Evolution of Messaging Standards

Stephen J. Griesmer
Richard W. Jesmajian

There are two dominant sets of standards for messaging systems and services: the International Telecommunications Union (ITU) X.400 Recommendations and the Internet messaging standards. This paper describes and compares their architectures, message formats, naming and addressing schemes, protocols, security features, multimedia support, and management services.

Introduction

The X.400 and Internet standards for messaging, each with a wide array of implementations, have evolved over the past decade. The X.400 Recommendations are created by the ITU, an international treaty standardization group within the United Nations. (See Panel 1 for definitions of abbreviations, acronyms, and terms.) Internet standards development is overseen by the Internet Activities Board, now part of the Internet Society, a worldwide federation of networking professionals representing educational, military, and commercial organizations. The number of Internet systems has grown explosively during the past few years, to an estimated 15 million users. X.400 connections are also growing among public and private messaging service providers, but not as rapidly. X.400 connections exist among most of the major public messaging services, forming a worldwide backbone. For these reasons, both standards will likely coexist and continue to evolve through their respective standardization processes. Moreover, because of common concerns and contributors, the two sets of standards will probably converge over time on topics of common interest, such as multimedia, asynchronous access, security infrastructure, simplified naming and addressing, and messaging management.

X.400 Recommendations

This section reviews the X.400 architecture, message format, naming and addressing, protocols, security architecture, content architecture, and management model.

Background. At the end of the 1970s, many members of the International Telegraph and Telephone Consultative Committee (CCITT), part of the ITU, began experimenting with public electronic messaging services. During that time, they expressed the need to produce standards to link these public messaging services internationally over a public X.25 data networking backbone. In 1980, the CCITT approved a program of work within its Study Group 7 to develop a set of recommendations on Message Handling Systems (MHS). These recommendations govern messaging interfaces among public messaging services, and between public and private messaging services. Because the ITU uses a formalized consensus process for developing recommendations during a four-year study period, the X.400 series of recommendations for MHS was first published in 1984.[1] The 1984 series is often referred to as the "red book," named for the color of the CCITT recommendations' cover during that study period. About the same time, a number of computer vendors, working through the International Organization for Standardization (ISO) forum of Subcommittee 18, began work on the Message-Oriented Text Interchange System (MOTIS), an electronic mail standard for office automation.

In the fall of 1984, the ISO and the CCITT agreed to jointly enhance the 1984 X.400 Recommendations and publish the results. In 1988, the CCITT published the X.400 "blue book" recommendations.[2] By 1990, the ISO published its ISO 10021, parts 1 through 7, which were technically aligned with CCITT's 1988 version of X.400. With its publication by ISO, X.400 became the first application to use the Open Systems Interconnection (OSI) seven-layer Reference Model.[3]

Panel 1. Abbreviations, Acronyms, and Terms

ADMD — administration management domain
ADPCM — adaptive differential pulse code modulation
ANSI — American National Standards Institute
API — application program interface
ASCII — American Standard Code for Information Interchange
ASN.1 — Abstract Syntax Notation One
AU — access unit
BER — Binary Encoding Rules
CCITT — International Telegraph and Telephone Consultative Committee
CMIP — Common Management Information Protocol
DEK — data-encrypting key
domain — an organizational area
EDI — electronic data interchange
e-mail — electronic mail
FTP — File Transfer Protocol
G3 — Group 3
G4 — Group 4
GDMO — Generic Data Model for Objects
GIF — Graphical Interchange Format
GOSIP — Government Open Systems Interconnection Profile
IEC — International Electrotechnical Commission
IETF — Internet Engineering Task Force
IK — interchange key
IP — Internet Protocol
IPM — interpersonal messaging
ISDN — Integrated Services Digital Network
ISO — International Organization for Standardization
ITU — International Telecommunications Union
JPEG — Joint Photographic Experts Group
LAN — local-area network
MHS — Message Handling Systems
MIB — management information base
MIC — message integrity check
MIME — Multipurpose Internet Mail Extensions
MOTIS — Message-Oriented Text Interchange System
MPEG — Motion Picture Experts Group
MS — message store
MTA — message transfer agent
ODA — Open Document Architecture
OR — originator/recipient
OSI — Open Systems Interconnection
PDS — Physical Delivery Service
PEM — Privacy Enhanced Mail
POP — Post Office Protocol
PRMD — private management domain
RFC — Request for Comments
RSA — Rivest–Shamir–Adleman algorithm
RTSE — Reliable Transfer Service Element
SGML — Simple Generalized Markup Language
SMTP — Simple Mail Transfer Protocol
SNMP — Simple Network Management Protocol
TCP — Transmission Control Protocol
TFTP — Trivial File Transfer Protocol
UA — user agent
X.25 — ITU standardized packet-switching protocol

In 1992, CCITT and ISO developed a set of optional enhancements to the 1988 X.400 Recommendations. The 1992 version[4–18] is completely backward-compatible with the 1988 X.400 Recommendations. As a result of standardization, most links among commercial electronic mail (e-mail) service providers today use X.400. In addition, most major messaging system providers offer an X.400 system or gateway software, based on the 1984 version of X.400. A subset of messaging system vendors and public messaging service providers offers the 1988 version.

The X.400 market also has been boosted by the endorsement of a number of governmental bodies. Since 1988, the U.S. Government has mandated the procurement of OSI-based communication products and services, including X.400, through its Government OSI Profile (GOSIP). The U.S. Government's FTS-2000 messaging network, split between AT&T and Sprint, based its interoperability on X.400. In addition, in 1992, the European Commission recommended X.400 as the standard means of messaging.

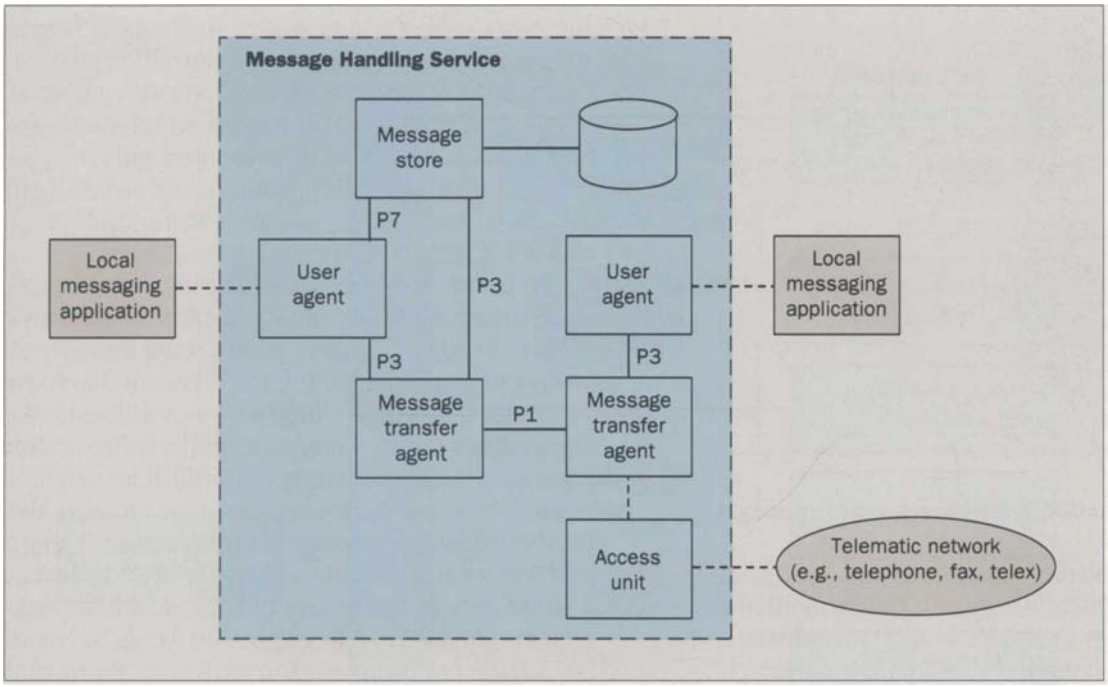

Figure 1. Messaging functional objects and protocols.

X.400 Architecture. X.400 specifies a distributed messaging model and defines an application communication protocol designed to operate over the lower six layers of the OSI Reference Model. It may also operate over other communication stacks, such as the Transmission Control Protocol/Internet Protocol (TCP/IP), but that is outside the scope of the X.400 Recommendations.

X.400 defines four types of functional objects in a messaging system: user agent (UA), message store (MS), message transfer agent (MTA), and access unit (AU). All four objects communicate using a full seven-layer OSI stack. The functional model of the objects, shown in Figure 1, along with the protocols used between the objects, is based on a client-server model.

User agent. The user agent sends and receives standardized X.400 messages on behalf of an MHS user, communicating through a standardized peer-to-peer application protocol. The MHS user, normally a messaging application on a user's local workstation or personal computer, composes or reads messages. Interaction between the MHS user and the user agent may occur either through a locally defined, proprietary interface or a standardized application program interface. Currently, X.400 defines three specialized types of user agents for different types of messaging: interpersonal, electronic data interchange (EDI), and voice.

