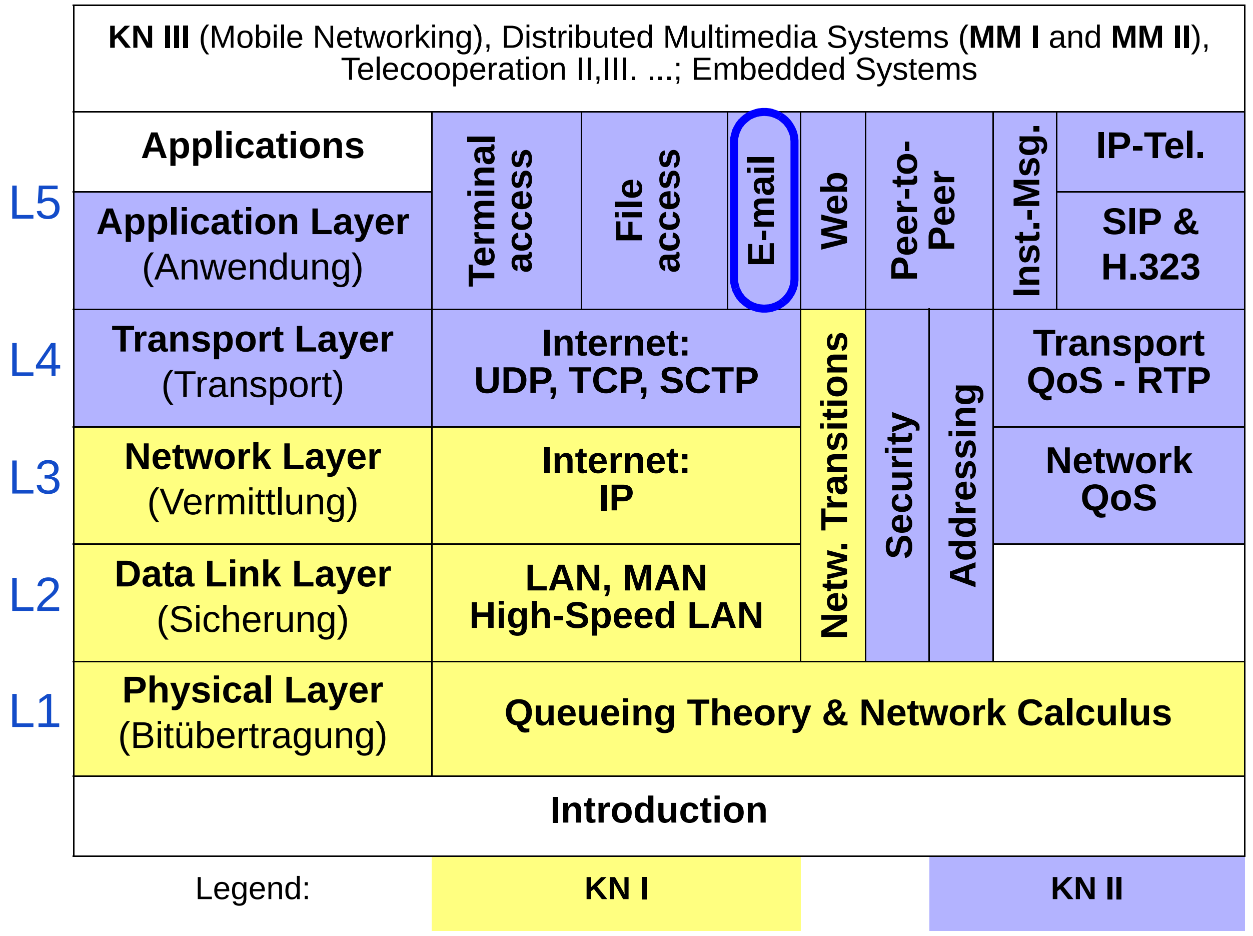
Communication Networks II
Network Applications - Electronic Mail

Prof. Dr.-Ing. **Ralf Steinmetz**

TU Darmstadt - Technische Universität Darmstadt,
Dept. of Electrical Engineering and Information Technology, Dept. of Computer Science
KOM - Multimedia Communications Lab
Merckstr. 25, D-64283 Darmstadt, Germany, Ralf.Steinmetz@KOM.tu-darmstadt.de
Tel.+49 6151 166151, Fax. +49 6151 166152

httc - Hessian Telemedia Technology Competence-Center e.V
Merckstr. 25, D-64283 Darmstadt, Ralf.Steinmetz@httc.de

Scope


KN III (Mobile Networking), Distributed Multimedia Systems (**MM I** and **MM II**), Telecooperation II,III. ...; Embedded Systems
L5
Applications
Application Layer (Anwendung)
Terminal access
File access
E-mail
Web
Peer-to-Peer
Inst.-Msg.
IP-Tel.
SIP & H.323
L4
Transport Layer (Transport)
Internet: UDP, TCP, SCTP
Netw. Transitions
Security
Addressing
Transport QoS - RTP
L3
Network Layer (Vermittlung)
Internet: IP
Network QoS
L2
Data Link Layer (Sicherung)
LAN, MAN High-Speed LAN
L1
Physical Layer (Bitübertragung)
Queueing Theory & Network Calculus
Introduction
Legend:
KN I
KN II

