Experience in implementing an e-mail/web gateway

Francesco Gennai, Marina Buzzi, Laura Abba
Istituto per le Applicazioni Telematiche, National Research Council (CNR) - Pisa, Italy
E-mail: {F.Gennai, M.Buzzi, L.Abba}@iat.cnr.it

ABSTRACT

This paper describes our experience in implementing an e-mail/web gateway. The system permits a user to send one or more large documents via e-mail, and at the same time overcomes problems that the receiver may encounter when downloading large size files with e-mail agents. The user sends a document as an attachment to the destination address, on fly the e-mail server extracts and saves the attachments in a web-server disk file and substitutes them with a new message part that includes the URL pointing to the saved document. The receiver can download these large objects by means of a user-friendly browser. This integration between electronic mail and web services is useful for people such as library operators who need to send large files to Internet users and it can be customized in order to satisfy the needs of different organizations.

MOTIVATION

KEYWORDS: e-mail, web, MIME, attachments.

INTRODUCTION

The simplest and most common mechanism for sending a document is the electronic mail system. In electronic mail we can identify two important components of the delivery process:

- The transport protocol SMTP (Simple Mail Transfer Protocol) [Pos 82] [Cro 82] used to send messages. Depending on the network configuration the sent message can pass through one or more mail servers (Message Transfer Agents) to reach the destination mailbox;
- Mailbox access protocols such as POP (POP Office Protocol) [Mye 96] and IMAP (Internet Message Access Protocol) [Cri 96] which enable users to access their own remote mailboxes, using a user-friendly e-mail client.

However, the transfer of large files via e-mail can present some limitations in both transport and mailbox access protocols:

- Mail servers can impose message size limitations. Sometimes problems emerge when delivering large messages over congested networks. A possible solution is to fragment messages at the sender MTA, recomposing them at the destination MTA. Both MTAs must support the MIME (Multipurpose Internet Mail Extensions) message fragmentation/de-fragmentation mechanisms.
- Mailboxes access protocol time-outs. POP offers off-line and pseudo on-line ('leave messages on server' option) operational modes. Full messages (header and body) are downloaded in a sequential order. In transferring large files, time-out parameters set into the POP client can interrupt the connection making it impossible to download the message and those following. In this case the user will not be able to download new messages until the server's administrator removes the blocking message.

  The IMAP protocol offers on-line, off-line and disconnected operational modes. In contrast to POP, IMAP selects remote messages by downloading only mail headers. Users can activate a selective download of the full body message or body parts. This implies that a user using IMAP is able to verify the message structure in order to understand whether it is composed of large parts which he or she can then decide not to download – for example, when using a slow dial-up connection.

  Using IMAP to access remote mailboxes could be a better solution than POP access, because users

download only the mail header, preventing large messages from blocking the access to new e-mail. IMAP is a step ahead of POP in flexibility and features, but nevertheless it does not entirely solve the problem.

- The availability of remote and local resources (such as disk space) can influence the behavior of the Internet Delivery Service.

Our idea is to combine the e-mail service (for document transmission) with the web service (for document downloading). Downloading documents using a web browser is more flexible and reliable than when using one e-mail client. The user can move between different platforms (where a web client is present) without requiring any client reconfiguration (such as that required by e-mail clients).