Message store. As the name implies, a message store is a mailbox that stores messages. Historically, it has been co-located with the user agent, mostly for performance reasons. In the local-area network (LAN) environment, however, it is sensible to locate the message store on a server, and the user agent on a client workstation. In a mobile messaging environment, the user agent and the message store may be decoupled to allow different types of devices to access a universal mailbox from the office, at home, or on the road. In X.400, the message store is independent of message type; however, the types of messaging information that can be extracted vary according to the message type.

Access unit. An access unit, typically tailored to a particular communication service, connects X.400 MHS to users of other services, such as postal, telephone, telex, or facsimile. Subscribers to other services are modeled as *indirect* users of X.400. Their regular service (e.g., telex) may only allow them access to a subset of the messaging service and protocol features available to *direct* X.400 users (e.g., MHS users).

Message transfer agent. The message transfer agent routes messages submitted directly by a user agent, or indirectly from the user agent, through the

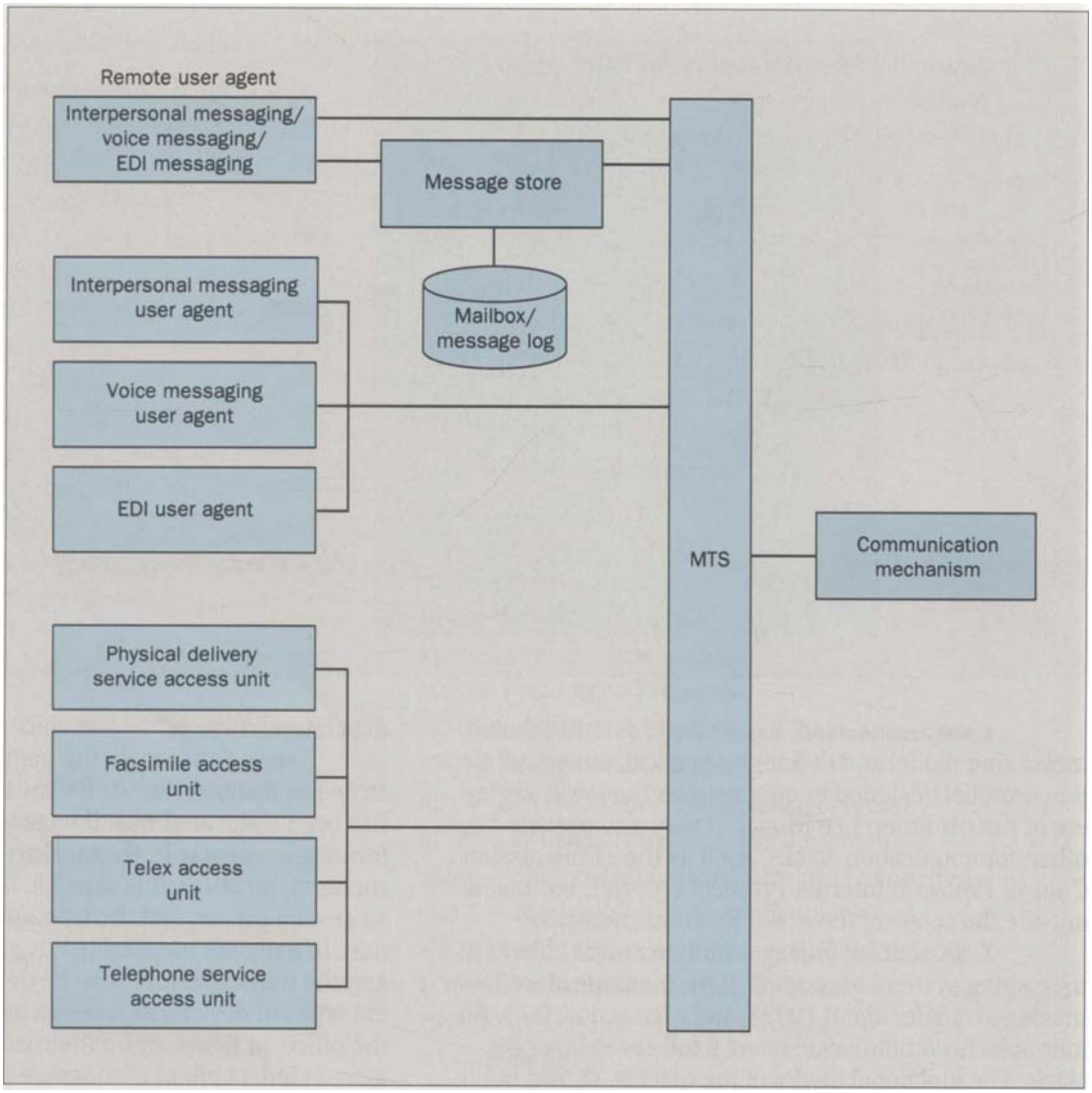

Figure 2. Architectural components of the 1992 X.400.

message store. It directs the messages to other message transfer agents, and manages the delivery of messages to a message store, user agent, or access unit. Message transfer agents are generic, and operate on all types of messages. They also expand distribution lists, generate message (and distribution list expansion) trace information, and originate delivery reports, as described in later sections of this paper.

In its 1984 version, X.400 defined a user agent for interpersonal messaging, the message transfer agent, and an access unit for telex, as well as protocols for message transfer and simple submission and delivery between the user agent and message transfer agent. In its 1988 version, X.400 added definitions of the message store, remote user agent, and a physical delivery service (PDS) access unit. The protocol definitions from 1984 were also upgraded to support new features and permit future extensions. Details are described in Recommendations X.400,[5] X.413,[8] and X.419.[9] In 1990, X.400 added an EDI user agent that composes and retrieves electronic store-and-forward transactions, such as purchase orders and bills of lading that conform to EDI standards. The 1992 version of X.400 added support for a voice messaging user agent specially tailored for users with telephone

access, a telephone access unit to deliver voice messages over the public telephone network, and messaging for store-and-forward facsimile services. An enhanced message store for multiple mailboxes, foldering, and message logging services is planned for release in 1994. Figure 2 shows the architectural components in the 1992 X.400 Recommendations.

In addition to functional objects, X.400 also explicitly specifies enterprise entities, called domains, which, when linked together, form the global messaging service and infrastructure envisioned by the ITU. The two types of domains are: (1) those managed by an ITU Administration (e.g., a public messaging service provider), called administration management domains (ADMDs); and (2) those managed privately by organization (e.g., a corporation or nongovernmental organization), called private management domains (PRMDs). ADMDs provide message routing and relay services to direct and indirect subscribers and PRMDs. In the X.400 model of global messaging, the messaging transfer infrastructure is a set of interconnected ADMDs, each supporting a set of zero or more PRMDs that, when linked together, enable international communications. Distinctions between ADMDs and PRMDs are few, but arise in X.400 naming and addressing, to be discussed later.

X.400 Message Format. An X.400 *message* is divided into three components: a message envelope, a messaging service protocol, and a message body (or content architecture).

- A message envelope — Generic information needed to route any type of message, e.g., identifiers and addresses.
- A messaging service protocol — Protocol specific to a type of message and notification exchanged among user agents. Standardized types of messages supported by X.400 today include interpersonal, EDI, and voice. Message types that have been privately agreed on may also be exchanged. For more details, see "X.400 User Agent Protocols," later in this paper.
- A message body — The message body is represented by zero or more components, called body parts. Details are described in "X.400 Content Architecture," later in this paper.

Figure 3 illustrates the three-component X.400 message format.

X.400 also specifies reports and notifications to be returned to the message originator. A *report* is

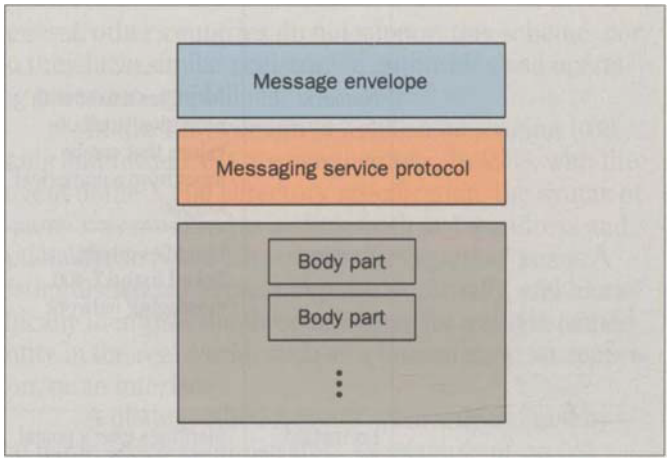

Figure 3. Three-component X.400 message format.

generated by the message transfer agent to indicate whether or not a message was delivered successfully to a user agent, message store, or access unit. A delivery report is automatically generated when delivery succeeds, but only if a report has been requested; a nondelivery report is automatically generated whenever delivery fails. Reports also indicate whether a message was delivered to a distribution list, was expanded, and was passed on to each member of the list. The syntax of reports is specified in X.411.[7]

A *notification* is a special user-agent-level message that indicates that responsibility for the message's contents was accepted or rejected by the intended recipient. The definition of responsibility differs by service type: interpersonal messaging defines receipt and nonreceipt notifications; EDI messaging defines positive, negative and forwarding notifications; and voice messaging defines receipt, nonreceipt, and service notifications. The triggering mechanism for notifications is a local issue.