Overview

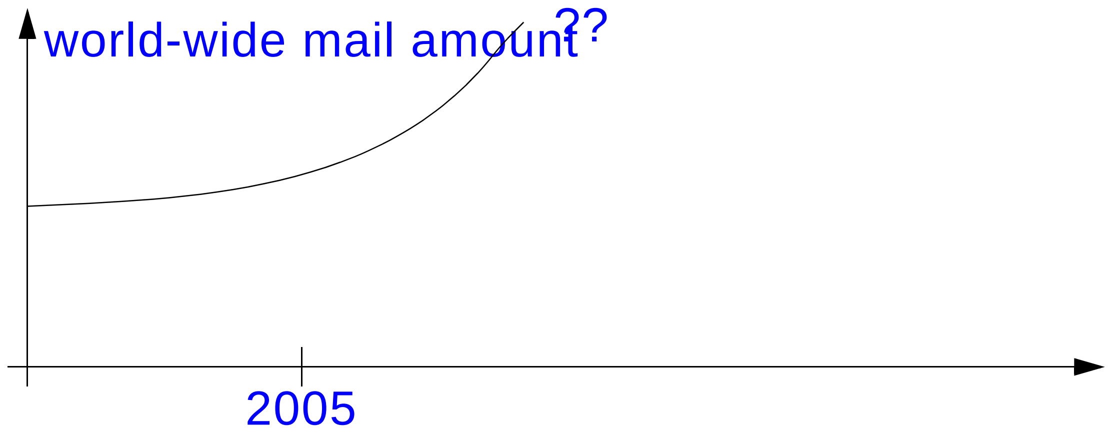
1. Motivation, History and EMail-Address

Function

- **"open memo"**
- **as in regular correspondence**

Some remarks

- **informal way to communicate**
- **cheap**
- **quantity typically approx. 5-60 e-mails per day (without spam)**
  - in business well established
  - at home well established
  - at many countries well established



History

1972

- **first e-mail sent between 2 systems**
- **Ray Tomlinson**
  - Question: Which was the first email message ever sent?
  - Answer: “QWERTYUIOP”
  - Ray Tomlinson sent it to himself…
    - he left MIT to join BBN, Boston, USA

e-mail was THE application of the internet

- **until the web was introduced**
- **and, more recently**
  - peer-to-peer communication is in place

Users

• **until 1990:**	**universities, research**
• **until 2000:**	**companies, usually first within the engineering departments**
• **today:**	**everybody**

Email Address

Electronic mailbox

- **person/addressee is assigned to an electronic mailbox**
- **address' form is "MAILBOX@COMPUTER"**
  - unique
  - split in
    - "**MAILBOX**":
      Mailbox name assigned only locally
      in accordance with the respective local conventions

  @ at

    - "**COMPUTER**"
      for file transfer between systems
- **address today in Internet is usually "MAILBOX@DOMAINNAME"**

  @ at

  - "**DOMAINNAME**"
    - name of the destination domain
    - "domainname" is assigned the appropriate "computer"
      by being entered into the MX-record (MX = Mail eXchange)
      of the domain's DNS server

2. Simple Mail Transfer Protocol SMTP

Simple Mail Transfer Protocol SMTP

a protocol for sending e-mail messages between servers

- SMTP is also used to send messages from a mail client to a mail server

consists of

1. **message format (ASCII presentation)**
   - in 1982 defined in RFC 822
   - how the messages are structured

2. **data transfer protocol (ASCII presentation)**
   - in 1982 defined in RFC 821
   - how the messages are transferred

2.1 SMTP - Message Format & Structure

Defined in RFC 822

Messages consist out of:

- **an envelope; defined in RFC 821**
- **SMTP commands:**
  - HELO, MAIL, RCPT, DATA, QUIT,...
- **header fields (see the following table)**
- **one blank line**
- **message text**
  - originally only 7 bit, i.e. 0-127
  - (extension see also MIME)

SMTP - Message Format & Structure: Header Fields (2)

Header Field	Meaning
To:	Recipient’s email address (several addresses may be given).
Cc:	Carbon Copy. Email address of second recipient (several addresses may be given).
Bcc:	Blind Carbon Copy. Email address of recipients not supposed to be visible to the other recipients (deleted before delivery).
From:	Originator of the message.
Sender:	Sender of the message.
Received:	Displays the route a message has followed until then. A new line is added for each transfer agent.
Return-Path:	May be used to list a path back to the sender.

• difference To: and Cc:	solely psychologically
• difference Cc: and Bcc:	bcc line will be removed from the message and is thus not visible for the recipient
• Sender: and From:	if these are one & the same, then sender omitted
• Return-Path:	optional

SMTP - Message Format & Structure: Other Optional Header Fields (3)

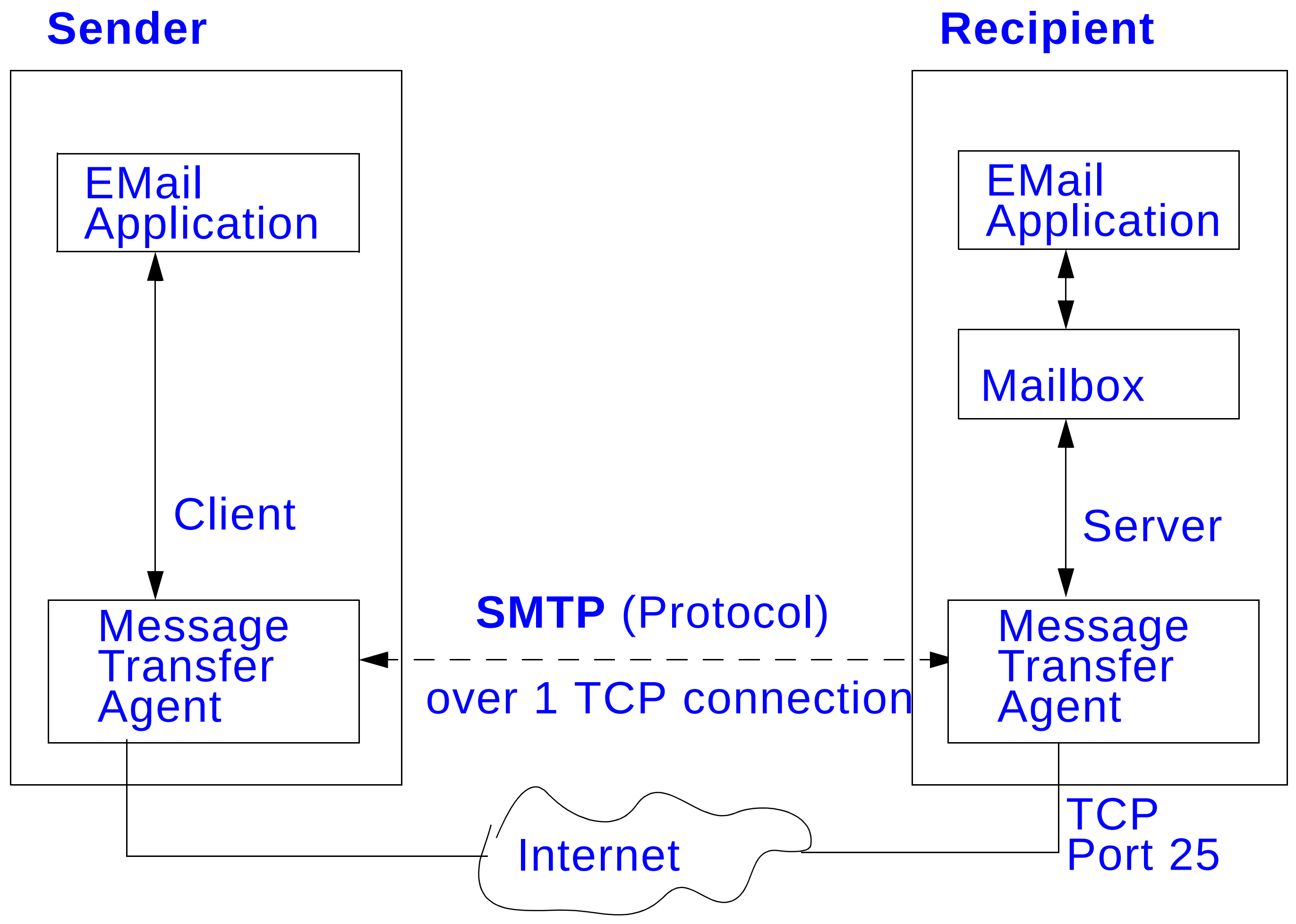
Header Field	Meaning
Date:	Day and time when message was sent.
Reply-To:	EMail address to which the response is to be sent.
Message-Id:	Unique number by which the message may be identified.
In-Reply-To:	Id of the message to which this message is a reply.
References:	Other relevant message Ids.
Keywords:	User defined keywords.
Subject:	Short summary of the contents.

