AOT LAB

Agent and **O**bject **T**echnology **Lab**
Dipartimento di Ingegneria dell’Informazione
Università degli Studi di Parma

Computer Network

Application Layer Services and Protocols

Prof. Agostino Poggi

Application Layer Responsibilities

- Identifying and establishing the availability of intended communication partners
- Synchronizing cooperating applications
- Establishing agreement on procedures for error recovery
- Controlling data integrity

Services and Protocols

- TCP/IP supports different services and protocols at the application layer
  - Remote computing: TELNET
  - File transfer: FTP and TFTP
  - Electronic mail: SMTP, POP3 and IMAP
  - Symbolic name resolution: DNS
  - Information browsing: HTTP

Client-Server Relationships

- One application component, called **Server**, provides well-defined services for application components running, called **Client**
  - Client makes a request for services by transmitting data to the server
  - Server replies by sending data back to the client

TELNET

- Remote login service allows user to establish a login session on a remote machine
- The service is called transparent because it gives the appearance that the user terminal attaches directly to the remote machine
- TCP/IP protocol suite includes a simple remote terminal protocol called TELNET
- TELNET allows user to establish a TCP connection to a login server

TELNET Protocol

- TELNET protocol specifies exactly how a remote login client and a remote login server interact

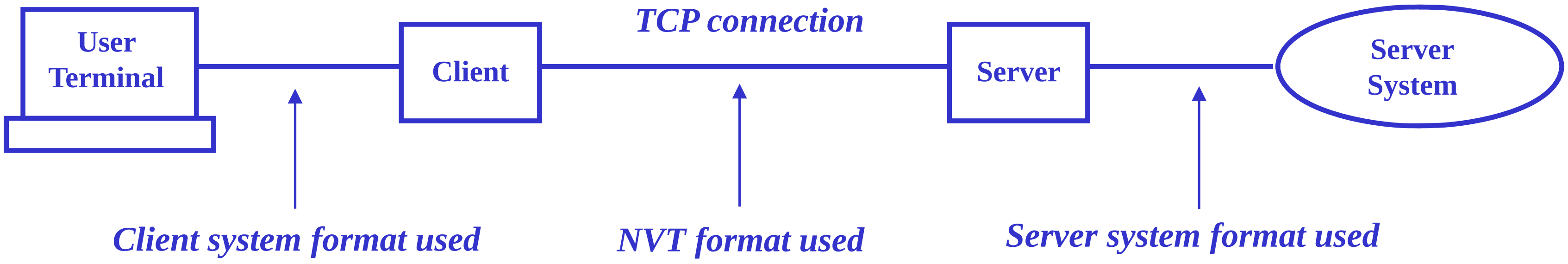
- TELNET protocol is built upon two main ideas
  - **Network Virtual Terminal**
    - Eliminates the need for "server" and “client" hosts to keep information about the characteristics of each other's terminals and terminal handling conventions
  - **Negotiated options**

TELNET Client

- TELNET client (telnet) emulates dumb terminals (e.g., VT100)
  - Client computer acts as if it were a locally connected dumb terminal
  - Local computer does no processing of data
  - Does not support file transfer but allows the capture of received data
  - Provides a character-oriented interface

Terminal Heterogeneity

- Problems
  - Different set of characters
  - Different coding
  - …
- TELNET defines how data and command sequences are sent across the internet: *Network Virtual Terminal* (NVT)
  - Provides a standard language for communication of terminal control functions
  - Intermediate representation of a generic terminal



Security and Graphical Issue

- In the TELNET protocol, everything is sent in “free text” or “in the clear”
  - A packet sniffer can see each letter that was being pressed as it was being typed
  - Given that telnet is so frequently used to connect to remote machines that request passwords, telnet is a clear security danger
- Moreover, telnet provides only character-oriented interface
- As a result, many organizations have tried to move away from using TELNET
  - SSH solutions are used in many organizations providing an improved security
  - *Remote desktop* technology displays an exact copy of one computer’s screen on another computer