Another type of message is a *probe*, a message with an empty body. A probe's main purpose is to test the deliverability of a message to a recipient. Although the probe itself is never delivered, it always generates a delivery report. The usefulness of a probe will diminish with the emergence of globally interconnected directories.

All information objects in X.400, as with other OSI protocols, are represented using a common abstract syntax notation, Abstract Syntax Notation One (ASN.1).[19] Although ASN.1 can be encoded in a variety of ways, the Basic Encoding Rules (BER)[20] are the most widely

Table I. Forms of OR Addresses

Form	Description	Attributes	Optional/ mandatory	Example
Mnemonic	Identifies user based on geographical and organizational attributes	Country ADMD PRMD Organization Organization unit Personal name Common name Domain-defined attribute	M M O O O O O O	c=us/a=attmail /s=jesmajian
Numeric	Identifies a user with numerical attribute values that can be input from a numerical keypad	Country ADMD PRMD Numeric user ID Domain-defined attribute	M M O M O	c=us/a=attmail /n-id=2134567
Terminal	Identifies a terminal linked to the X.400 messaging network	Country ADMD PRMD Terminal ID Terminal type Domain-defined attribute	O O O M O O	/t-id=19098484311
Formatted Postal	Identifies user's postal address with specific attributes	Country ADMD PRMD Physical delivery service Physical delivery country Postal code Office name Office number Organization name Street address P.O. Box Post Restante address Postal name Extension OR Addr. Comp. Extension Phys. Del. Addr. Comp. Local postal attribute	M M O O M M O O O O O O O O O O	c=us/a=attmail /pd-c=us /pd-pc=04322 /pd-of=Shiffer, IN /pd-s=20 Rosy Lane /pd-pn=S. Brown
Unformatted Postal	Identifies user's postal address with general attributes	Country ADMD PRMD Physical delivery service Physical delivery country Postal code Unformatted postal address	M M O O M M M	c=us/a=attmail /pd-c=us /pd-pc=04322 /pd-a=S. Brown, 20 Rosy Lane, Shiffer, IN

supported. BER encodes information elements as 8-bit octets in a type-length-value sequence.

X.400 message size is unconstrained. However, regional implementation agreements have limited message size to 2 megabytes within the U.S., and 10 megabytes elsewhere.

X.400 Naming and Addressing. An X.400 address, called an originator/recipient (OR) address, is based on attributes commonly associated with a user, e.g., country, service provider, organization, personal name, etc. In keeping with postal addressing, messaging treats an underspecified address as deliverable if it identifies a unique mailbox or rendering device.

An OR address is composed of a set of hierarchical attributes, each consisting of a type and one or more values (e.g., the attribute type "country" with the attribute value "US"). To accommodate the range of addresses used in messaging systems and to maximize interworking with other communication mechanisms, five forms of OR addresses, listed in Table I, were defined in the 1988 version of X.400. The different forms of addresses are specified to reach a variety of terminal devices, for delivery to different access units, and to work with existing networks (e.g., the telephone network and public data networks). The examples in Table I are written according to Recommendation F.401,[21] which

specifies a business card format for X.400 addresses. An OR address may denote either an individual recipient or a distribution list; no syntactic distinction is made between the two types of addresses. Distribution lists may have other distribution lists as members, and list owners may restrict specific user access by setting access permissions. Members of a distribution list may be named using an OR name containing any of the five OR address forms.

X.400 addresses satisfy two requirements: ease of global routing and flexible addressing. For backbone routing, the country, ADMD, and PRMD attributes are used by ADMDs to reach a destination management domain. Within a destination domain, the remaining attributes are used to route messages to specific mailbox sites. The attribute type and value syntax of the X.400 address support addresses for interpersonal communications, EDI messaging, voice messaging, fax interworking, and telephony interworking.

Using ADMD and PRMD attributes for routing creates administrative naming problems for PRMD operators (and users) when a PRMD subscribes to more than one ADMD, possibly for redundancy. In this case, each PRMD user would have multiple addresses, depending on the ADMD chosen. Arguably, from a PRMD administrator's perspective, this is a major X.400 addressing design deficiency, because its users would need to have multiple X.400 addresses whose only difference is the value of the ADMD attribute. To remedy this situation for PRMD users within the U.S., a scheme that is backward-compatible with X.400 (1984) has been devised to allow null or "single-space" ADMD names to indicate whether a nationally registered PRMD name value is being used. To support this scheme, the U.S. has established a national PRMD name registration authority, administered by the American National Standards Institute (ANSI), to register nationally unique PRMD name values. U.S. ADMD names are also registered with ANSI.

Messages move transparently over X.400 1984, 1988, and 1992 implementations when PRMDs use this construct to assign OR addresses to its users. In cooperation with the Electronic Messaging Association, ADMDs operating in the U.S. are preparing to support this naming convention. Within the next few months, they will be sharing PRMD reachability information and developing cooperative settlement procedures, as described in the *U.S. ADMD Behavior Guidelines*, jointly developed by ANSI and the U.S. Department of State subcommittees. In general, other countries do not support this scheme, nor do they have similar registration authorities and operating procedure guidelines.

In the 1984 version of X.400, a messaging (OR) name and OR address are synonymous. In 1988, with the advent of the X.500 Directory specification, the syntax of a name was modified to include both an OR address and, optionally, an X.500 Directory[22] *distinguished name*. A distinguished name unambiguously, globally, and hierarchically identifies the directory entry for a single named entity in the real world, such as a human user, an application, or an interface.

A distinguished name is arbitrarily defined by the name space administrator. An example of an OSI Directory-distinguished name is:

C=US/L=NJ/L=Holmdel/O=AT&T/CN=Gorgi

where C is country, L is locality, O is organization, CN is common name, and / is the delimiter between attributes. It is useful to note that distinguished name attributes do not always use the same syntax as corresponding OR address attributes. X.400,[5] X.402,[6] and X.411[7] contain more information about X.400 naming and addressing.

X.400 Message Transfer Protocol. The first MHS protocol defined was the Message Transfer System protocol, used between message transfer agents. Because it was the first protocol defined, it was labeled "P1." The P1 protocol transfers messages, probes, and reports, and also contains mechanisms to support message redirection, distribution list expansion, security services, and content conversion (e.g., text to fax). The protocol initiates a session with a remote MTA using an MTA bind command that transmits the identity and credentials of the originating MTA to support secure message transport. Next, the message is transferred with its envelope, which indicates the type of actions that should be performed on it, i.e., relay or attempt to deliver it to the named recipient. Each message transferred is confirmed by the receiving message transfer agent.

P1 specifies two modes of transfer: monologue and two-way alternate. In monologue mode, only the initiator of the bind can transfer messages. In two-way alternate mode, the receiving MTA can send a request to the originator asking for a "turn" to send messages. When the originating MTA finishes sending messages, it passes a token, or signal, for a turn to the receiving MTA. This

process can be repeated a number of times until the session is closed. The upper-layer OSI services underlying P1 may be used to ensure the reliable transfer of messages. This reliable delivery is provided by the checkpoint and recovery services of the Reliable Transfer Service Element (RTSE)[23] when OSI Transport Class 0 is used. The checkpoint and recovery services of RTSE are not needed when OSI Transport Class 4 is used, because the lower layers of OSI perform these services.

X.400 Message Access Protocols. To support remote submission and delivery to and from an MTA (e.g., to and from a PC or a smart terminal), a message transfer service access protocol (P3)[7] was defined. P3 is built on top of an OSI remote procedure call facility, the Remote Operations Service Element and, optionally, the RTSE. P3 defines three interfaces — submission, delivery, and administration — each supporting a set of operations. The submission interface supports the submission of messages and probes, the cancellation of deferred delivery requests, and the restriction of services that a user can request. The delivery interface supports the delivery of messages and reports to the end user, as well as restricting delivery parameters, such as the length of a message delivered. The administration interface enables a user to change individual profile parameters. It also allows the user or message transfer service to change credentials (e.g., a password). The P3 protocol only applies to user agents or message stores that are remote from an MTA; co-located user agents, message stores, and MTAs may use proprietary protocols to communicate. Interactions may be initiated by either the MTA, the remote user agent, or the remote message store.

In the 1988 version of X.400, the message store is defined in terms of the interfaces it presents to remote user agents. These interfaces are accessed through the P7 protocol.[8] Like P3, P7 is built on the Remote Operations Service Element and, optionally, the RTSE. P7 accesses three interfaces in the message store: retrieval, indirect submission, and administration. The retrieval interface allows the user agent to access the message store, summarize its content, list selected information about a set of entries in a message store (e.g., header information, such as subject or originator), retrieve information from individual messages, delete messages, register information about actions to perform when messages satisfying certain attributes arrive, and alert the user agent to incoming messages. The indirect submission interface allows the user agent to submit messages through the message store to a message transfer agent. As with P3, the administration interface allows the user to change profile parameters or credentials. A novel aspect of the message store, which closely mimics the OSI Directory information model, is the modeling of the user's mailbox as a general information store. Here, the entries are individual messages, and headers and content are treated as attributes of the entry, which allows a very general retrieval mechanism to be specified. In addition, the X.400 message store also contains the notion of registration of actions to be performed automatically when messages matching certain characteristics (e.g., fax messages) are delivered. In the 1994 version of X.400, the message store will support folders and logging services for its users. Folders will allow messages to be stored and retrieved according to user-defined groupings.