Based on RFC 822, additional (later defined) fields

- **may be defined**
- **these fields have to start with X**
- **examples:**
  - X-No-Archive:
  - X-Auth:
  - X-SPAM:

2.2 SMTP - Data/Mail Transmission

e.g. simple example (no hop inbetween)



SMTP - Data/Mail Transmission (2)

Steps

1. **sender: application**
   - generates the message in the correct format
     (often also the "mail user agent")
   - may store a copy of the message that was sent
2. **sender: transmission program**
   - distributes a copy of each message to each recipient
   - e.g. "sendmail" in UNIX systems
3. **receiver: email server**
   - receives message and files it in the appropriate mailbox
4. **receiver: application**
   - reads mailbox
     - makes e.g. use of POP, IMAP protocols
   - converts the messages into an adequate presentation

Transfer protocol (RFC 821)

- in the internet email is transfered over a TCP connection to Port 25

Transfer Over Several MTAs

i.e. route sender to receiver

- **over several Mail Transfer Agents (MTA)**

SMTP uses the store-and-forward principle to transfer messages

- **identifies the sender**
- **verifies if receiver’s mailbox exists**

system name not always known, but domain is

- **address usually “mailbox@domainname”**
- **domain name server**
  - resource records:
    - information entered about the systems
  - among others that is Mail eXchange Record (MX-Record) with
    - information about preferred system nodes for accepting mail
    - i.e. possibly different systems with different priorities

2.3 SMTP Characteristics

Characteristics

- **all transfered characters are 7 bit ASCII**
- **commands consist out of 4 letters**
- **forwarding option**
- **mailing list administration**
- **receiver confirms command with numerical value**

Example:

```
HELO mysystem.org                              (establish contact)
  250 flute.kom.tu-darmstadt.de Hello ...
```

Problems:

- **initial issue: message length limited to 64KB (in older versions)**
- **if sender and receiver have different timeouts**
  - it may result in misunderstandings
- **"mailstorms" may occur**
  - for example because mailing lists refer to each other

Improvements on some of the above mentiones SMTP problems

- **ESMTP (extended SMTP), defined innitially in RFC 1425**
- **differentiation by contacting (same syntac as HELO)**

```
EHLO <systemname>
```

2.4 SMTP: Example Protocol of Direct Interaction

```
[saxophon] >TELNET TUBA 25
   Trying 130.83.139.132...
   Connected to tuba.kom.tu-darmstadt.de.
   Escape character is '^]'.
   220 mailserver.KOM.tu-darmstadt.de ESMTP
    Sendmail 8.12.6/8.12.6; Mon, 9 Dec 2002 13:58:09
    +0100 (MET)

HELO TUBA.KOM.TU-DARMSTADT.DE
   250 mailserver.KOM.tu-darmstadt.de Hello
   saxophon.kom.tu-darmstadt.de
    [130.83.139.133], pleased to meet you

MAIL FROM: <RALF.ACKERMANN@SAXOPHON>
   250 <ralf.ackermann@saxophon>... Sender ok

RCPT TO: <BAUMANN>
   250 <baumann>... Recipient ok

DATA:
   500 Command unrecognized
```

SMTP: Example Protocol of Direct Interaction (2)

```
DATA
   354 Enter mail, end with "." on a line by itself

TESTMAIL
THIS MAIL TESTS THE MAIL SYSTEM
.
   250 OAA20896 Message accepted for delivery

QUIT
   221 mailserver.KOM.tu-darmstadt.de closing
   connection

[Connection closed by foreign host.]
[saxophon]~ >
```

2.5 SMTP: Example Messages

Example of sent message:

```
From rst Fri Jan 17 08:34:50 2003
Subject: Lecture CN II
To: eveking@maigret.rs.tu-darmstadt.de (H. Eveking)
Date: Sat, 18 Jan 2003 17:48:50 +0100 (MET)
Cc: monika.jayme@kom.tu-darmstadt.de (Monika Jayme)
Cc: jan.baum@kom.tu-darmstadt.de (Jan Baum)
X-Mailer: ELM [version 2.4 PL25]
MIME-Version: 1.0
Content-Type: text/plain; charset=ISO-8859-1
Content-Transfer-Encoding: 8bit
Content-Length: 1139

The second exercise re. CN II is ambiguous:
 ..
best regards Ralf
```