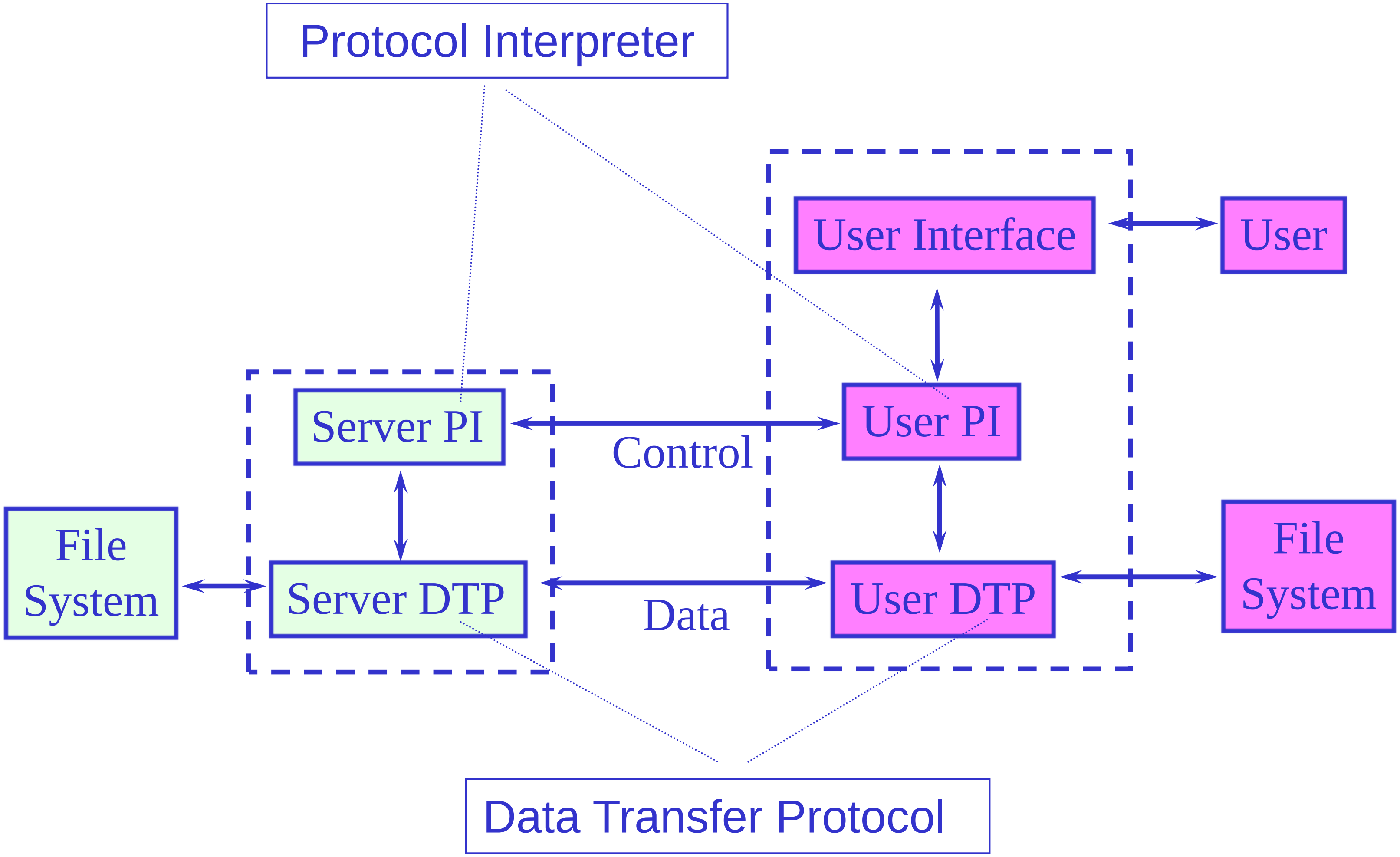
FTP

- Many network system provide computers with the ability to access file on remote machines
- The TCP/IP protocol suite includes a simple remote terminal protocol called File Transfer Protocol (FTP)
- FTP provides transfer data reliably and efficiently
  - Secure and reliable
  - Optimized for large file transfers
- FTP is designed mainly for use by programs, but can be used directly by a user at a terminal
  - The attempt is to satisfy the diverse needs of users of maxi-hosts, mini-hosts, personal workstations, with a simple, and easily implemented protocol design

FTP Model

- FTP is an unusual service in that it utilizes two ports, a 'data' port and a 'command' port
  - Control functions (commands) and reply codes are transferred over the *control connection*
    - The control connection is the “well known” service (port 21)
    - The control connection uses the TELNET protocol
  - All data transfer takes place over the *data connection* (traditionally port 20, but it depends on the mode)
    - The control connection must be “up” while data transfer takes place
    - The data connection may be used in either direction
    - The data connection need not exist all of the time

FTP Model


Protocol Interpreter
User Interface
User
Server PI
Control
User PI
File System
Server DTP
Data
User DTP
File System
Data Transfer Protocol

Client – Server Control Interaction

- Client sends FTP commands through TELNET character strings transmitted over the control connection
  - The commands begin with a command code followed by an argument field (no case sensitive)
- Server responds with replies over the control connection
  - Replies are a single line containing a status code (for programs) followed by a text message (for humans)

Data Transfer Modes

- Three different transfer modes are possible
  - *STREAM*: file is transmitted as a stream of bytes
    - There is no restriction on the representation type used (e.g., record structures are allowed)
  - *BLOCK*: file is transmitted as a series of blocks preceded by one or more header bytes
    - The header bytes contain a count field, and descriptor code
  - *COMPRESSED*: file is transmitted as a series of compressed blocks
    - Block are compressed through a simple compression scheme

Text/Binary File Management

- FTP handles all types of files (“binary” and “text”)
  - In text mode appropriate conversion takes place
    - if you are going from Unix to Windows, every <LF> character is replaced with two characters, <CR><LF>
  - Non-text files (e.g., images, MS Word document, ...) require binary mode

FTP Commands

Command	Description
open	Open a connection to a remote host
close	Closes a host connection
dir, ls	List the directory of files
mkdir, rmdir	Directory operations
get, mget	Retrieve file(s) from remote directory
put, mput	Copy file(s) to remote directory
verbose, status	Provides information about session
rename	Rename file
delete	Delete file

A Sample FTP Session

```
C:\>ftp aot.ce.unipr.it
Connected to aot.ce.unipr.it.
220 aot FTP server (Version 4.1 Sat Nov 23 12:52:09 CET 2002)
   ready.
Name (rs60002): poggi
331 Password required for poggi.
Password: xxxxxx
230 User poggi logged in.
ftp> put file01.txt file01.txt
200 PORT command successful.
150 Opening data connection for file01.txt (1252 bytes).
226 Transfer complete.
local: file01.txt remote: file01.txt
1285 bytes received in 0.062 seconds (20 Kbytes/s)
ftp> close
221 Goodbye.
ftp> quit
```

Anonymous FTP

- For acceding to a FTP server it is necessary to know
  - host machine name
  - login id or user name
  - password
- For anonymous FTP, the user name is "*anonymous*", and the password is usually the email address
- The Server gives read-only access to a set portion of their file system

```
    ftp> ...
    ftp> 230 guest login OK, access restrictions
apply
```

- “ftp” URLs are considered anonymous ftp

TFTP

- Trivial File Transfer Protocol (TFTP) is a simplified version of FTP
- Transfer files between processes
  - It cannot list directories
- Minimal overhead (no security)
  - No user authentication
- Designed for UDP, although could be used with many transport protocols
- Easy to implement
- Small - possible to include in firmware
  - Often uses to bootstrap workstations and network devices