X.400 User Agent Protocols. As stated earlier, X.400 has defined three types of user agents: interpersonal messaging (defined in X.420), EDI messaging (defined in X.435), and voice messaging (defined in X.440). Each type of user agent uses a different protocol to exchange information with its peers. Each protocol specifies the set of messages and notifications exchanged, the message service protocol (header) fields in the message and notifications, the content architecture of the messages and notifications, and user agent behavior rules. The difference in behavior rules is especially apparent in the forwarding function. During forwarding, the received message is repackaged, either unmodified or modified, and submitted to a message transfer agent as the content of a new user agent message destined for other users or to a distribution list. In addition, each user agent has a set of attributes defined to extract message service protocol fields from the message store.

The peer-to-peer interpersonal messaging protocol[10] between user agents is called P2 or IPM. IPM is composed of a set of header fields and a message body. The header fields correspond to those of an interoffice memorandum: originator, primary recipients, copy recipients, blind copy recipients, subject, language, etc. P2 also includes the definition of the interpersonal notification. The interpersonal notification signals whether a message has been accepted or rejected by the recipient.

Table II. X.400 Body Part Types

Body part type	Subtypes
Text	Telex, IA5
Voice	32-Kbit/sec ADPCM
Image	G3 Fax G4 Class 1 Fax
Teletex	Teletex
Videotex	Videotex
Encrypted	Encrypted
Message	X.400
Mixed mode	Processible Group 4
Bilaterally defined	Bilaterally defined, uses object identifiers
Nationally defined	Nationally defined, uses object identifiers
Externally defined	Externally defined, uses object identifiers (e.g., ODA)
General text body part defined	Characters encoded per ISO/IEC 8859
File	Subtypes defined by body part parameters OSI object identifier specifies file type
Multipart	Supported without part relationships
Message digest	Supported with message body part

In addition to interpersonal messaging, two other peer-to-peer user agent protocols have been defined: EDI messaging protocol, and voice messaging protocol.

The EDI messaging protocol,[11] P_{edi}, or P_{35}, relays EDI transactions, defined by other standards such as X12.[24] The EDI messaging protocol contains header fields that enable a receiving EDI application to evaluate the need to immediately operate on the message body without actually accepting, parsing, or processing the EDI interchange (e.g., encoding of the interchange and character repertoire, transaction set types included in the interchange, the originator, message and body part cross-referencing information, and responsibility-passing-allowed indication).

EDI messaging defines three types of notifications: positive, negative, and forwarding. Positive and negative notifications signal the acceptance (or rejection) of responsibility for a message by the recipient or user application. Forwarding notifications confirm that the message was forwarded by the intended recipient's user agent to another recipient.

A voice messaging protocol,[12] P_{voice}, or P_{40}, exchanges voice messages whose encoding is defined in other standards, such as Recommendation G.721 (which standardizes a 32-Kbit/second adaptive differential pulse code modulation [ADPCM] digitized voice encoding).[25]

The voice messaging protocol contains header fields specific to voice, such as spoken subject or message cross-reference information and body part cross-reference information. Forwarding is unconstrained.

Voice messaging defines three types of notifications: receipt, nonreceipt, and service. Receipt and nonreceipt notifications signal the acceptance (or rejection) of responsibility for a message by the recipient or user application. Service notifications indicate a received message has been made available to the local voice mail application, although certain message service elements are not supported. A message originator who receives this notification may then take alternate actions to remedy the matter. The behavior of a receiving voice

messaging user agent regarding generation of these notifications is governed by the originator requesting them.

X.400 Content Architecture. Recommendation X.420 defines the content architecture of an X.400 message as a message body consisting of an ordered set of body parts, each tagged by the messaging protocol according to its type. Body parts of some types not only have data, but also parameters that describe the body part and typically contain formatting and control information for reassembly of the message upon receipt.

For extensibility, the 1988 version of X.400 contains an externally defined body part whose syntax and semantics are denoted by an OSI object identifier. This allows X.400 to carry types of information not specifically defined in the X.400 recommendations, e.g., other standardized image formats, such as the Joint Photographic Experts Group (JPEG)[26] format, and the Motion Picture Experts Group (MPEG)[27] format. A body part consists of two distinct components: a parameter component and a data component. The parameter component is specified by the owner of the body part identifier. It may contain other object identifiers or information needed for the recipient to reconstitute the data component. In general, the data component is viewed as a binary object whose syntax and encoding is indicated by the body part type or parameter information. Table II defines the body part types supported by the 1992 version of X.400.

Although all X.400 user agent definitions use the same content architecture, some user agent specifications define constraints or extensions that distinguish it from its peers. For example, EDI messaging defines the message body to include a primary body part that contains one EDI-formatted interchange encoded according to EDI encoding rules, and zero (or more) related body parts that contain other non-EDI encodings, such as multiple drawings or images available to the recipient when the primary body part's EDI interchange is being processed. In voice messaging, the message body includes one primary body part that contains one voice object encoded according to voice-encoding rules, and zero (or more) related body parts encoded according to non-voice-encoding rules. Because related body parts may be in the same message or other messages, EDI messaging and voice messaging specifications define mechanisms for cross-referencing body parts within a single message, or across one (or more) messages; interpersonal messaging does not. In all user agent types, other related body parts include a forwarded message, which could have been forwarded by a chain of recipients. The forwarding model differs, however, among user agent types.

X.400 Security. Security functionality and service features added in the 1988 version of X.400 include:

- Authentication;
- Access management;
- Confidentiality of content and message flow;
- Integrity of content and message sequence;
- Nonrepudiation with proof of delivery, receipt, retrieval, submission, and transfer;
- Nonrepudiation with proof of notification;
- Nonrepudiation of content;
- Security labeling; and
- Security management.

X.400 supports both symmetric and asymmetric cryptographic techniques to ensure data confidentiality and to generate digital signatures for message authentication. Symmetric encryption uses a private encryption key shared by a message originator and recipient to encrypt and decrypt a message. Asymmetric encryption uses a pair of matched keys for encryption and decryption. A "private" key is kept secret by its owner, and a "public" key is made publicly available through the services of a certification authority, described in the Privacy Enhanced Mail (PEM) portion of the Internet messaging security section.

A message encrypted by the private key can be decrypted by the public key; a message encrypted with a public key can only be decrypted by the possessor of the corresponding private key. The greatest advantage of an asymmetric encryption scheme is its ability to share keys securely. A symmetric encryption scheme is hampered by the problem of securely sharing the symmetric key among communicants. The asymmetric encryption scheme is useful for digital signatures. A message originator encrypts unique data calculated from a message using his private key; recipients use the originator's public key to decrypt the data, recalculate the unique data, and compare it with the value sent by the originator, to prove that the originator indeed encrypted the signature and sent the message.

The security features in X.400 are optional. Most of the security services are provided by the message transfer agent protocol; others are provided as part of the user agent protocols. EDI messaging and voice messaging user agent protocols provide extensive mechanisms