SMTP: Example of Received Message

Example of received message:

```
From eveking@maigret.rs.tu-darmstadt.de  Fri Jan 17
10:30:32 2003
X-UIDL: ee1a889ea7fece3665d9aaeaa3c558c4
Return-Path: eveking@maigret.rs.tu-darmstadt.de
Received: from KOM.tu-darmstadt.de by
mailserver.KOM.tu-darmstadt.de (8.12.6/8.12.6) with
ESMTP id KAA01703 for <Ralf.Steinmetz@KOM.tu-
darmstadt.de>; Fri, 17 Jan 2003 10:30:30 +0100 (MET)
Received: from mailhost.rs.TU-Darmstadt.DE by
gatekeeper (8.12.6/8.12.6) with ESMTP id KAA26173 for
<Ralf.Steinmetz@KOM.tu-darmstadt.de>; Fri, 17 Jan 2003
10:26:48 +0100 (CET)
Received: from maigret.rs.TU-Darmstadt.DE (maigret
[130.83.34.40]) by mailhost.rs.TU-Darmstadt.DE
(8.12.6/8.12.6) with SMTP id KAA28568
for <Ralf.Steinmetz@KOM.tu-darmstadt.de>; Fri, 17 Jan
2003 10:30:30
+0100 (MET)
```

```
Received: by maigret.rs.TU-Darmstadt.DE (5.x/SMI-SVR4)
id AA04555; Fri, 17 Jan 2003 10:30:28 +0100
Date: Fri, 17 Jan 2003 10:30:28 +0100
From: eveking@maigret.rs.tu-darmstadt.de (H. Eveking)
Message-Id: <9801160930.AA04555@maigret.rs.TU-
Darmstadt.DE>
To: Ralf.Steinmetz@KOM.tu-darmstadt.de
Subject: Re: Lecture CN II
X-Sun-Charset: US-ASCII
Status: OR

May even be an error.
```

2.6 Electronic Mail: Critical Issues of Classical SMTP

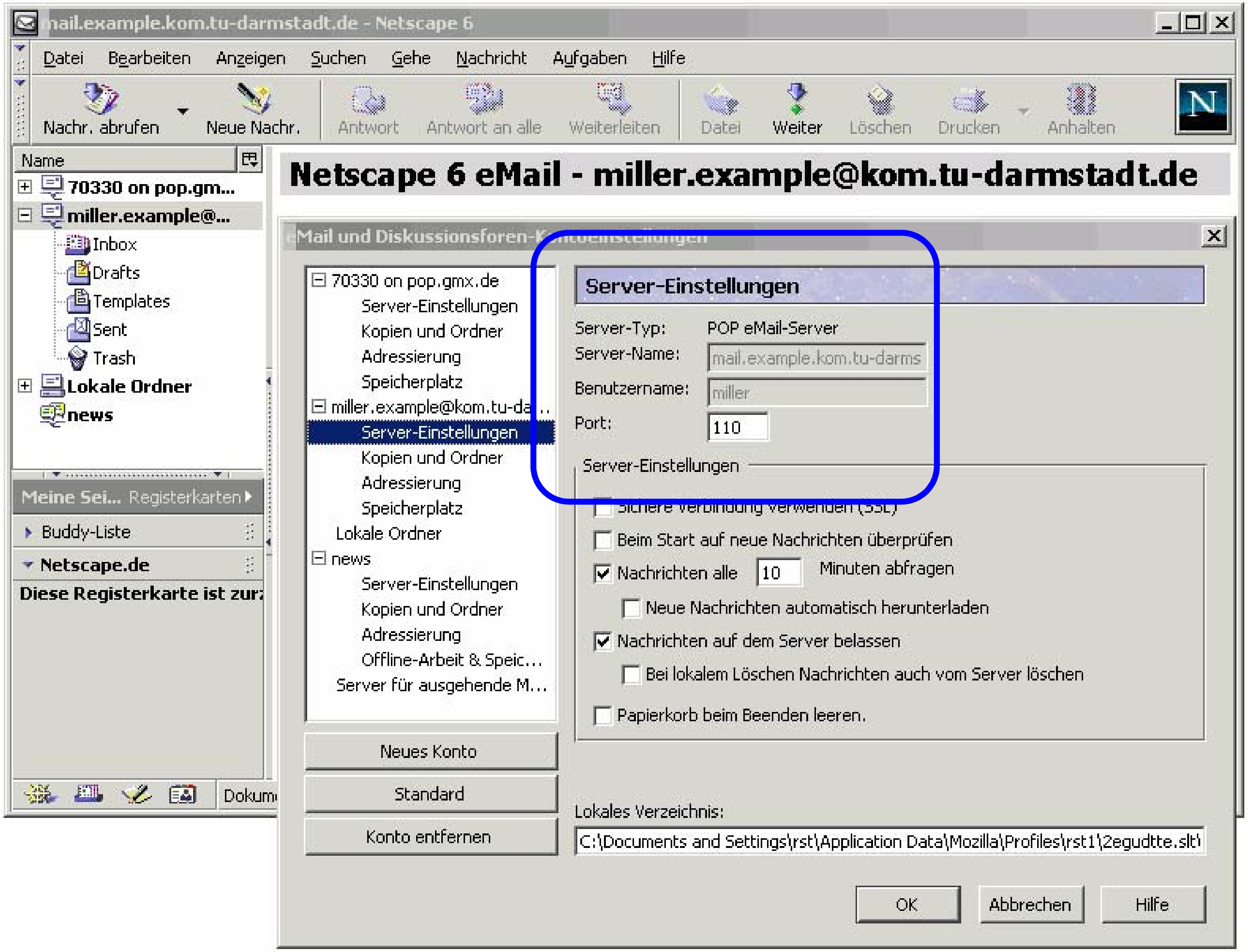
With SMTP and original message format

- **sending a message to various recipients**
  - done by sending the same data to all of them individually
- **messages do not have an internal structure**