TFTP Protocol

- 5 message types
  - Read request
  - Write request
  - Data
  - ACK (acknowledgment)
  - Error

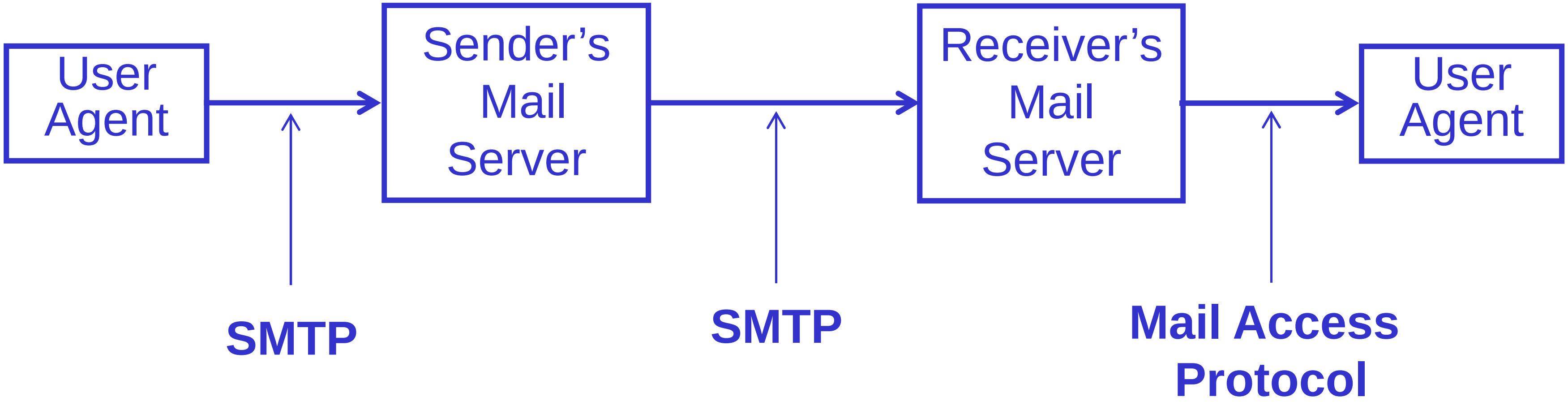
- TFTP transfer modes
  - Netascii: for transferring text files
  - Octet: for transferring binary files

SMTP Protocol

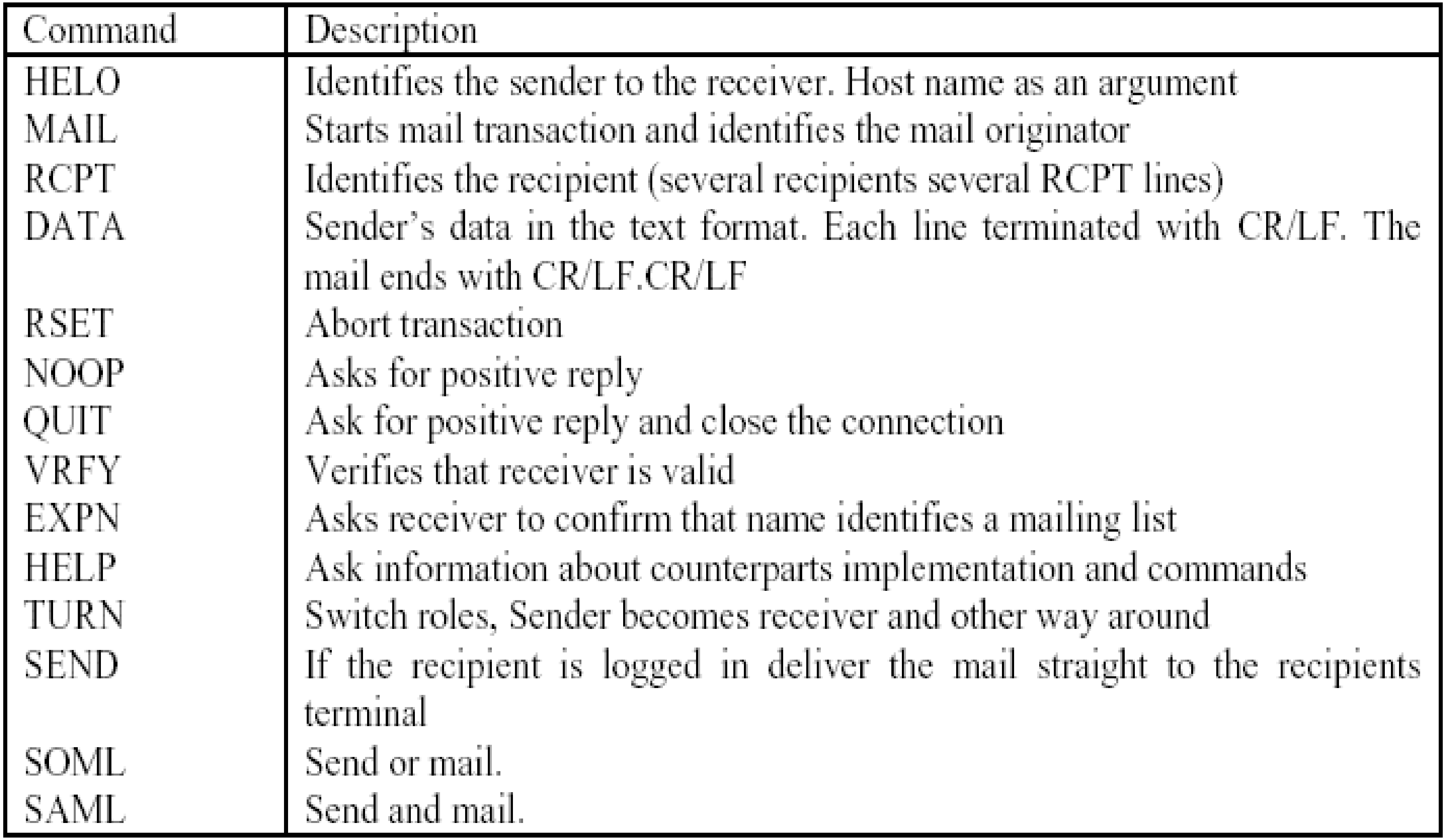
- The simple mail transfer protocol (SMTP) provides a simple way to transfer electronic mail between a sender (client) and a receiver (server)
  - Sender is a client (Outlook, Sendmail, a mail server, ...) and establishes a TCP connection (port 25) to the receiver (mail.libero.it, smtp.tiscali.it, ...)
  - Receiver is a server (mail server) that accepts incoming connections (port 25)
    - It can be the final receiver (copies messages into the appropriate mailboxes)
    - Some intermediate host (relay MTA)