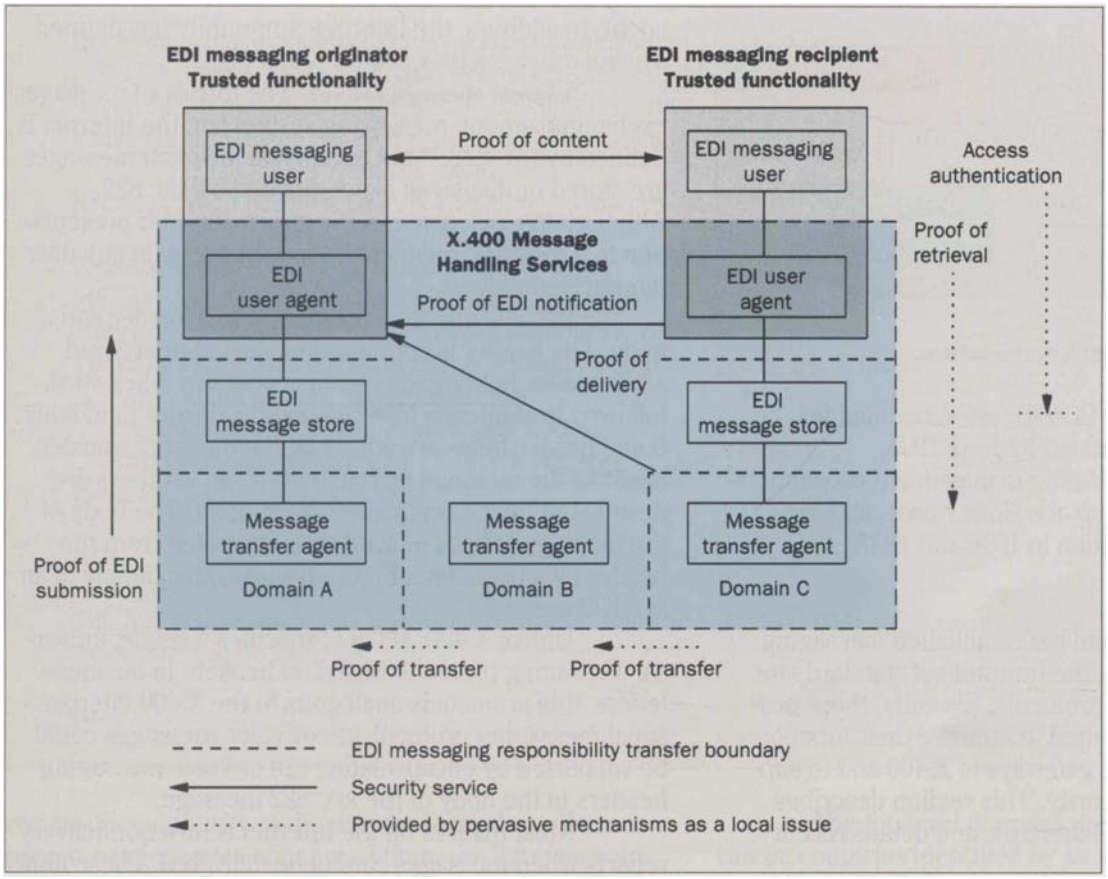

Figure 4. EDI messaging security model.

for securing the message content, whereas interpersonal messaging primarily supports encryption of the body parts of a message, discussed earlier in "X.400 User Agent Protocols." The EDI messaging security model, shown in Figure 4, best exemplifies the security services provided in MHS. Additional information is provided in F.435 annex C,[28] F.440 annex G,[29] X.402 annex D, X.411, X.420 annex E, X.435 annex I, and X.440.

X.400 Management. More businesses and service providers are seeking common ways to administer and manage their X.400 messaging system components. Work on the management of X.400 messaging systems addresses this need. The ISO and ITU messaging management experts are collaborating to develop a common management framework and infrastructure. They are using the ITU's Telecommunication Network Management tools to *model* the management of X.400 entities and resources. In telecommunications networks, the protocol for data collection and exchange is Common Management Information Protocol (CMIP).[30] In addition to specifying a rich set of management operations and reporting functions, CMIP standardizes the management information base (MIB) to hold the managed resource data. CMIP's management information base is specified using the Generic Data Model for Objects (GDMO), which was developed specifically to define managed objects. A user of CMIP and the Telecommunication Network Management uses GDMO to define specific MIB extensions for managing the resources and entities of message handling systems. These are specified in X.463 through X.470. To date, ISO/ITU efforts, initiated in 1992, have progressed as follows:

- *MHS Management: Model and Architecture* (X.460) has progressed to draft recommendation status (and to ISO Draft International Standard status). This work should be finalized in November 1994.
- Draft recommendations for *MHS Management: Logging Functions* (X.463), and *MHS Management: Message*

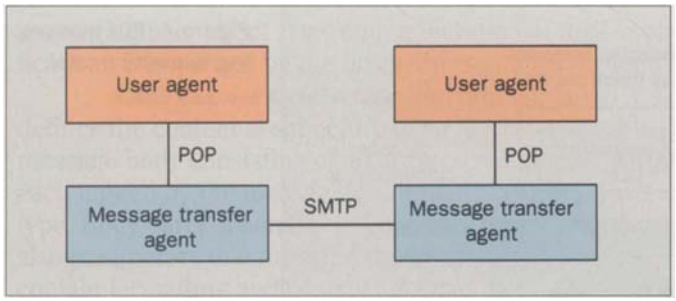

Figure 5. Internet messaging architecture.

Transfer Agent Entity (X.464), are scheduled for approval in the ITU and ISO by June 1995.

- The other seven messaging management recommendations are progressing at a slower pace, and are scheduled for publication in 1996 and 1997.

Internet Standards

Before the ITU and ISO established messaging standards, developers of the Internet set standards for messaging formats and protocols. Recently, these protocols have been augmented to improve their functionality, as well as to create gateways to X.400 and to support multimedia and security. This section describes the Internet messaging standards, and details recent enhancements.

Internet standards are specified by Requests for Comments (RFCs). RFCs are generally available on-line on the Internet. Internet standards are developed by the Internet Engineering Task Force. Not all RFCs are Internet standards; the Internet Activities Board determines whether an RFC will be elevated to an Internet standard.

In recent years, Internet standards have gained public attention. Branches of the U.S. Government are now studying ways to include Internet protocols in their procurement profiles.

Internet Messaging Architecture. Internet messaging protocols are designed to be sent over the TCP/IP protocol stack. Like X.400, Internet messaging distinguishes between a user agent and a message transfer agent. The message transfer agent communicates with other message transfer agents using the Simple Mail Transport Protocol (SMTP).[31] Figure 5 illustrates the Internet messaging architecture. Internet RFC 1225 also defines the Post Office Protocol (POP),[32] a simple message retrieval protocol between a user agent and a message transfer agent. In addition, the Internet community has defined several other mailbox access protocols.[33–35]

Internet Message Format. The format of messages exchanged among messaging systems on the Internet is defined by RFC 822,[36] but the format in which messages are stored or displayed is not dictated by RFC 822. Although RFC 822 conveys the information for presentation to a message recipient, it does not contain any user interface guidelines.

RFC 822 divides a message into a header and a body. The header is composed of a set of unordered ASCII header fields, each of which contains a keyword, followed in sequence by a colon and a header field body. Some header fields are added by the message transfer agent as the message is transmitted, while others are designated by the originator's user agent. The body of the message is lines of ASCII text, separated from the header by a blank line. Figure 6 shows an example of an RFC 822 message.

Unlike X.400, RFC 822 specifies a single, universal messaging protocol. Based on headers in business letters, this protocol is analogous to the X.400 interpersonal messaging protocol. EDI or voice messages could be supported by encapsulating EDI or voice messaging headers in the body of the RFC 822 message.

Most mailers on the Internet return nondelivery reports when messages cannot be delivered. These nondelivery reports, usually accompanied by the original message, often contain error information obtained from the SMTP, to be discussed later. There is no standardized option in RFC 822 for the originator of a message to request delivery or nondelivery reports, either on a per-recipient or per-message basis. Also, RFC 822 does not standardize a specific report format.

In addition, the Internet and RFC 822 do not provide a means for requesting that receipt notifications be sent when a message is delivered. Notifications have been controversial, particularly in the Internet environment, which emphasizes privacy of a messaging user.

Internet Message Naming and Addressing. In addition to defining message formats, RFC 822 specifies the messaging address format used on the Internet. An address consists of a local part, which identifies a user of the messaging system (i.e., sender or recipient), and a domain *label*, which identifies a content in which the local part is unique. As an ordered, unlimited hierarchy, each domain contains subdomains that register with a

```
Received: from att!tango.cos.com by arch4.ho.att.com (4.1/EMS-1.0 main.cf 1.36
   9/1/93 (SMI-1/SVR4)) id AA17926; Fri, 28 Jan 94 12:07:54 EST
Received: by gw1.att.com; Fri Jan 28 12:05:43 EST 1994
Received: from tango.cos.com by coincd4000.cos.com id SMTP-0012d494604016578;
        Fri, 28 Jan 94 12:06:12 -0500
Received: from head.cos.com (head-other) by tango.cos.com (4.1/SMI-4.1)
        id AA15199; Fri, 28 Jan 94 12:05:22 EST
From: atm@tango.cos.com (Amy Morris)
Message-Id: <9401281705.AA15199@tango.cos.com>
Subject: X.500 Interoperability Testing Demonstration
To: RDS@ahse.cdc.com (Ron Swan), edgar@osi.ncsl.nist.gov (Carol Edgar),
      sjg@arch3.att.com (Steve Griesmer),
      rgass@coincd4000.cos.com (Robin T. Gass),
      nkp@coincd4000.cos.com (Nancy K. Pierce)
Date: Fri, 28 Jan 94 12:05:22 EST

All:

The X.500 interoperability testing demo went extremely well.  No glitches.  15 reporters
showed to feature the event.

Amy
```

Figure 6. Sample RFC 822 message.

registration authority of the domain level above them to preserve uniqueness of names. Often, the leftmost component of a domain name is a machine name. RFC 822 addresses may also specify distribution lists or groups. If explicit source routing is desired, a route may also be associated with the address. When the domain labels are used to form a user's address, they appear left to right as an ordered set in increasing hierarchy, and are limited to a total of 256 characters, including the label delimiter (“.”). The syntax of an RFC 822 address is:

user@subdomain1.subdomain2....

where user identifies a user within subdomain1, subdomain1 is a registered subdomain with subdomain2, etc. The *user* component of the address is determined locally by the administrator of *subdomain1*. Examples of RFC 822 addresses are:

sjg@arch4.ho.att.com
s.griesmer@att.com

The top-level domains used in RFC 822 addresses can be countries identified by an ALPHA-2 country code listed in ISO 3166,[37] or one of seven reserved organizational names. The organizational domain names identify the types of organizations registered immediately subordinate to them. The assigned organization name values are:

- com — commercial enterprises,
- edu — educational institutions,
- gov — government,
- mil — military,
- net — a network type,
- org — nonprofit organization, and
- int — international organization.