Makes automatic processing difficult

- **no acknowledge: sender does not know**
  - if the message he sent has actually been received by the recipient
- **message rerouting arduous ("mühsam")**
- **user interface not integrated in transfer system**
- **no way to send message containing a mixture of text, graphics and audio**
- **messages may contain ASCII characters only**
  - no accents or special characters ä,ö.ü, etc.(e.g. French, German)
  - no non-latin alphabets
    - e.g. Hebraic
  - no possibility to present languages that are not bound by an alphabet
    - e.g. Chinese, Japanese

3. Post Office Protocol / Interactive Mail Access Protocol



Server-Einstellungen

Server-Typ: POP eMail-Server
Server-Name: mail.example.kom.tu-darms
Benutzername: miller
Port: 110

Server-Einstellungen
- [ ] Sichere Verbindung verwenden (SSL)
- [ ] Beim Start auf neue Nachrichten überprüfen
- [✓] Nachrichten alle 10 Minuten abfragen
  - [ ] Neue Nachrichten automatisch herunterladen
- [✓] Nachrichten auf dem Server belassen
  - [ ] Bei lokalem Löschen Nachrichten auch vom Server löschen
- [ ] Papierkorb beim Beenden leeren.

Lokales Verzeichnis:
C:\Documents and Settings\rst\Application Data\Mozilla\Profiles\rst1\2egudtte.slt\

3.1 Post Office Protocol

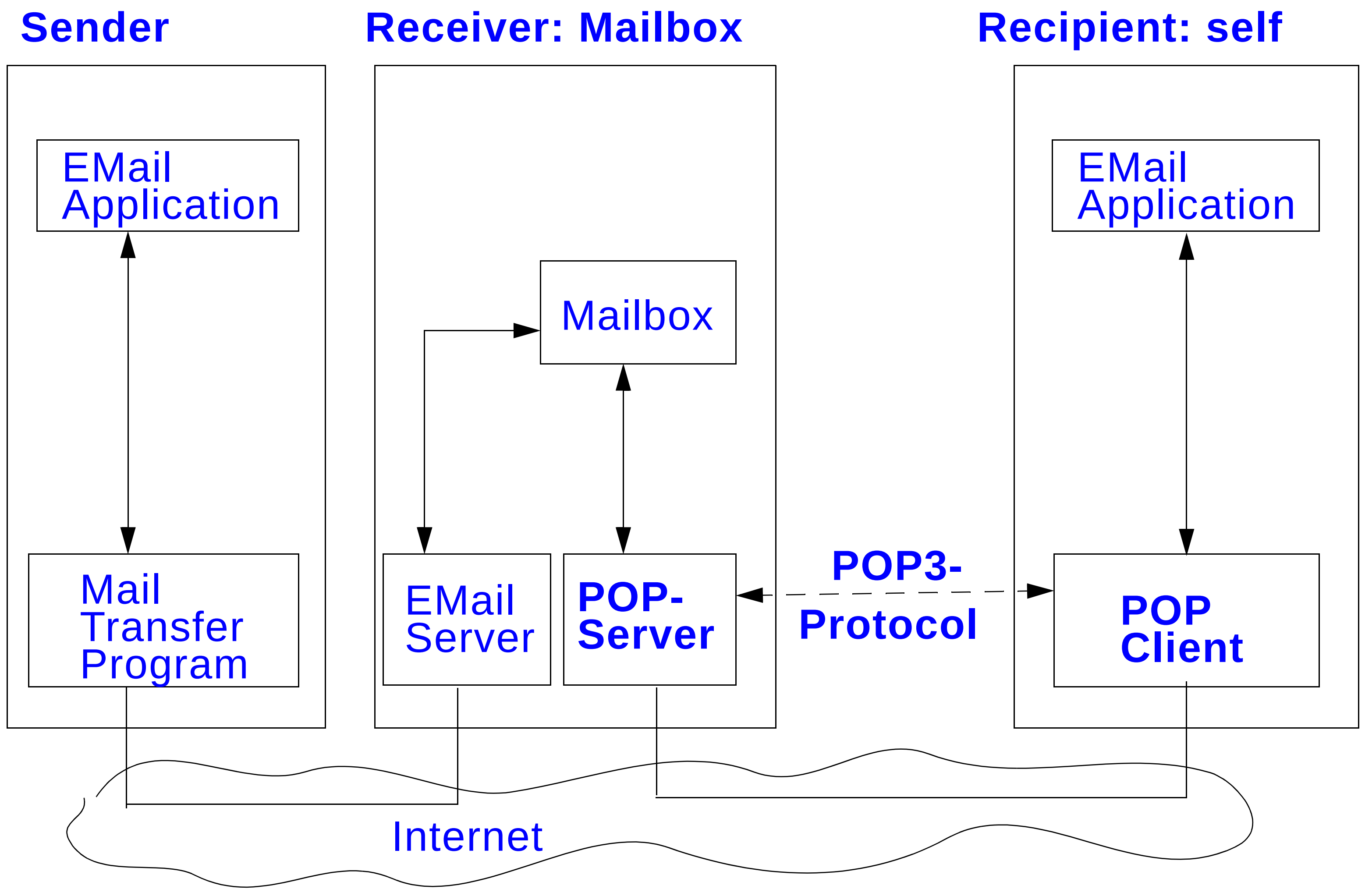
Internet Message Access Protocol is

a protocol used to retrieve e-mail from a mail server
- defined in RFC 1225, 1939, 2449

Motivation
- **user (mail reipient) uses different systems**
  - but his mailbox should always be the same
- **server has to run reliably for 24 hours**
  - but not necessarily his system
- **mailbox and applications**
  - often on different systems

Protocol for remote mailbox access:
- **user (usually) transfers mail for further processing**
  - to his local system
- **this transfer is defined in a protocol: Post Office Protocol (POP)**
- **characteristics**
  - access permitted only after authentication
  - can provide information about contents without actually transfering them
  - port
    - uses Port 110
    - SSL encrypted Port 995