SMTP Protocol

- The sender and the receiver are called message transfer agent (MTA)
  - User agent (Outlook, Mail, ...) can
    - Act directly as a sender MTA (Outlook)
    - Communicate with a separate local sender MTA (Mail uses Sendmail)
  - Mail servers are MTA



SMTP Commands



Command	Description
HELO	Identifies the sender to the receiver. Host name as an argument
MAIL	Starts mail transaction and identifies the mail originator
RCPT	Identifies the recipient (several recipients several RCPT lines)
DATA	Sender's data in the text format. Each line terminated with CR/LF. The mail ends with CR/LF.CR/LF
RSET	Abort transaction
NOOP	Asks for positive reply
QUIT	Ask for positive reply and close the connection
VRFY	Verifies that receiver is valid
EXPN	Asks receiver to confirm that name identifies a mailing list
HELP	Ask information about counterparts implementation and commands
TURN	Switch roles, Sender becomes receiver and other way around
SEND	If the recipient is logged in deliver the mail straight to the recipients terminal
SOML	Send or mail.
SAML	Send and mail.

SMTP Message Structure

- Electronic Mail is composed of two pieces
  - **Headers**
  - **Body**
- Each header field contains a name, followed by a colon, followed by the field value

```
Return-Path: <smith@any.com>
Received: from <any.com> ...
Reply-to: <smith@here.com>
From: “John Smith” <smith@any.com>
To: “Paul Brown” <brown@some.com>
Subject: mail testing
Date: Sat, 8 Feb 2003 23:14:47
```

SMTP Message Structure

- Headers beginning with an X- are user-defined fields
  - X-Priority: 3 (Normal)
  - X-MSMail-Priority: Normal
  - X-Mailer: Microsoft Outlook, Build 10.0.2627
- Long header fields are folded onto multiple lines with the additional lines starting with white spaces
- The body is the content of the message