Registering subdomain names with the Internet registrar is free of charge.

RFC 822 addresses may be accompanied by comments, enclosed in parentheses. These comments, which typically carry the name of the user displayed in header summaries, are passed uninterpreted by RFC 822 conforming messaging systems.

Occasionally it is useful, for administrative and testing purposes, to direct the route of a message. This is

Table III. SMTP Interaction Commands

SMTP command	Code	Description
DATA	DATA	Send message header and body.
EXPAND	EXPN	Return membership of mailing list.
HELLO	HELO	Establish an SMTP session.
HELP	HELP	Send help information.
MAIL	MAIL	Identify originator's address.
NO OP	NOOP	Null command.
QUIT	QUIT	Terminate an SMTP session.
RECIPIENT	RCPT	Identify recipient addresses.
RESET	RSET	Abort a session.
SEND	SEND	Send a message to one or more terminals.
SEND AND MAIL	SAML	Send a message to a terminal and, in addition, to a mailbox for a user.
SEND OR MAIL	SOML	Send a message to a terminal. If fails, send message to mailbox for a user.
TURN	TURN	Allow SMTP receiver to send messages.
VERIFY	VRFY	Validate a user address or resolve name to address.

done using source routing — the specification of a route by a message originator. In the RFC 822 specification (although not always in practice), a source route is bracketed by angle brackets, and each route component is preceded by an "@" and terminated with a colon. For example:

<@arch1.att.com, @arch4.att.com:
s.griesmer@arch4.ho.att.com>

where the route is an ordered list of domains through which the message should be routed. RFC 822 also permits an address without a route to be enclosed in these angle brackets. Some mailers generate this form of addressing as their native mode. Because the Internet is engineered to determine routes automatically, using the Domain Name Service, source routing is officially discouraged. (The Domain Name Service, the normal name resolution software used on the Internet, translates a domain name into an Internet address.)

RFC 822 also permits host network addresses, called *domain-literals*, to be used in the address (e.g., sjg@[135.16.27.90]), where the content between "[" and "]" is an IP address. Addresses of this form bypass the Domain Name Service. This addressing form may be used when a host name is not yet registered. Although RFC 822 permits this form of addressing, its general use is strongly discouraged.

RFC 822 defines a specific syntax for naming a distribution list. Distribution lists most commonly use the same syntax as a single-user address, where resolution of the list name to a set of addresses is done locally or remotely. Less frequently, a distribution list may be composed of a group name with an accompanying list of addresses. A distribution list with a group name takes the form:

$$\text{group\_name: mailbox}_1, \text{mailbox}_2, \ldots, \text{mailbox}_n;$$

Although the list of addresses is optional, the mail system that held the distribution list would automatically resolve the group name to a set of addresses. RFC 822 does not include a history of distribution list resolution in a message.

The format of names in Internet messages is not yet defined, although the form *name@domain* may be used today. No processing is done to use the name in the message to search for an address or to verify the address. The Internet Engineering Task Force is working to adopt the X.500 Recommendations over the Internet, which would standardize the name form.

Internet Message Transfer Protocols. Before 1982, there was no messaging transfer protocol defined for the Internet. Until then, electronic mail had been transferred using special commands, MAIL and MLFL, within the File Transfer Protocol (FTP). The header and body of the message were carried as data within these commands.[38]

In 1982, the Internet Activity Board approved a messaging transfer protocol known as SMTP. SMTP is a peer-to-peer, half-duplex protocol that consists of a set of 14 text commands, 7 of which are mandatory, and a set of optional responses. In 1993, SMTP was enhanced to better support binary-encoded contents in messages.[39] Table III lists the SMTP commands.

In SMTP, commands are sent from the sending message transfer agent to the relaying, or receiving, message transfer agent. A command consists of a four-letter code, followed by command arguments in 7-bit ASCII text. Commands are typically single lines, except for the transfer of the message header and body. Each command receives a reply, consisting of a three-digit return code, followed by a single-line or multi-line textual explanation.

There are four phases in an SMTP message exchange:

1. Establish session — The sending SMTP engine sends a HELLO (HELO) command, with its domain name, to the receiving SMTP engine. If the receiving SMTP engine accepts the session, it sends a positive acknowledgment, with its domain name, to the sending SMTP engine.
2. Verify sender and recipient — Once a session is established, the sending SMTP engine sends a MAIL command to the receiving SMTP with the originator's mailbox address. After this is acknowledged by the receiving SMTP engine, a command is sent, for each recipient of the message, using the RECIPIENT (RCPT) command, with the address of the recipient as an argument. Only recipients for whom the receiving SMTP engine is responsible for delivering the message are sent the commands. The sending SMTP engine may contact multiple systems for the delivery of messages. These systems may be either the recipients' systems or mail relay systems. Each recipient command is acknowledged by the receiving SMTP engine.
3. Transfer message — Once a set of recipients is established, the message header and body are sent using a DATA command. If the command is accepted, the receiving SMTP engine returns an "Intermediate Reply" message and considers all succeeding lines to be the message. To signal the end of the message, the sender sends a period on a line by itself. The receiving SMTP engine responds accordingly.
4. Terminate session — To terminate a session, the sending SMTP engine sends a QUIT command.

Figure 7 illustrates this scenario.

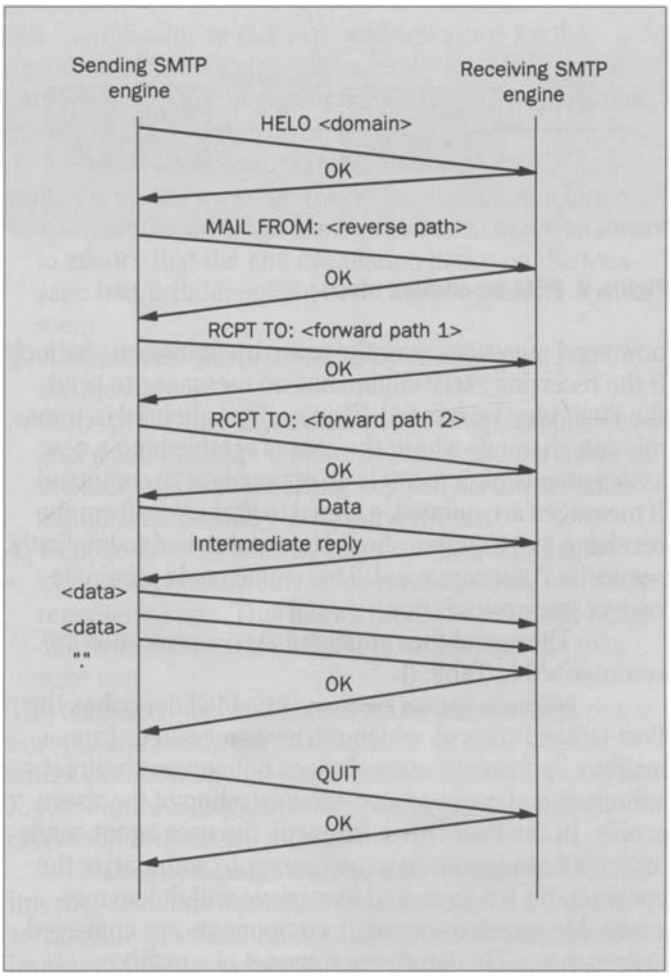

Figure 7. SMTP protocol sequence.

Because SMTP is a half-duplex protocol, only one side at a time can send messages. SMTP provides for token passing to allow the receiving SMTP engine to send messages during the current association. A TURN command passes a token from the sending SMTP engine to the receiving one, to signal that the receiving one can

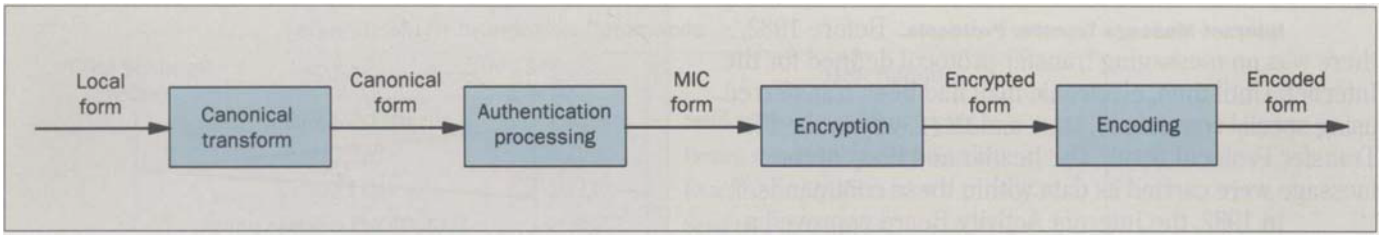

Figure 8. PEM processing steps.

now send messages over the same transmission channel. If the receiving SMTP engine has no messages to send, the TURN may be refused. This facility is helpful in transmission channels where the cost of establishing a new association is high. TURN is an optional SMTP command. If messages are queued, a signal to that effect from the receiving SMTP engine should be established to implicitly request a TURN command. This signal could eliminate unnecessary round trips.