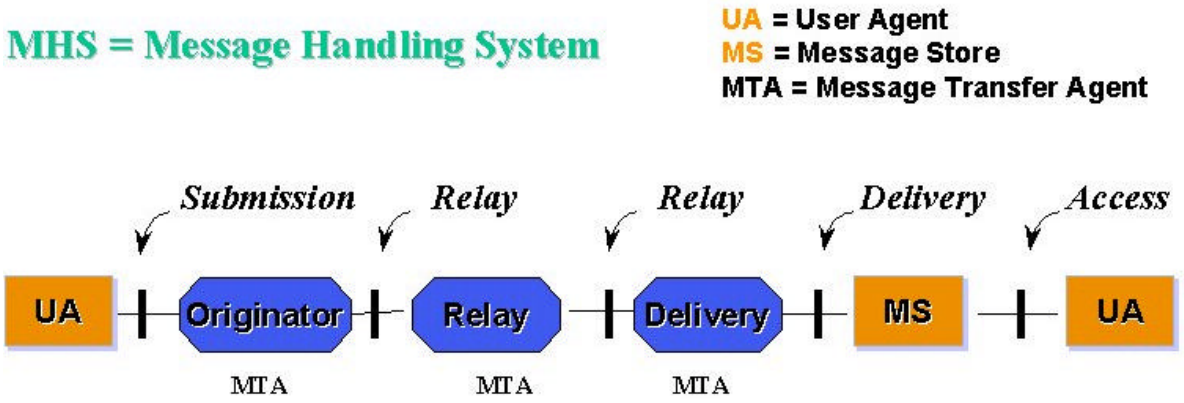

Fig. 1 - Message Handling System (logical scheme)

MIME

The Multipurpose Internet Mail Extensions is the keystone of our service. MIME was conceived as an extension of the electronic mail message format (RFC822) to allow for the inclusion of multimedia data in a standard message format. RFCs 2045 [Fre 96a], 2046 [Fre 96b], 2047 [Moo 96], 2048[Fre 96c], 2049 [Fre 96d] contain the basic definitions and descriptions of MIME mechanisms and message formats. An RFC822 message is composed of two parts: the header and the body. The body is a flat ASCII document with line length limitation. MIME redefines the RFC822 body's format in order to include also non-textual data and structured body parts.

MIME defines data types: text, image, audio, video, application, message and multipart. For each data type more subtypes can be defined (i.e. image/tiff, image/gif, etc.). A multipart MIME message is a structured message composed of multiple parts containing different data types [Fig.2].

A content-type header field is used to specify the media type and subtype of data in the body of a message part. A Content-Transfer-Encoding header field is used to specify both the encoding transformation applied to the body and the domain of the result [2].

MIME is a very powerful and flexible message format. New body formats and content transfer encoding mechanisms could be defined by introducing new content-type and the content-transfer-encoding header fields. Its structure is suitable for automatically intercepting and extracting parts of messages. This feature is potentially applicable to many fields.

```
From: Apple Green <Apple.Green@iat.cnr.it>
Subject: test
To: Rose.PinkCheeks@iat.cnr.it
Message-id: <3AC0983A.7694A88B@iat.cnr.it>
MIME-version: 1.0
X-Mailer: Mozilla 4.7 [en] (Win98; I)
Content-type:multipart/mixed; boundary="Boundary_(ID_LGRYaGz50S9A4OELCpP+zg)"
…

--Boundary_(ID_LGRYaGz50S9A4OELCpP+zg)
Content-type: text/plain; charset=us-ascii
Content-transfer-encoding: 7bit

Dear Rose,
… blah blah blah …

--Boundary_(ID_LGRYaGz50S9A4OELCpP+zg)
Content-type: image/tiff; name="thegateway.tif"
Content-transfer-encoding: base64
Content-disposition: inline; filename="thegateway.tif"

SUkqAE4zAQAyMDAxOjAzOjI3IDEzOjM2OjA1AEgAAAABAAAASAAAAAEAAAAIAAgACAA6AgAA
MgQAAFkGAAD2CAAAOgsAALENAAAdEAAAohIAAGgVAAC2GAAA3hsAANIeAAByIgAAUSYAAA8r
…
MgECABQAAAAIAAAAPQEDAAEAAAACAAAAAAAAAA==

--Boundary_(ID_LGRYaGz50S9A4OELCpP+zg)--
```

Figure 2 - MIME message example

THE SYSTEM

Using a MIME-compliant mail server, we were able to write simple scripts to implement a selective e-mail/web gateway. Figure 3 shows how the system works:

- The user sends a message containing one or more file attachments.
- On fly the e-mail server extracts document types (.doc, .pdf, .gif, .jpg, .tiff, etc.) contained in MIME message parts and saves them in a web-server disk file.
  The extracted part is replaced with a brief text (including filename, size, type, and the file availability expressed in days) and the link pointing to the saved part.
  Text message parts remain unchanged.
- The resulting multipart message will continue the network trip towards its own final destination.
- The receiver opens the message, clicks on the URLs and downloads the original files from the web server.