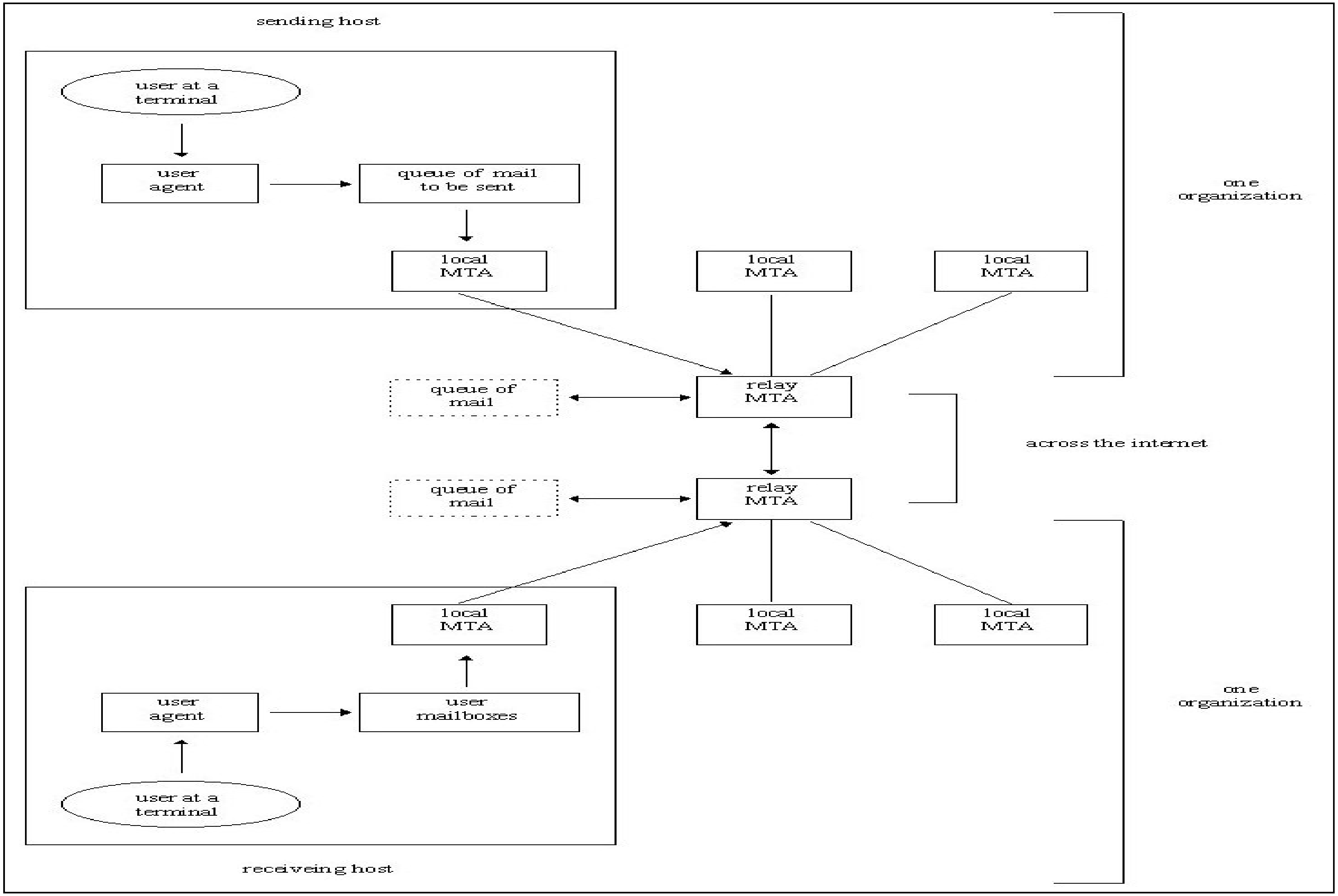
SMTP Message Structure

- Headers and body are sent by the sender with the DATA command
  - Headers are sent first, followed by a blank line, followed by the body
- The user agent takes what we specify as a body, adds some headers and passes the result to the MTA
- The MTA adds a few headers (time stamp, ...) and sends the result to another MTA

Relay MTA

- Most systems are configured to send all non local outgoing mail to a relay MTA for delivery
  - It simplifies the configuration of all MTAs other than the relay system’s MTA
  - It allows one system at an organization to act as the mail hub, possibly hiding all the individual systems
- The relay MTA takes the direct connection to the receiver relay MTA, which directs the mail to the local MTA on the receivers host

Relay MTA


sending host
user at a terminal
user agent
queue of mail to be sent
local MTA
local MTA
local MTA
one organization
queue of mail
relay MTA
across the internet
queue of mail
relay MTA
local MTA
local MTA
local MTA
user agent
user mailboxes
user at a terminal
one organization
receiveing host

ESMTP Protocol

- One shortage in the SMTP protocol is that it can send only simple text messages, expressed using 7-bit ASCII text (high order bit is cleared) with a maximum line length of 1000 characters

- This problem is solved with the extended SMTP message transfer agent (ESMTP protocol)

MIME

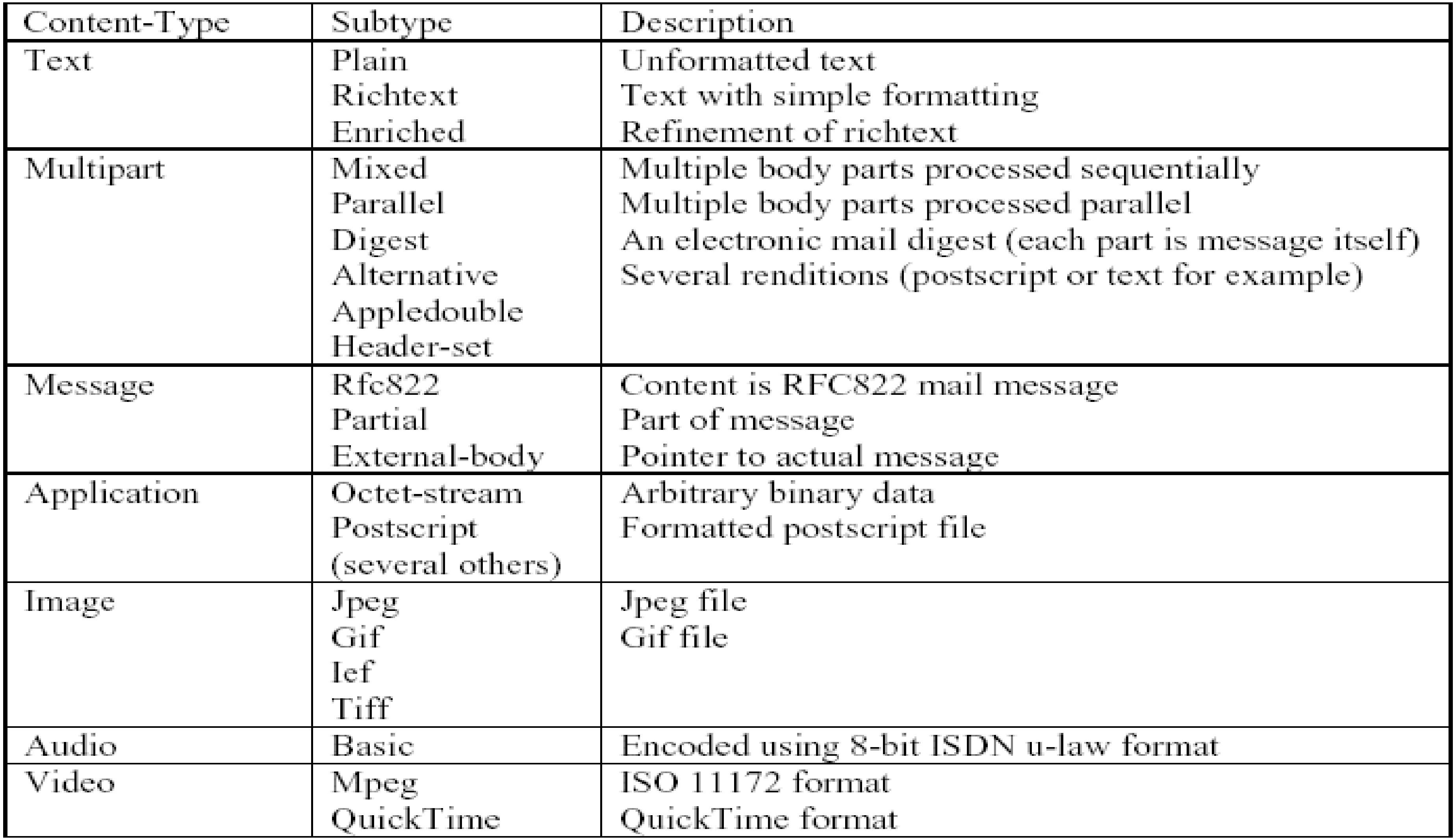
- MIME (Multipurpose Internet Mail Extensions) defines the format of message bodies to allow multi-part textual and non-textual (non-ASCII) message bodies
  - Includes multiple objects in a single message (text, attachments, …)
  - Represents body text in character sets other than US-ASCII
  - Represents formatted multi-font text messages
  - Represents non-textual (non ASCII) material such as images, audio fragments, programs, …, and in general, binary files