These and the remaining SMTP commands are summarized in Table II.

Message Access Protocol. RFC 1463 describes the Post Office Protocol, which retrieves messages from a mailbox or message store. It does not support indirect submission of messages or administration of the user profile. In the Post Office Protocol, the user agent sends ASCII text commands to a mail server to summarize the contents of a mailbox, and to retrieve and delete messages. Messages or message components are conveyed in responses. The information model of a mailbox assumed by Post Office Protocol is a numbered list of messages, each with a header and a numbered list of content lines.

Internet Messaging Security. The introduction of security services into messaging standards is a relatively recent event. Because electronic mail has been used as an informal means of communication, security has been a lower priority than other end user functions. When electronic mail is used for electronic commerce (e.g., electronic data interchange), and in highly sensitive situations, security facilities need to be improved. This section describes the Internet draft standards for messaging security, known collectively as Privacy Enhanced Mail.

The Internet PEM activity began in 1985.[40] The specifications resulting from this effort are RFCs 1421–1424,[41–44] proposed Internet standards. PEM supports a set of user agent security services, including authentication, content integrity, content confidentiality, and nonrepudiation by the originator. It does not support message flow confidentiality, message sequence integrity, nonrepudiation by the recipient, or message security labeling.

All PEM services are defined independently of the message transfer service, making them usable for other transfer services. However, the current PEM specifications are oriented toward Internet 822 message format and SMTP transfer, both in design and encoding algorithm. PEM messages can be carried within X.400 messages, as well as within SMTP. Creating gateways between X.400 security services and PEM services is impossible, however, because of the encryption and encoding of security elements within the messages themselves.

PEM uses an encapsulated header in the message body to carry security information, although no header information is added to the RFC 822 header itself. The PEM-encapsulated header fields follow the RFC 822 guidelines for field specification. There are two basic types of PEM messages: (1) message integrity check (MIC), and (2) encrypted. The MIC type provides message origin authentication, content integrity, and nonrepudiation of origin (i.e., the originator cannot deny having sent a message). The encrypted message type adds content confidentiality to the list of supported services.

PEM allows the messaging system to choose the type of key systems used (asymmetric or symmetric), the MIC computing algorithm, the encryption algorithm, and the key management scheme. For message confidentiality, PEM currently only specifies one algorithm, a symmetric encryption method, the U.S. Data Encryption Standard[45] with Cipher Block Chaining mode of operation.

PEM supports a two-level keying hierarchy: a data-encrypting key (DEK) used for protecting the message content, and an interchange key (IK) used for encrypting the DEK. The PEM message contains both the protected DEK and the protected user's subject message

content. Each recipient has one interchange key, determined by the type of encryption algorithm used. If a single symmetric key shared by the originator and recipient external to this PEM message is used as the interchange key, this interchange key is used by the recipient to decrypt the originator's data-encrypting key. If the public component of the recipient's asymmetric key is used as the interchange key, the recipient decrypts the protected data-encrypting key with the private component of the recipient's asymmetric key. In both cases, the decrypted (unprotected) data-encrypting key becomes the key used by the recipient to decrypt (unprotect) the originator's subject text or the message content.

The assignment of an interchange key to each recipient does not prohibit sharing a symmetric (secret) key with more than one recipient. As more recipients share the same key, however, it is more likely to be compromised.

The MIC computation algorithms specified by RFC 1423 calculate a 16-octet string from the message text. Two algorithms are defined in RFC 1423[43] for MIC computation: Rivest–Shamir–Adleman (RSA) Message Digest 2[46] and RSA Message Digest 5.[47] The input to these algorithms is the message text itself. Once the MIC is calculated, it is encrypted using the private key of the message originator.

To decrypt the MIC, the message recipient must be able to retrieve the originator's public key from a trusted source. In PEM, the originator's public key is carried with the message in the encapsulated header field "Originator-Certificate," embedded in a certificate in this field. The certificate links the name of the originator with his or her public key. The certificate is signed by a trusted certification authority using that authority's private key. Thus, the discovery of the originator's public key now depends on the public key of the originator's certification authority.

To provide an infrastructure that allows a user to find the public key of the certification authority of any originator on the Internet, RFC 1422 defines an international hierarchy of certification authorities. From this hierarchy, a user can establish a list of certificates linking his or her certification authority with that of the originator. To facilitate the chaining through a certification path, PEM messages also carry the list of certificates to a common node of the originator and to all message recipients in an ordered list of "Issuer-Certificate" fields. While this certification mechanism was described for the decoding of the MIC, it is also used by PEM to authenticate the originator of a message and for nonrepudiation of origin.

PEM's four-step transformation process represents encrypted message text in a transmittable form:

1. Convert the message from a local to a canonical form to ensure that the MIC calculation based on the message text is independent of the local operating environment.
2. Authenticate the processing by calculating a MIC for the message.
3. Encrypt the message content. This step, applicable for encrypted message types only, involves encryption of the message text, inserting key and certificate information, and possibly encrypting the MIC.
4. Encode the PEM message by transforming binary keys, and possibly encrypted message text, into ASCII representations. This allows the SMTP protocol, which only transmits ASCII characters, to transfer the message text.

The message type, MIC-CLEAR, omits this fourth processing step and sends the message text as binary text, where permitted by message transport protocols, e.g., X.400. Figure 8 diagrams these four steps, which are reversed for message reception.

Internet Message Content Architecture. In 1992, an Internet standard was approved that created a content architecture for Internet messages, the Multipurpose Internet Mail Extensions (MIME),[48] that extends RFC 822 to include message header fields related to the content of messages. It also explicitly delimits multiple body parts in a single message. Such extensions are necessary for multimedia messages to carry tagged contents other than text, such as video, audio, image, and multipart. MIME is extensible by individual users so that arbitrary content may be privately shared by individual end users.

The MIME architecture supports a two-level, hierarchical-type space for messaging content. At the top level, a content type defines a large category of content objects that can be carried in a message (e.g., "text"). To delineate the content further, a second level, called a subtype, is specified. The subtypes are also standardized by the specification. For example, one subtype for text is "plain," meaning unformatted. Both the content type and subtype are denoted by character strings, rather than by numeric identifiers. In addition to designating the

```
From: Stephen Griesmer <sjg@arch5.att.com>
To: Al Gore <vice.president@whitehouse.gov>
Subject: Clipper chip
Content-type: multipart/mixed; boundary="cut here"

--cut here
Content-type: image/jpeg
Content-Transfer-Encoding: base64

ulajfkKSLDFIDFLSAKF...
[additional Lines of base64 encoding]

--cut here
Content-type: text/plain; charset=US-ASCII
Content-description: annotation

Al:

Here is a cartoon about the Clipper controversy that
I thought you would enjoy.

Steve
--cut here--
```

Figure 9. Sample MIME message.

content with a type and subtype, MIME also specifies attributes for content types and subtypes. Attributes may pertain to a content type, and thus to all subtypes within the content type, or to a specific subtype. Each attribute is specified by an attribute-value pair (e.g., charset=US-ASCII). The content type, subtype, and attributes are all carried in a single content header field, "Content-Type." Figure 9 shows examples of the Content-Type header field.

MIME also defines a "Content-Transfer-Encoding" field that specifies the encoding of content in a message. Encoding of message content is necessary to transfer the content through messaging systems that cannot support the transfer of binary content. As noted earlier, in most systems on the Internet today, SMTP supports only the transfer of 7-bit US-ASCII content. Gateways may further restrict the character sets that are transparently conveyed through them. MIME defines two specific encodings for message content: base64 and quoted-printable. Base64 is an encoding of binary data into a 65-character subset of ASCII characters, six bits per character. The quoted-printable encoding attempts to preserve the readability of text by representing nontextual information with ASCII characters. The encoding makes sense when most of the content is ASCII text. In this encoding, 8-bit characters are represented by a two-digit hexadecimal representation, tabs or spaces at the end of lines are eliminated, line breaks are canonically encoded, and lines are limited to a maximum of 76 characters.

Table IV. MIME Content Types

Content type	Description
Text	Formatted and unformatted textual information
Image	Still images
Audio	Audio or voice data
Video	Moving images, optionally with audio
Message	Encapsulated messages
Multipart	Multiple content types in a single message body
Application	Data whose format is defined by its application

MIME further defines two optional header fields that describe the content, "Content-ID" and "Content-Description". Content-ID identifies the content as uniquely as possible. Content-Description provides the originator with a place to describe the content in text. A mandatory "MIME-Version" header field is also defined.

MIME Content Types. MIME specifies seven content types: text, image, audio, video, message, multipart, and application, each described in Table IV. Table V describes MIME subtypes for each of the content types. The MIME specification[48] details the text, image, audio, video, and application types. The more complex content types, message and multipart, are explained briefly below.

