Method Not Allowed
Allow: GET, HEAD, PUT
```

HTTP Exchanges (4)

- A more complex example of a Response:

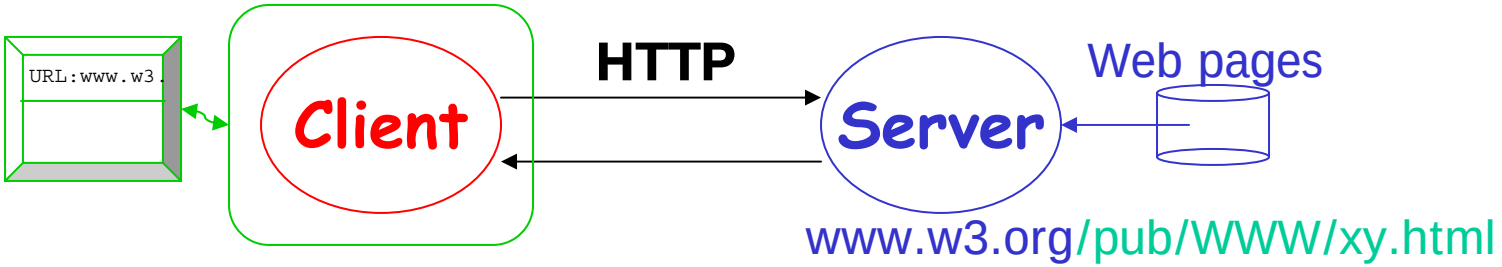
```
HTTP/1.1 200 OK
Date: Thu, 8 Aug 2002 08:12:31 EST
Content-Length: 332
Content-Type: text/html; charset=iso-8859-1
Content-Encoding: identity
Content-Language: en
Content-MD5: ohazEqjF+PGOc7B5xumdgQ==
Last-Modified: Mon, 29 Jul 2002 23:54:01 EST
Age: 243

<html>
  …
</html>
```

- Gives more details of the resource content.

Web Browser Application

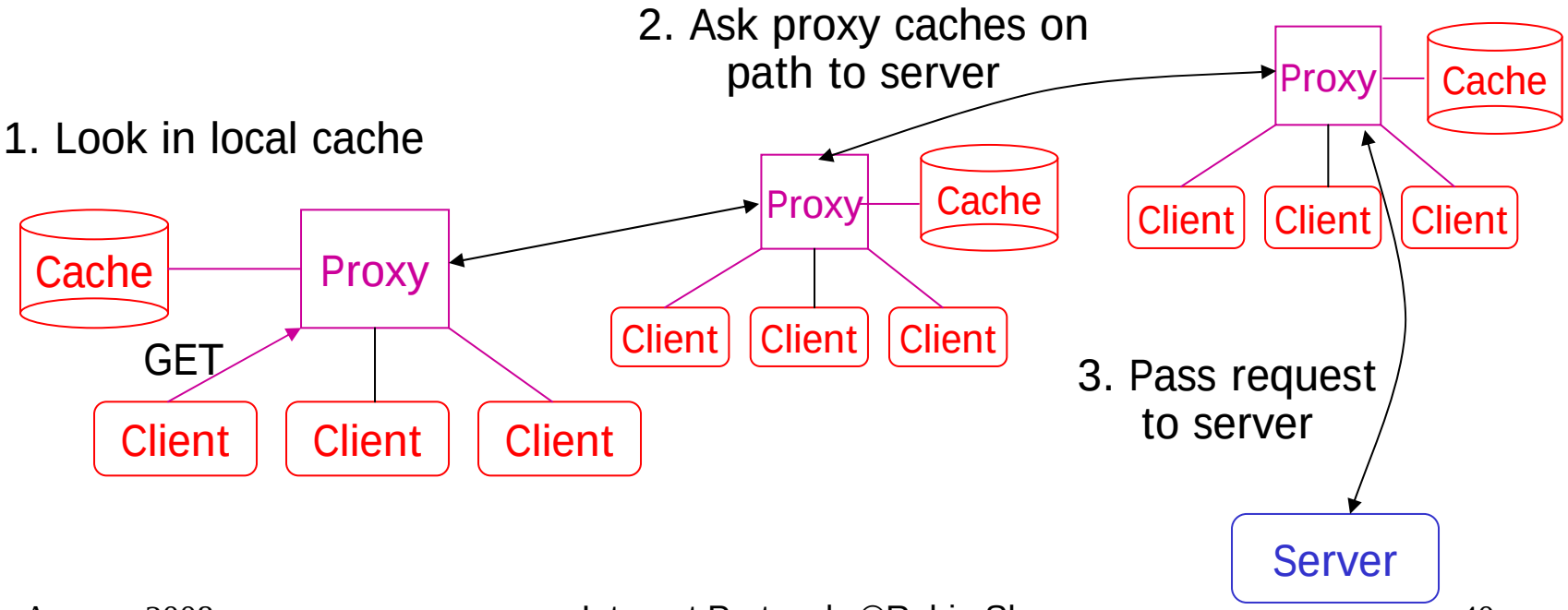
- That was the protocol,

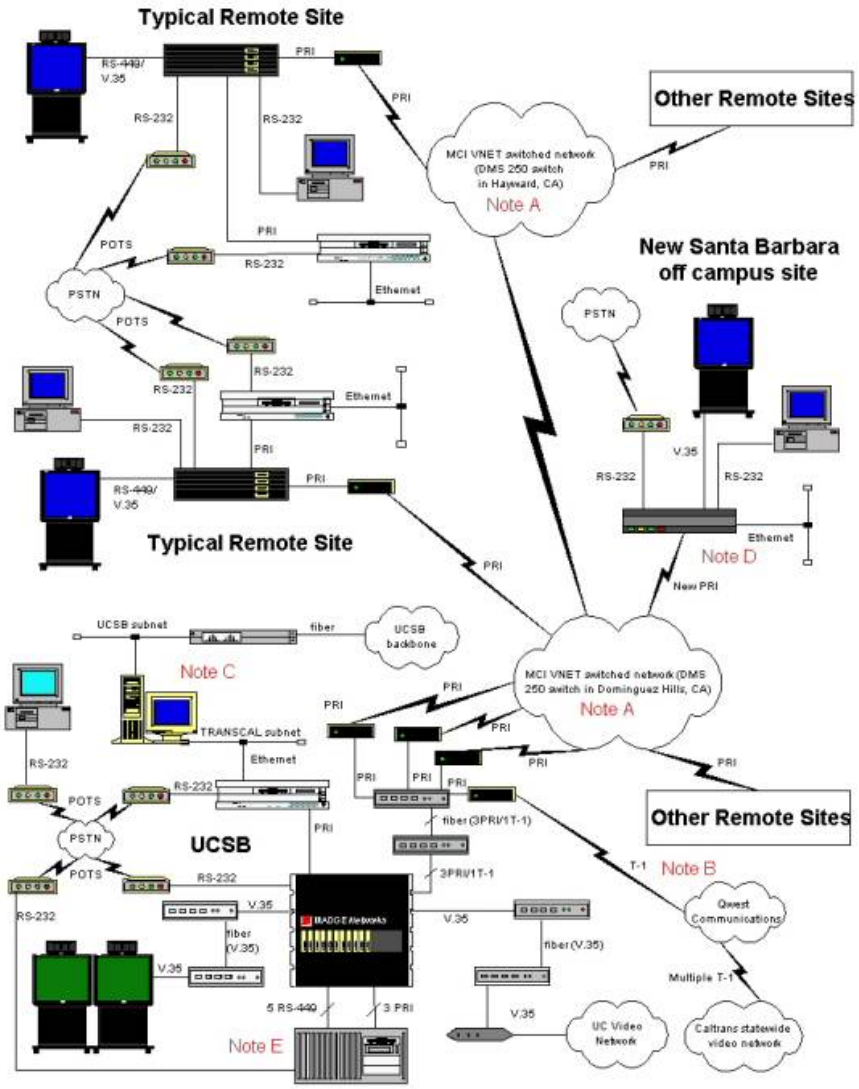


- But what does the application do?
  - Offers a user interface so user can specify where to start browsing (the "URL") and other requirements.
  - Runs the HTTP client to retrieve Web page given by URL.
  - Interprets the Web page in a manner depending on which mark-up language it is written in. This may involve:
    - Display of static elements (text, images).
    - Retrieval of further pages referred to by embedded links.
    - Execution of programs to generate dynamic elements.

Caching

- To avoid repeated retrieval of the same data from the server, many client/server systems use a cache, which contains a (recent) copy.
- Access to cache can be direct or go via a proxy which serves several clients on the same system.





Thank you for your attention