MIME

- The format of MIME message sent contains some additional headers

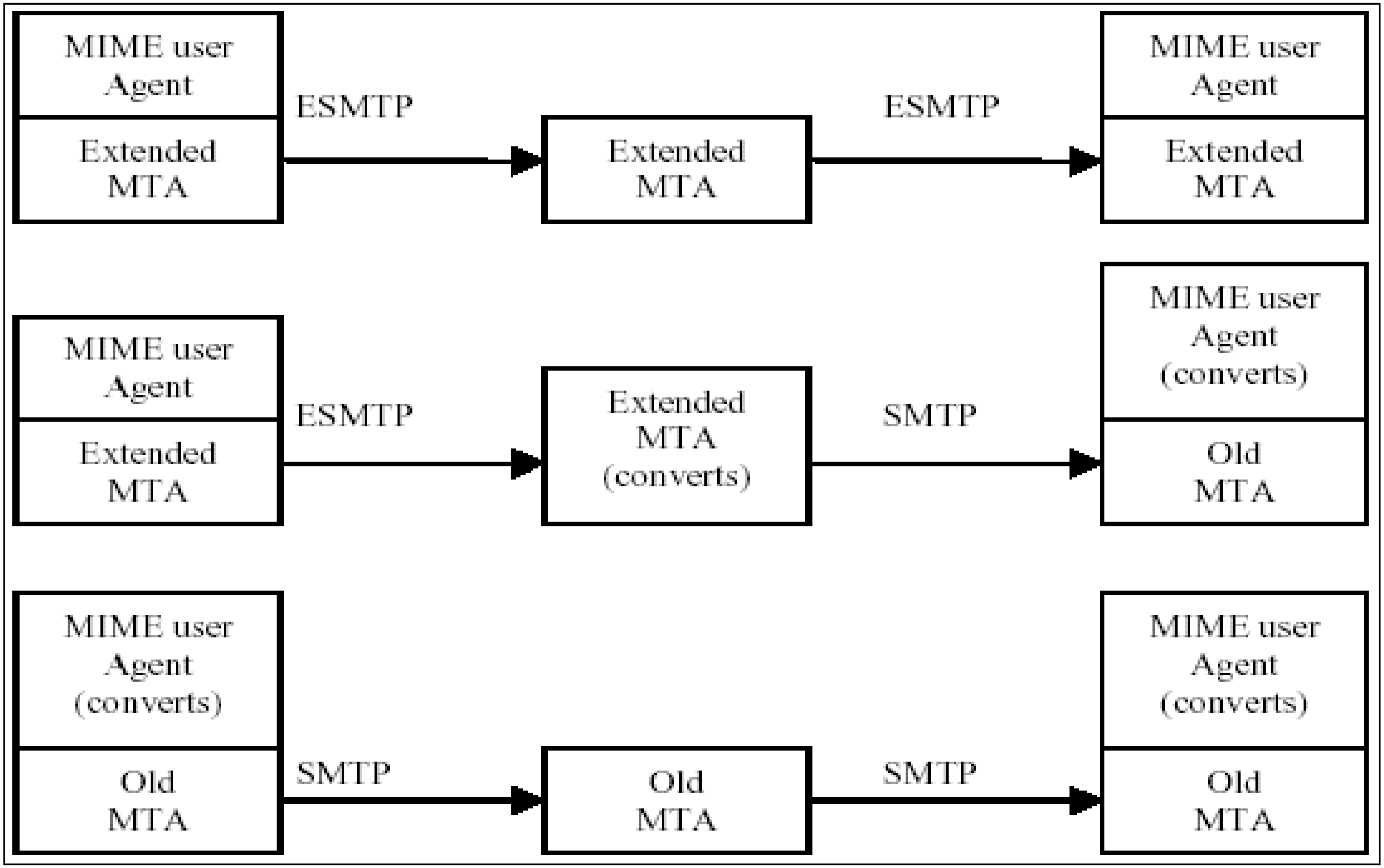
- These headers contain information about the structure and the content of the message
    - A MIME-Version header field
    - *Content-Type* header fields, to specify the type, subtype of data in the body and other parameters

MIME



Content-Type	Subtype	Description
Text	Plain Richtext Enriched	Unformatted text Text with simple formatting Refinement of richtext
Multipart	Mixed Parallel Digest Alternative Appledouble Header-set	Multiple body parts processed sequentially Multiple body parts processed parallel An electronic mail digest (each part is message itself) Several renditions (postscript or text for example)
Message	Rfc822 Partial External-body	Content is RFC822 mail message Part of message Pointer to actual message
Application	Octet-stream Postscript (several others)	Arbitrary binary data Formatted postscript file
Image	Jpeg Gif Ief Tiff	Jpeg file Gif file
Audio	Basic	Encoded using 8-bit ISDN u-law format
Video	Mpeg QuickTime	ISO 11172 format QuickTime format

MIME & ESMTP Protocol



POP Protocol

- Post Office Protocol (POP) is used to allow an user (client host) to retrieve mail that the server (server host) is holding for it

- When a client host wishes to make use of the service, it establishes a TCP connection on port 110 with the server host

POP Protocol

- Once the TCP connection has been opened and the POP3 server has sent the greeting, the session progresses through three states
  - Authorization state
    - The client must identify itself to the POP3 server
    - Commands: USER name, PASS string, QUIT
  - Transaction state
    - The client requests actions on the part of the POP3 server; terminated by the command QUIT
    - Commands: STAT, LIST [msg], RETR msg, DELE msg, NOOP, RSET, QUIT
  - Update state
    - The POP3 server releases any resources acquired during the transaction state; the TCP connection is then closed
    - Commands: UPDATE, QUIT

