

Internet Engineering

240-362

Robert Elz

kre@munnari.OZ.AU
kre@coe.psu.ac.th

<http://fivedots.coe.psu.ac.th/~kre>

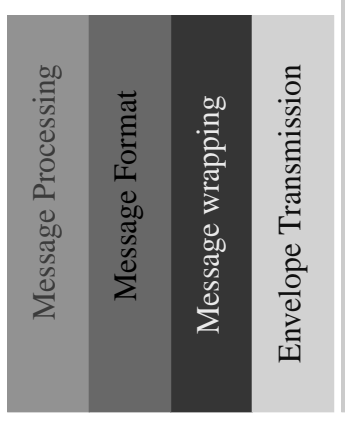
E-mail

- ◇ One of the oldest network applications
 - After Remote Login & File Transfer
- ◇ Network designed to allow remote access
 - Remote login a requirement
- ◇ To work on remote system
 - Need to transfer data
 - Input data for program
 - Output to be examined/printed
 - File Transfer a requirement
- ◇ With File Transfer Protocol (FTP)
 - Can send file to remote system
 - there deliver to user
 - ~~rather than store in named location~~

Design of E-Mail

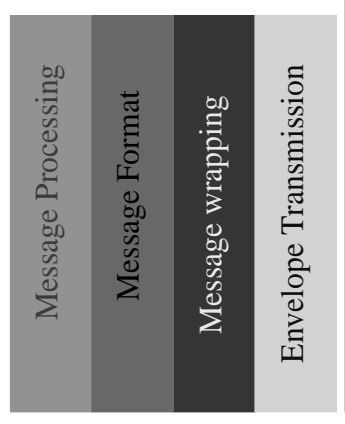
- ◇ Models Office memorandum
 - Message from one person to another
 - Semi-formal setting
 - Rules for format
 - Rules for processing
- ◇ Not a model of letters
 - Not ordinary postal mail
 - Not structured enough
 - Too hard to automate processing

E-Mail protocol layering



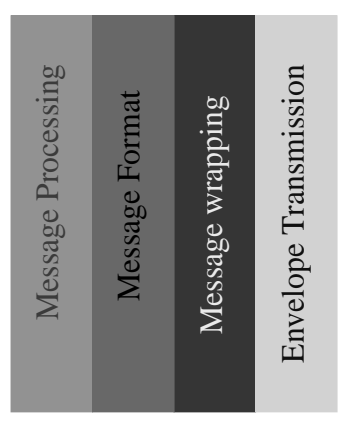
- ◇ Multiple layers of protocol
 - All in the application layer
 - Above the sockets interface

Message Processing



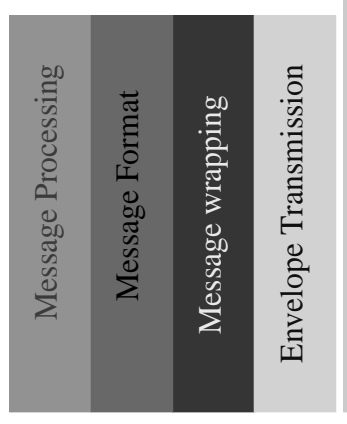
- ◇ Rules/Procedures how to handle messages
 - When to send replies
 - Where to send replies
 - How important message is
 - (and more)

Message Format



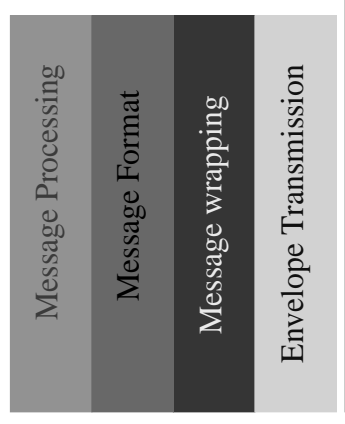
- ◇ Specification of message format
 - Useful so software can assist humans
 - Also helps humans
 - ▷ Less ambiguity as to meaning

Message Wrapping



- ◇ How to put message in envelope
 - Rules for writing envelopes
 - What kind of message fits
 - What envelope means when received
- ◇ Getting message out of envelope