POP Interaction



3.2 Interactive Mail Access Protocol (IMAP)

IMAP: Interactive Mail Access Protocol

a protocol used to retrieve e-mail from a mail server, alternatively to POP

- RFC 1056

Motivation

- **electronic letters remain on the server**
- **that means that server management is necessary**

characteristics

- **port**
  - port 143
  - SSL encrypted Port 993
- **security problem**
  - access to server data
  - possible actions: copy, delete, move

4. Multipurpose Internet Mail Extensions (MIME)

Defined in RFC 1341 and RFC 1521

Possibilities:

- **messages may contain non ASCII character**
  - accents or special characters ä,ö.ü, etc.(e.g. French, German)
  - non-latin alphabets
    - e.g. Hebraic
  - languages that are not bound by an alphabet
    - e.g. Chinese, Japanese
- **messages that may contain audio data, video data or general data**

Idea:

- **using the format defined in RFC 822 for messages**
- **define a structure for the message text**
- **define rules for coding non-ASCII messages**

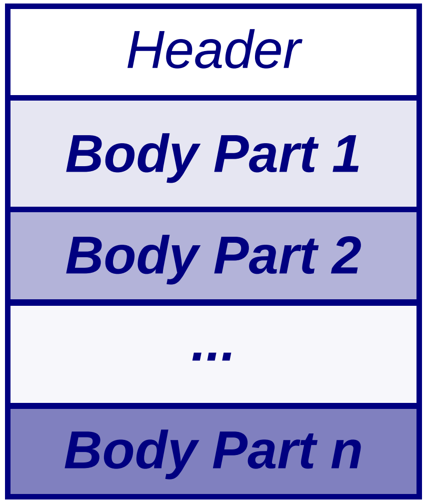
$\Rightarrow$ **only programs for generating and displaying messages to be modified**

$\Rightarrow$ **Programs for sending and receiving remain unmodified**

4.1 MIME Messages

Chosen approach:

- **MIME messages consist of multiple parts**
- **Each part may have a different type: text, audio, image, ...**



Content types:

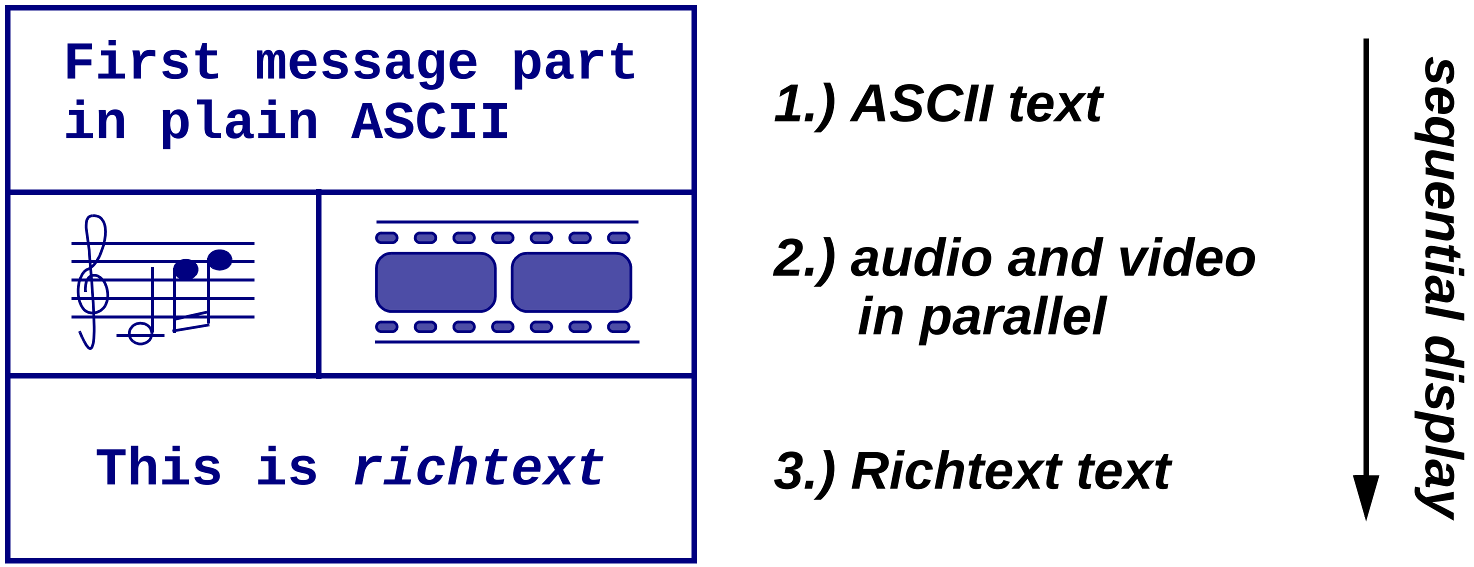
- **Text(subtypes: plain, richtext)**
- **Image(subtypes: gif, jpeg)**
- **Audio(subtypes: basic)**
- **Video(subtypes: mpeg, h261)**
- **Message(subtypes: partial, external-body)**
- **Multipart(subtypes: mixed, alternative, parallel)**
- **Application(subtypes: postscript, oda)**

Subtypes:

- **Additional subtypes can be registered**
- **Designated subtypes for private usage**

MIME Message: Example

Structure of an example message:



MIME message must include

- Data in multiple message parts
- Definition of content types of individual parts
- Boundaries between parts

MIME Message: Example (cont.)

```
...
Content-type: multipart/mixed;
boundary=unique-boundary-1
--unique-boundary-1
Content-type: text/plain
First message part in plain ASCII.
--unique-boundary-1
Content-type: multipart/parallel;
boundary=unique-boundary-2
--unique-boundary-2
Content-Type: audio-basic
Content-Transfer-Encoding: base64
... base64-encoded audio data goes here ...
--unique-boundary-2
Content-Type: image/jpeg
Content-Transfer-Encoding: base64
... base64-encoded image data goes here ...
--unique-boundary-2--
--unique-boundary-1
Content-Type: text/richtext
This is <italic>richtext.</italic>
--unique-boundary-1--
```