IMAP Protocol

- Internet Message Access Protocol (IMAP) allows user to organize messages in folders with hierarchical mailbox naming
- Allows mailbox/message management on server
  - Folder create, delete, rename, message copy, move, delete, …
  - Permanent message flag (seen, answered, urgent, deleted, …)
  - Selective fetching of message attributes, texts, and portions
  - Full MIME management
- IMAP does not specify a means of posting mail
  - This function is handled by a mail transfer protocol such as SMTP

IMAP Protocol

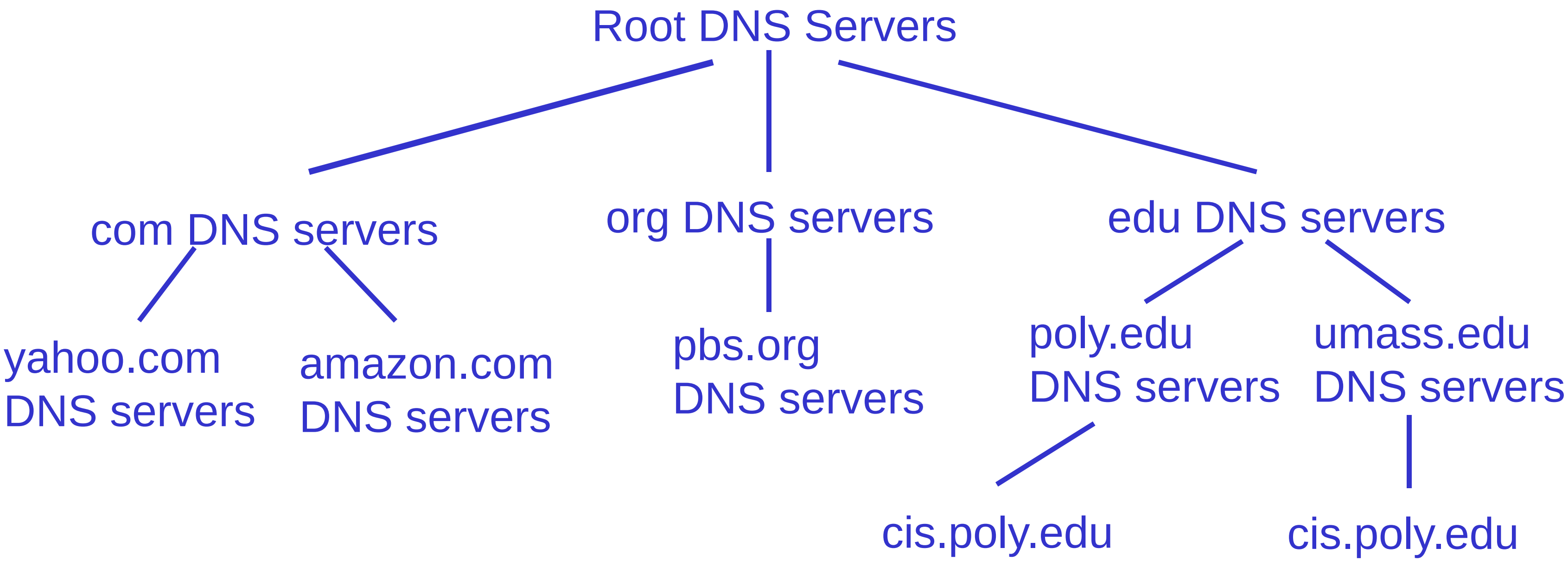
- An IMAP connection consists of the establishment of a client/server connection, initial greeting from the server, and client/server interactions
  - Client/server interactions consist of a client command, server data, and a server completion result response

- An IMAP session progresses through four states
  - Non-Authenticated state
    - The client MUST supply authentication credentials
  - Authenticated state
    - The client is authenticated and MUST select a mailbox to access
  - Selected state
    - A mailbox has been selected to access
  - Logout state
    - The connection has being terminated, and the server will close the connection

DNS

- Humans use symbolic names to identify hosts, but hosts use IP addresses
- A DNS (Domain Name Server) maps symbolic names to IP addresses taking advantage of a hierarchy of other DNS
- DNSs work as a distributed database
  - Root Name Servers
  - Top-level domain (TLD) servers
  - Authoritative DNS servers
  - Local Name Servers