Envelope Transmission

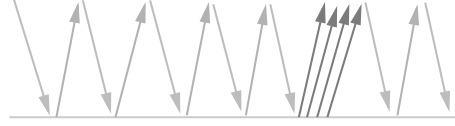


- ◇ How to send envelope to recipient
 - Envelope containing message
- ◇ What to do with messages that
 - cannot be delivered
- ◇ How to do that

Simple Mail Transfer Protocol

- ◇ Protocol between MTAs
 - Mail Transfer Agent
 - Also often used for message submission
 - ▷ From MUA to MTA
 - Mail User Agent
- ◇ Aim is reliable mail transfer
 - At least one MTA is always responsible
 - Whatever happens
- ◇ Very Simple protocol
 - Startup dialog
 - Identify sender
 - ▷ The envelope sender (becomes Return-Path)
 - Identify recipients

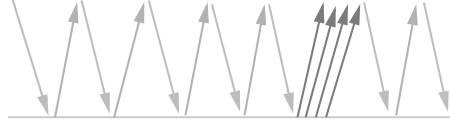
Passing Responsibility



Greeting
Hello, my name is ...
Nice to meet you
I have mail from ...
Sender is OK
The recipient is ...
Recipient address is OK
Here is the message
Please send it now
The message
including headers
I have the message
Goodbye
Goodbye

- Examine which system
 - is responsible for safety of e-mail

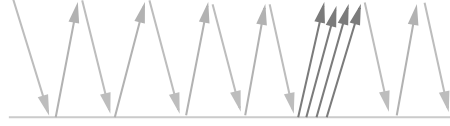
Sender responsible



Greeting
Hello, my name is ...
Nice to meet you
I have mail from ...
Sender is OK
The recipient is ...
Recipient address is OK
Here is the message
Please send it now
The message
including headers
I have the message
Goodbye
Goodbye

- ◇ Before message is sent
 - Sender must be responsible
 - Recipient has no data yet

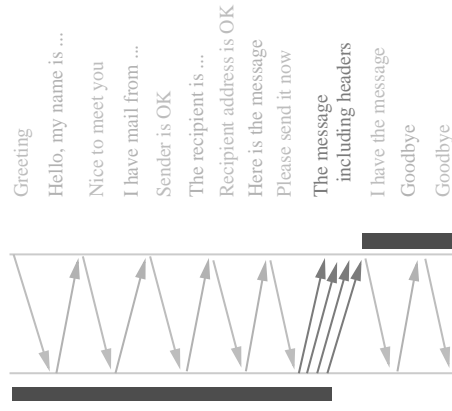
Recipient Responsible



Greeting
Hello, my name is ...
Nice to meet you
I have mail from ...
Sender is OK
The recipient is ...
Recipient address is OK
Here is the message
Please send it now
The message
including headers
I have the message
Goodbye
Goodbye

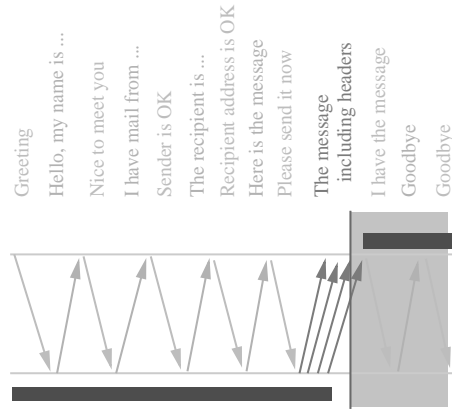
- ◇ After message is fully received
 - Recipient can become responsible
 - Now has complete message

Sender letting go



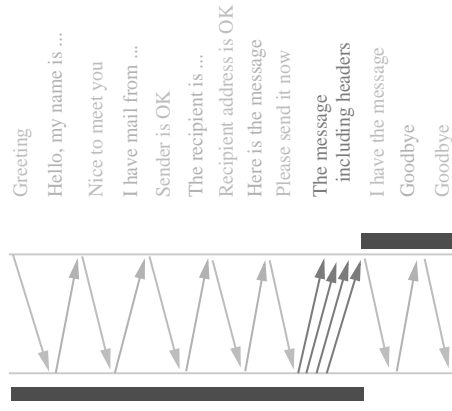
- ◇ But when does sender responsibility end?
 - When end of message is transmitted?