- Boundaries between message parts
- Definition of content types
- Data

4.2 MIME: Header Fields

Header Field	Meaning
MIME-Version:	Identifies the MIME version
Content-Description:	Legible description of the message
Content-Id:	Unique number to identify the message
Content-Transfer-Encoding:	Encoding type
Content-Type:	Message type

MIME version:

- **is necessary to identify the message as a MIME message**
- **example:**

```
MIME-Version: 1.0
```

Content description:

- **example:**

```
Content-Description: A picture of my guinea pig
```

Header Fields: Content-Transfer-Encoding

Content-Transfer-Encoding in 5 different types:

Type	Way
ASCII text	7-bit ASCII
ASCII text with 8 bit	8-bit ASCII violates protocol specification
binary	any desired 8-bit violates protocol specification
quoted-printable	ASCII presentation for short 8-bit information
base64 (ASCII armor)	ASCII presentation for 8 bit information

e.g. quoted-printable:

- **7-bit ASCII**
- **all characters > 127:**
    - presented as XXh
    - with XXh as a hexadecimal number representing the character

Header Fields: Content-Transfer-Encoding: base64 (2)

e.g. base64:

- **information viewed as a data stream**
- **64 characters are used (i.e. $2^6=64$)**
  - = has special function, i.e.
    - == last group contained only 8 bits
    - = last group contained only 16 bits
- **3 bytes which need to be coded (24 Bit) are divided into four 6-bit groups**
- **line breaks are ignored**

example

- **.. next slide**

MIME: Examples (2)

example base 64

```
Content-type: application/msword; name="A000001.doc"
Content-Disposition: attachment; filename=A000001.doc
Content-transfer-encoding: base64
```

```
0M8R4KGxGuEAAAAAAAAAAAAAAAAAAAAAPgADAP7/
CQAGAAAAAAAAAAAAAAACAAAAhgAAAA
AAAAAAEAAAiAAAAAEAAAD+////AAAAAIQAAACFAAAA//////////////////////
//////
////////////////////////////////////////////////////////////////
//////
////////////////////////////////////////////////////
spcEAcQAHBAAACBK/
AAAAAAAAEAAAAAAABAAAm0YAAA4AYmpianQrdCsAAAAAAAAAAAAAAAAAAAAAAAAH
BBYAQo
0AABZBAQAWQQEANUIAAAAAAABlAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAD//
w8AAAAAAAAA
AAD//w8AAAAAAAAAAAD//
w8AAAAAAAAAAAAAAAAAAAAAAF0AAAAAAPADAAAAAAAA8AMAAP
ADAAAAAAAA8AMAAAAAAAADwAwAAAAAAAPADAAAAAAAA8AMAAJQAAAAAAAAAAAAAAK
4FAAAA
AAAArgUAAAAAAACuBQAAAAAAAK4FAAD4AAAApgYAAEQAAADqBgAAhAAAAK4FAAAA
AAAAoT
```

Header Fields: Content-Type

Examples

Type	Subtype	Description
Text	Plain	Unformatted text
	Richtext	Text with simple formatting commands in SGML
Image	Gif	Image in GIF format
	Jpg	Image in JPG format
Audio	Basic	Audio
Video	Mpeg	Video in MPEG format
Applicati on	Octet-Stream	Uninterpreted byte stream
	Postscript	Printable document in Postscript format
Message	Rfc822	A MIME RFC 822 message
	Partial	This message has been split for transmission
	Externalbody	This message has to be retrieved from the network
Multipart	Mixed	Independent parts in the specified order
	Alternative	Same message but different formats
	Parallel	Parts have to be presented parallel
	Digest	Each part is a full RFC 822 message

4.3 MIME: Examples

Example:

```
Content-Type: text/targettext
"I am an <bold>owl </bold>", said the
<italic>walrus</italic>.
```

results in

"I am an **owl**", said the *walrus*.

MIME Example: Sent Text Message

```
From: matthias.hollick@saxophon.kom.tu-darmstadt.de

To: ralf.steinmetz@tuba.kom.tu-darmstadt.de

MIME-Version: 1.0

Message-Id: <199707011607.SAA20302@saxophon.kom.tu-darmstadt.de>

Content-Type: multipart/alternative; boundary= "-----------
1DA8FCD5D4D"

This is a preamble, ignored by the user agent.

-----------1DA8FCD5D4D

Content-Type: text/targettext

   "I am an <bold>owl</bold>", said the <italic>walrus</italic>.

   The marabu nodded <italic>wisely</italic> and said:

   "I am an owl, too!"
```