DNS Hierarchy



Root DNS Servers

com DNS servers

org DNS servers

edu DNS servers

yahoo.com DNS servers

amazon.com DNS servers

pbs.org DNS servers

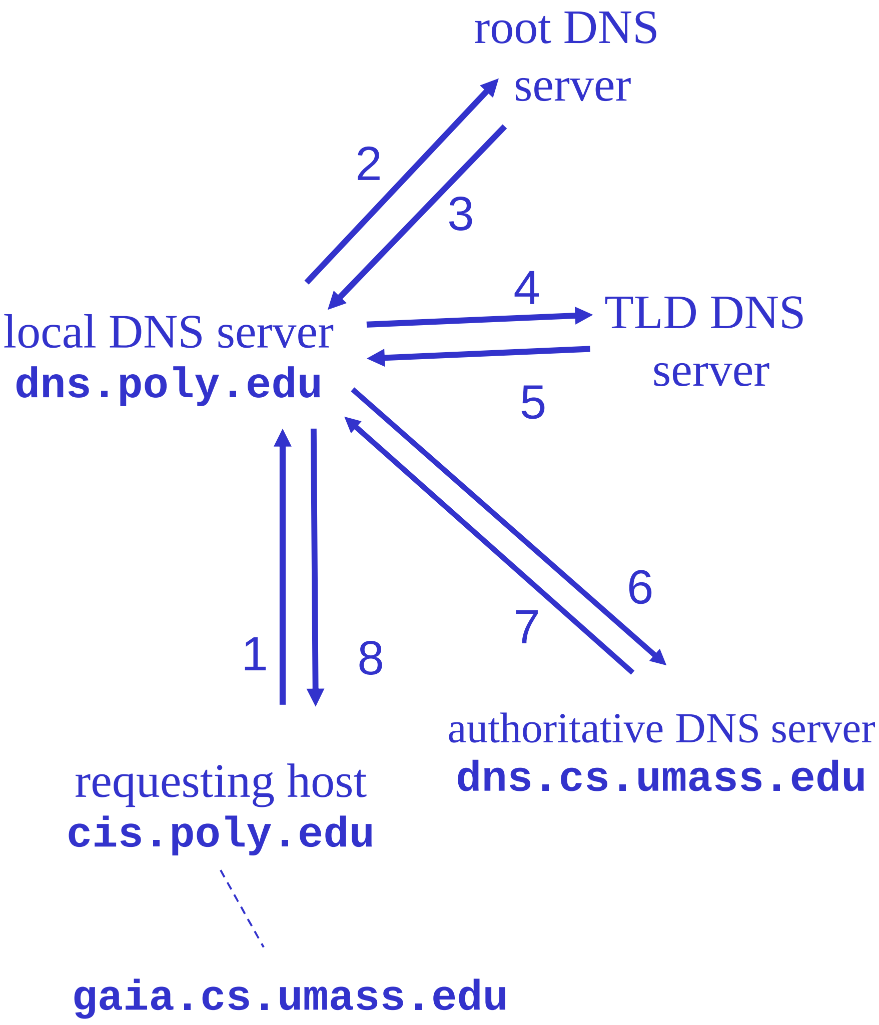
poly.edu DNS servers

umass.edu DNS servers

cis.poly.edu

cis.poly.edu

Iterative & Recursive Query



root DNS server

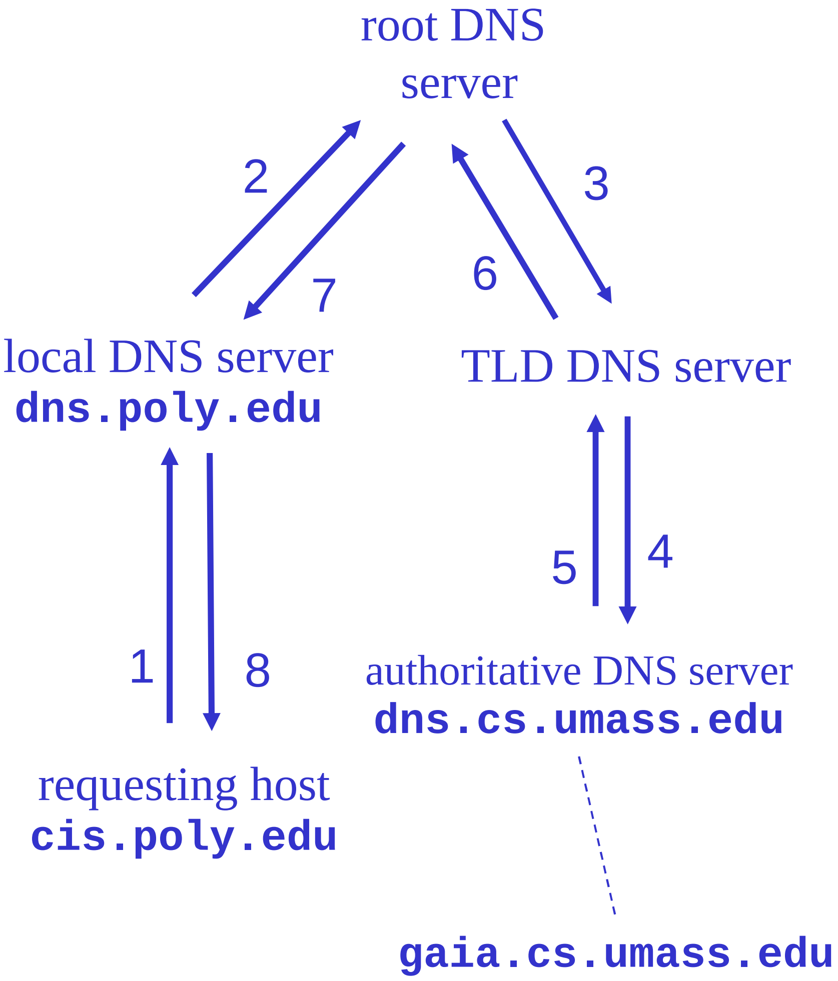
local DNS server
`dns.poly.edu`

TLD DNS server

authoritative DNS server
`dns.cs.umass.edu`

requesting host
`cis.poly.edu`

`gaia.cs.umass.edu`



root DNS server

local DNS server
`dns.poly.edu`

TLD DNS server

authoritative DNS server
`dns.cs.umass.edu`

requesting host
`cis.poly.edu`

`gaia.cs.umass.edu`

Caching

- DNS responses are cached
  - Quick response for repeated translations
  - Useful for finding servers as well as addresses
- DNS negative queries are cached
  - Save time for nonexistent sites, e.g. misspelling
- Cache entries disappear after some time
  - Cached data periodically times out
  - Lifetime (TTL) of data controlled
    - Low TTL values reduces inconsistencies, allows for dynamic mappings
    - Large TTL values reduce network and server load

DNS Record Format

- DNS records are called resource records and have the FOLLOWING FIELS:

  name, value, type, ttl

- There are four types of record:

- A
  - name is hostname
  - value is IP address
- NS
  - name is domain (e.g. foo.com)
  - value is IP address of authoritative name server for this domain
- CNAME
  - name is alias name for some “canonical” (the real) name www.ibm.com is really servereast.backup2.ibm.com
  - value is canonical name
- MX
  - value is name of mail server associated with name