Sender letting go (2)



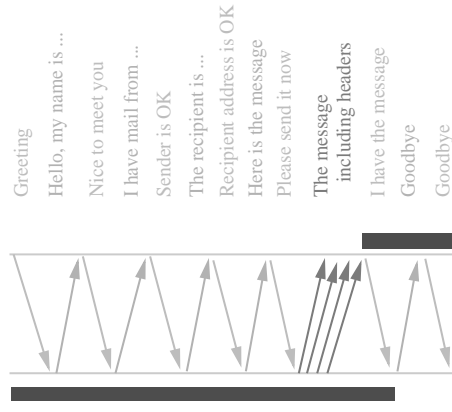
- ◇ What happens if system crashes here?
 - After end of message is transmitted
 - Before it arrives at recipient

Sender letting go (3)



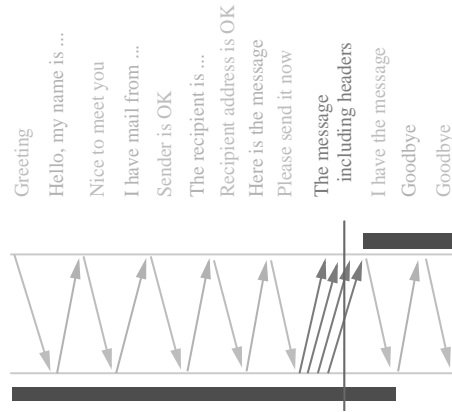
- ◇ Really want this
 - Sender responsible until recipient takes over
 - But how does sender know when that is?

Sender letting go (4)



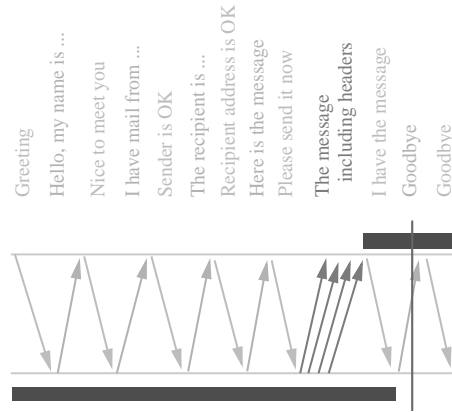
- ◇ **Must wait until sender knows**
 - That recipient is responsible for the data
 - After sender receives message

System crashes



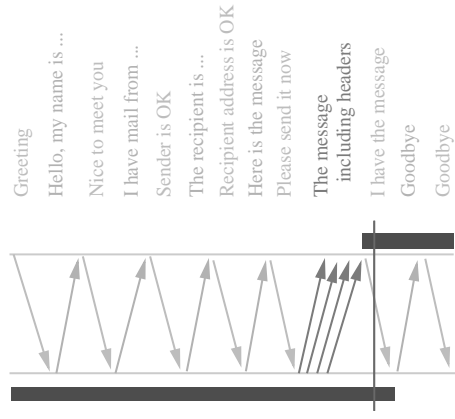
- ◇ **If system crashes here**
 - Sender still responsible
 - Retains the message

System crashes (2)



- ◇ **If system crashes here**
 - Recipient has the message
 - ▷ Can deliver now or later

System crashes (3)



- ◇ If system crashes here
 - Both systems are responsible
 - Recipient delivers message

Delivery Address

- ◇ Mail to
 - user@domain.name
- ◇ We know how to deliver it
- ◇ We know the sender MTA
 - The one that currently has the message
- ◇ What system is the destination MTA?
- ◇ Is it domain.name ?
- ◇ Consider user@psu.ac.th
 - What system is psu.ac.th ?
 - There isn't one!

◇ ~~Mt+ret-all-e-mail~~

Mail Exchanges

- ◇ Mail Exchange
 - System that takes mail for a domain
 - Delivers to all users in that domain
 - directly or indirectly
- ◇ Need method to find Mail Exchange for domain
 - As well as address for hostname
- ◇ Domain Name System
 - Translation from domain name to
 - Address (v4 or v6)
 - Mail Exchange
 - Geographic Location
 - System & OS type

Network Requirements

- ◇ **Reliable Communications**
 - Nothing lost
 - Nothing added
 - Nothing out of order
 - Not arriving too quickly

- ◇ **Clean Termination**
 - Last data always arrives
 - End indication follows last data

- ◇ **Transport Layer**
 - TCP
 - ▷ Transmission Control Protocol
 - Next Week