We offer multiple services, on the same e-mail system used to implement the gateway. We host e-mail services for different CNR domains, we offer a MIME attachment conversion service, e-mail to fax gateway service. For this reason we implemented a selective service. In order to have messages processed by our system, two conditions must be met:

- the IP address of the client must be registered in a service authorization list;
- the e-mail destination address must end with the fake top level domain (TLD): **.save** This fake TLD domain (added by the user) will be automatically removed by the e-mail server and it is necessary in order to request that the message pass through our service.

The first condition permits only authorized users to use the e-mail/gateway functionality.
The second condition is important in order to permit the user to switch between regular e-mail operations or those which can extract and save attachments.
When the messages match the above conditions, the e-mail server passes each message part and several variables including the MIME part type, MIME part subtype, the MIME Content-type name

parameter (containing file name and extension) to our scripts. The service is applied to any file type except text, which is required to travel with the message.
The script checks the MIME part type/subtype (text/plain, text/html, application/pdf, image/tiff, image/jpeg, image/gif, application/octet-stream, etc). If this value matches text type (whatever the subtype) the part is restored unchanged to its original position inside the message body, otherwise it is stored on the disk and substituted with a text/html part including the link to the saved part. To avoid collisions (that is, same-name files sent by different users) the filename is automatically created by the system joining a unique identifier with the file extension (derived from the Content-type). If the Content-type is application/octet-stream[1] the script extracts the file extension from the Content-type name parameter (if present) in order to save the part with the appropriate extension[2].
Content-type and content-transfer-encoding mime headers are changed to reflect the new part content.

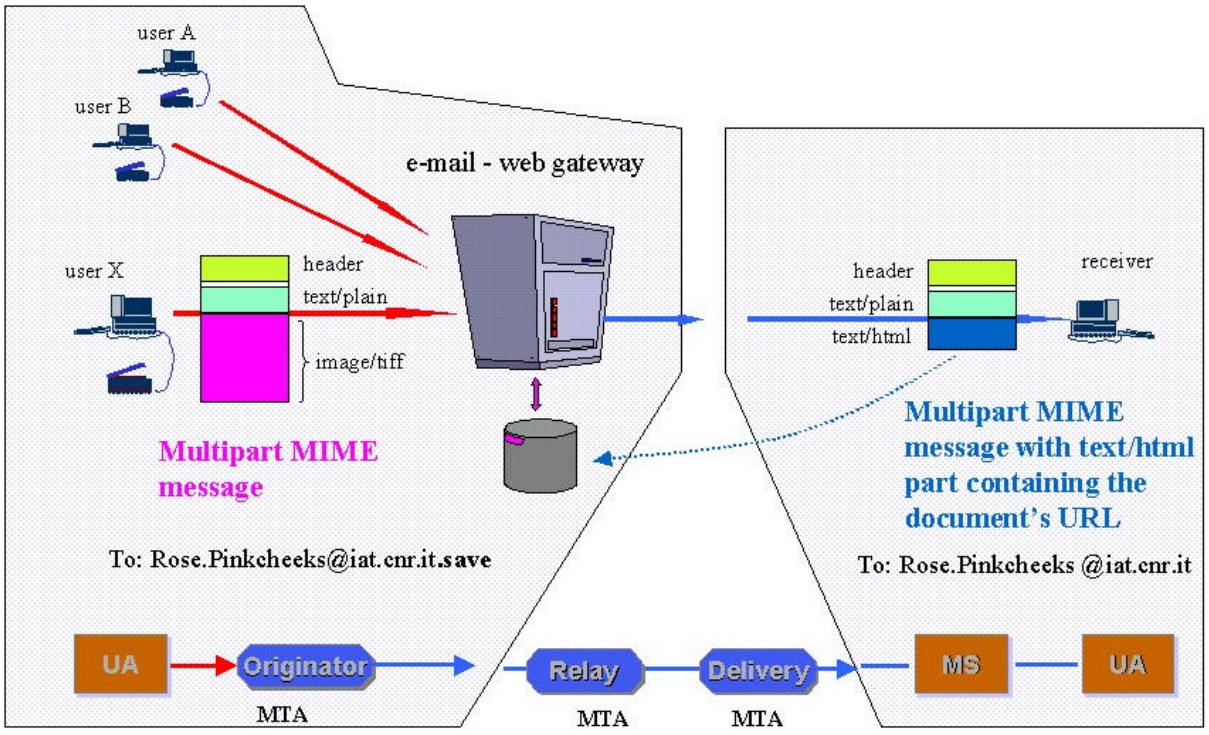

Figure 3 - E-mail/web gateway function

The development environment

The gateway is based on the PMDF mail system and the OSU HTTP server, and runs on a 2-node OpenVMS cluster. The script is developed in Data Command Language (DCL).