MIME Example: Sent Audio Message

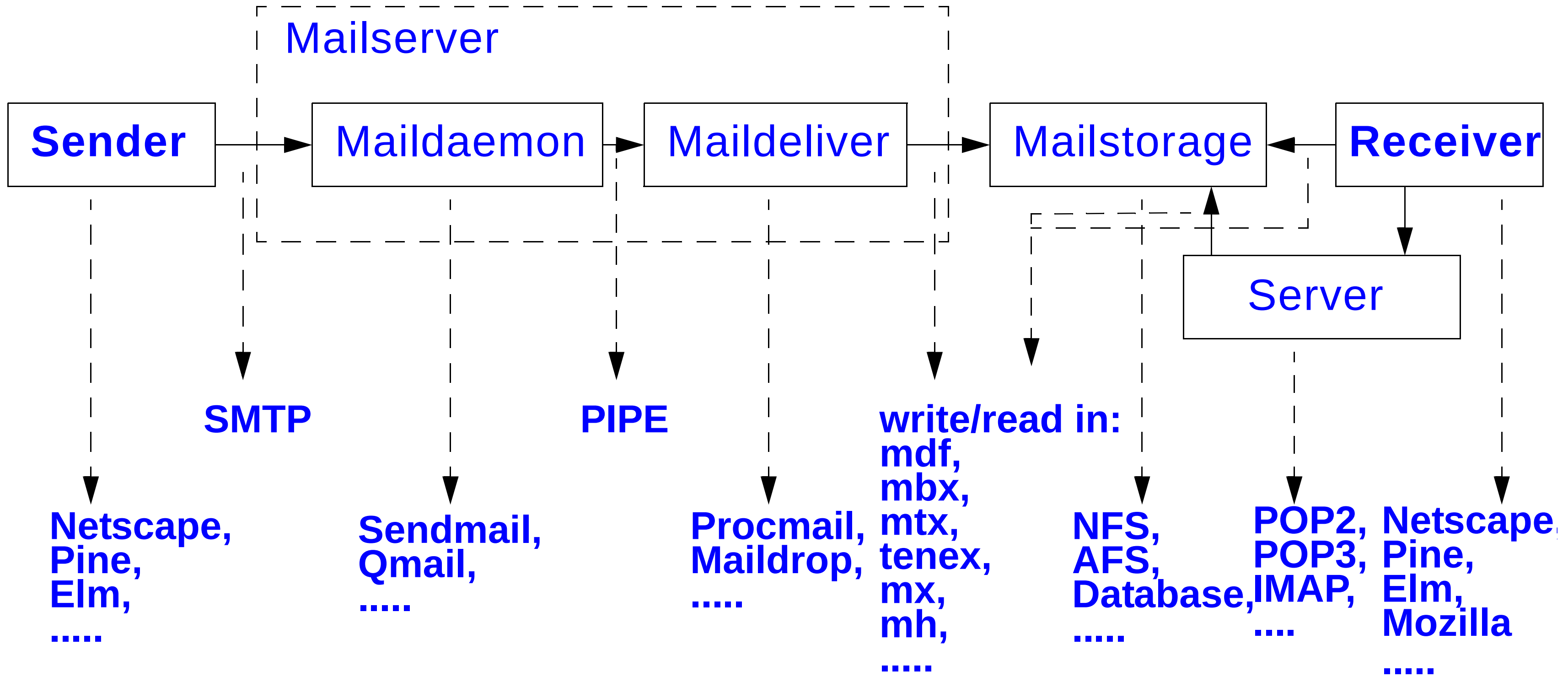
```
From: ralf.steinmetz@tuba.kom.tu-darmstadt.de

To: matthias.hollick@saxophon.kom.tu-darmstadt.de

MIME-Version: 1.0

Message-Id: <199707011607.SAA20302@saxophon.kom.tu-darmstadt.de>

Content-Type: multipart/alternative; boundary= "-----------
1DA8FCD5D4D"

This is the preamble, ignored by the user agent

   -----------1DA8FCD5D4D

   Content-Type: message/external-body;

   access-type="anon-ftp";
   site="ftp.kom.tu-darmstadt.de";
   directory="/pub/eulen";
   name="am_owls_too.snd"

   Content-Type: audio/basic
   content-transfer-encoding: base64
```

5. Further Concepts and Details of Electronic Mail

Topics

- **Implementation issues**
- **History**
  - X.400
- **Other Concepts**
  - References
  - Security

Mail Implementation Overview



Overall mailing process

- **of a daily used environment**

X.400 Mail

History

- **defined 2 years after RFC 821 and RFC 822 (1984)**
- **idea: to correct the disadvantages of the above RFC's**

Supported by:

- **CCITT - ITU**
- **telecommunication corporations, governments, industry**

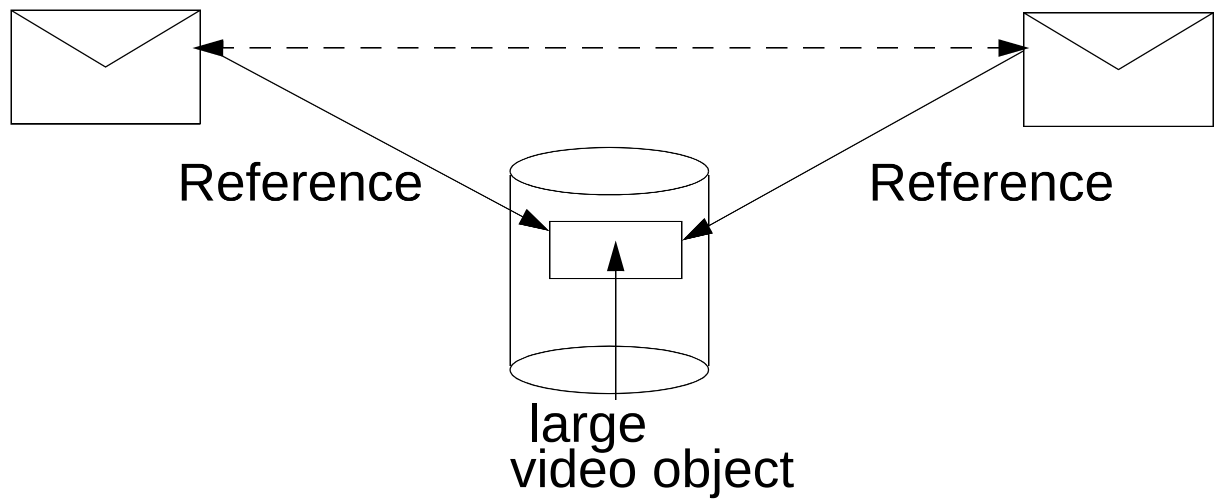
Defacto today: X.400 not very widespread anymore

- **reasons:**
  - poor design
  - extremely complex
  - SMTP had prevailed

Pragmatic decision

- **simple but functioning system (YES) or**
- **beautiful but very complex functioning system**

Referenced Based Mailing



Challenge:

- **many objects have a high amount of data (e.g. video)**
- **receiver has only a limited storage capacity**

Solution: global store

- **can be realized by url**
- **but:**
  - contents may not necessarily be available
- **future**
  - combined content managment systems & workflow environments

Secure Electronic Mail

Motivation

- **ASCII text is easy to read**
  - by e.g. any sniffer
- **is the sender really the one it claims it is?**

S/MIME

- **based on strictly hierarchic certification, X.509 certificates**
  - just like SSL

OpenPGP

- **Open Pretty Good Privacy**
- **based on “web of trust”**
  - user decides which certification entity he can trust