Security considerations

---

[1] The application/octect-stream MIME type/subtype should never be present as content type description, for instance, for images such as tif, jpeg and gif or for document files such as doc, pdf, rtf, etc., because their own media data descriptions are described as official MIME types/subtypes. Nevertheless it is known that some misconfigured clients or MIME agents which are imperfectly compliant can generate a generic application/octect-stream content type in the presence of the above data types. Our system is also able to deal with this case, saving the corresponding file, where a name with the file extension is given.

[2] The server HTTP uses the file extension to generate the appropriate Content-type of data transmitted to the client.

At present, documents stored in the web area are not protected. However the filename is based on a unique identifier difficult to guess (i.e. 01K1NEXTLNZK984J6Y.pdf) and the listing of the directory is not allowed.
In the near future, we may consider adding a mechanism to enhance document access protection.
Documents are stored in the web directories for a period determined by a system parameter (14 days) and after this time they are automatically removed.
All system activities are logged with, and accessible from, protected web pages.

CONCLUSION AND FUTURE WORK

This paper attempts to emphasize the ability and flexibility of MIME-aware agents in extracting and managing parts of structured data.
Our experimentation revealed multiple advantages:

- The system offers reliability in delivering and downloading documents, thus increasing the quality of the e-mail service.
- It is simpler than the classic "user file upload" approach;
- The service is user-friendly - no training is needed;

The proposed system could be advantageously adopted in any application where automatic filtering and elaboration of message parts are required. For example we tailored the system to support an Internet Document Delivery Service between libraries (BiblioMIME project) [Gen 00].
In the future we wish to extend the gateway's functionality to incoming messages, enabling our users to download received file attachments via web, whether the sender uses our system or not.

REFERENCES

[Pos 82] J. B. Postel. RFC821: Simple Mail Transfer Protocol. ftp://ftp.isi.edu/in-notes/rfc821.txt, August 1982.
[Cro 82] D. H. Crocker. RFC 822: Standard for the format of ARPA Internet text messages - ftp://ftp.isi.edu/in-notes/rfc822.txt, August 1982.
[Mye 96] J. Myers & M. Rose. RFC1939: Post Office Protocol - Version 3. ftp://ftp.isi.edu/in-notes/rfc1939.txt, May 1996.
[Cri 96] M. Crispin. RFC 2060: Internet Message Access Protocol - Version 4rev1. ftp://ftp.isi.edu/in-notes/rfc2060.txt, December 1996.
[Fre 96a] Freed, Ned and al. RFC 2045. MIME Part One. ftp://ftp.isi.edu/in-notes/rfc2045.txt November 1996.
[Fre 96b] Freed, Ned and al. RFC 2046. MIME Part Two. ftp://ftp.isi.edu/in-notes/rfc2046.txt November 1996.
[Moo 96] Moore, K. RFC 2047. MIME Part Three. ftp://ftp.isi.edu/in-notes/rfc2047.txt November 1996.
[Fre 96c] Freed, Ned and al. RFC 2048. MIME Part Four. ftp://ftp.isi.edu/in-notes/rfc2048.txt November 1996.
[Fre 96d] Freed, Ned and al. RFC 2049. MIME Part Five. ftp://ftp.isi.edu/in-notes/rfc2049.txt November 1996.
[Gen 00] Francesco Gennai, Laura Abba, Marina Buzzi, Maria Grazia Balestri, Silvana Mangiaracina. Experience in implementing a document delivery service (short paper) - ACM, Digital Library 2000, 2-7 Luglio 2000 - San Antonio, Texas, pp. 262-263