The message content type is used to carry an encapsulated message within another message. This type can be used for message forwarding and nondelivery notifications. Message has three subtypes defined in MIME: *rfc822*, *partial*, and *external-body*. The first subtype of message, *rfc822*, is used to indicate messages conforming to RFC 822 in the message body. The second subtype, *partial*, indicates that the content of a body part is only a fragment of the content of a larger message that can be reassembled by the receiving user agent. With

Table V. MIME Subtypes

Content type	Subtype	Description	Attributes	Attribute values
text	plain	Unformatted text	charset	US-ASCII ISO-8859-X[55]
	enriched[53]	Simple formatted text compatible with SGML paired formatting commands enclosed in < >	charset	
image	jpeg	Message body contains a JPEG[26] image with JFIF encoding	None	None
	gif	Message body contains a GIF[54] image	None	None
audio	basic	Message body is audio encoded using 8-bit ISDN μlaw.[56] Sample rate=8000 Hz. Single channel.	None	None
video	mpeg	Message body is video coded according to the MPEG[27] standard		
message	rfc822	Body contains an encapsulated RFC 822 formatted message		
	partial	Body contains a fragment of a large object to be reassembled by the receiving user agent	id number total	identifier integer integer
	external body	Body contains a reference to message data rather than the data itself	access-type expiration size permission name site directory mode	ftp, anon-ftp, tftp, afs, local-file, mail-server RFC 822 date-time octets read, read-write file name domain name directory name e.g., image
multipart	mixed	Multiple, independent body parts to be displayed serially	boundary	boundary string
	alternative	Multiple body parts that denote alternative representations of the same text (e.g., plain text, enriched text)	boundary	boundary string
	digest	Multiple encapsulated messages	boundary	boundary string
	parallel	Multiple body parts to be presented simultaneously (e.g., video and voice)	boundary	boundary string
application	octet-stream	Message to be processed by application program where body contains binary data	name type padding	file name application type octets padding
	postscript	Message body is a Postscript[57] program	None	None

this subtype, each fragment of the message is transported with attributes that indicate the sequence number of the fragment and an identifier for the message as a whole. The last piece of a message must also contain the total number of fragments in the message. This subtype is useful in transmitting large documents. Some messaging systems on the Internet set 64 Kbytes as the limit for message size. The third subtype, *external-body*, indicates

that the content of the body part is stored elsewhere. The attributes of the subtype provide an access method to receive the content of a message and the parameters needed to locate the content. These parameters are specific to the access method. The access methods supported are the Internet FTP,[49] the Internet Trivial File Transfer Protocol (TFTP),[50] a local file, an Andrew File System file, and an arbitrary mail server protocol. Enough information is contained within the attributes to retrieve the content automatically when the message is viewed by an intelligent mail user agent. The *external-body* subtype will be increasingly useful in: (1) multimedia documents, where the size of messages will increase exponentially unless some form of referencing for large body parts is used; and (2) bulk mailing lists.

The multipart content type is used to support multiple body parts in a single message. In this content type, the parts of the messages are separated by a boundary line that begins with a prefix of two hyphens, "--", and whose remaining content is defined by an attribute for this content type. Each body part may also be accompanied by a Content-Type header field to denote its content type, subtype, and attributes.

There are four subtypes of the multipart content type: *mixed, alternative, digest,* and *parallel.* The *mixed* subtype is used for body parts that are independent and intended to be displayed serially. The *alternative* subtype allows the sender to include different alternative representations of the same content in one message. For example, a formatted and plain ASCII version of a document may be included in a single message. The recipient's messaging system can then display the alternative it can process. The *digest* subtype is used to convey a series of messages. The default content type of each body part is message; the subtype is *rfc822.* The *parallel* subtype is used when the body parts of a message are to be presented in parallel, e.g., voice and text. There are no synchronization points associated with this subtype.

Comparing X.400 and Internet Standards

This section summarizes the similarities and differences between X.400 and Internet standards.

Architecture. Strong architectural and modeling similarities exist between X.400 and Internet messaging regarding the use of a client-server model for specifying a user agent, message transfer agent, and message store, and the use of a hierarchical naming scheme for address constructs. However, X.400 offers a richer protocol feature set than Internet mail. X.400 is designed to meet the broad needs of telecommunication service providers by specifying interworking protocols with other telematic services and support for a broad spectrum of access devices.

The organizational model of ADMDs and PRMDs in X.400 is not used on the Internet. Instead, the Internet treats public message service providers as private networks attached to the Internet (e.g., attmail.com). These different organizational models clash at two points: (1) addressing, and (2) settlements, or charging. By separating addresses from the organizational model, a common addressing model can evolve for both the Internet and X.400.

Today, settlements in X.400 have the originating ADMD charging for message transport. The receiving and relaying ADMDs do not charge. In the near future, ADMDs will be implementing support for Recommendation D.36,[51] which will enable them to recover relay and delivery costs from the initiating domain. On the Internet, users are generally charged a flat rate for access, with message transport bundled into the access charge. As public message service providers offer Internet access in addition to X.400 connectivity, these separate charging models will probably converge.

Message Format. The X.400 message format separates the message header into an envelope between message transfer agents and a service protocol between user agents. This distinction is not explicit in RFC 822. An implicit separation exists, however, between the header fields generated and used by the message transfer agents (e.g., message information, return path information) and those used by user agents or end users (e.g., subject).

When multiple message service protocols are used, or when access units are deployed, the explicit separation of headers in the message format becomes more important. These types of services are just emerging on the Internet today. Work is also under way to define reports and notifications on the Internet.[52] The inclusion of notifications has been controversial, particularly in the Internet environment, which emphasizes privacy of a messaging user. However, in a broader messaging context, it is critical that a recipient have the ability to accept a message.

A major difference between the Internet and X.400 today is that, generally, binary messages must be encoded to be sent over the Internet, whereas X.400

Table VI. X.400 and MIME Body Part Support

Body part type	X.400 (1988)	MIME
Text	Telex, ASCII	ASCII, 8859-x, enriched text
Voice	32-Kbit/sec ADPCM	ISDN μlaw
Image	G3 Fax, G4 Class 1 Fax	JPEG, GIF
Teletex	Teletex	—
Videotex	Videotex	—
Encrypted	Encrypted	—
Message	X.400	822
Mixed mode	Processible Group 4	—
Bilaterally defined	Bilaterally defined	Extension mechanism
Nationally defined	Nationally defined	—
Externally defined	Externally defined	Application
File	Any representation	Application
Content references	—	Message/external body
Partial messages	—	Message/partial
Multipart	Supported without part relationships	Supported with part relationships
Message digest	Supported with message body part	Multipart/digest
Video	Externally defined	MPEG

messages are binary in their native mode. This restriction requires the encoding of binary information transmitted over the Internet, and is recognized by the Internet community, which is specifying support for 8-bit transport by SMTP.[39]

Naming and Addressing. The X.400 and Internet naming models both rely on hierarchical addressing. The X.400 address is composed of unordered attribute type-value pairs, whereas the Internet address is an ordered set of values, both in the address itself and in the domain name. This distinction requires users to be aware of the messaging environment to which a message is destined when sending a message, and to encapsulate additional information in an address for gatewaying from one messaging environment to the other. Agreement on a common addressing model for X.400 and the Internet would be a major step forward in easing interworking. An interesting proposal from the U.S., now surfacing in the ITU, is the use of the Internet address as an additional standardized address form in X.400. This proposal has the intriguing possibility of a common global addressing form endorsed by the ITU, ISO, and the Internet Society.

Message Transfer Protocols. The functions of X.400 P1 Message Transfer protocol include message, probe, and report transfer; message redirection; security; distribution list expansion; and content conversion. In X.400, the receiving message transfer agent operates on the message, report, or probe envelope. If the operations are successful, the message is transferred to the next message transfer agent or deposited in a message store. If the operations are unsuccessful, a nondelivery notification is returned. A transfer may be checkpointed so that failure of a transfer does not require retransmitting an entire message.

In SMTP, an interactive messaging dialogue is set up to transfer a message. First, the originator and recipients are verified, and then the message text is sent. Other commands may also be interspersed within a

session. The advantages to the SMTP approach are that the commands and the message format are independent; the same command set could be used to transfer many different types of messages. In addition, the command approach would allow the sender to repair failed addresses, rather than simply transfer nondelivery notifications, as X.400 does.

Message Access Protocols. X.400 defines a full suite of message access protocols for remote message submission, delivery, and retrieval based on a client-service model. Internet standards in this area are also being implemented and deployed. In general, however, proprietary solutions dominate this aspect of messaging, because the mailbox provider can add value by enhancing the message access protocols without affecting message transport interoperability.

Content Architecture. The Internet MIME architecture was intentionally designed to be compatible with the content architecture of X.400. Table VI compares support of different types of body parts by MIME and X.400 as built-in types. Both MIME and X.400 define new body part types — MIME through registration of subtypes, and X.400 through object identifiers. It would be relatively easy to ensure that MIME and X.400 identifiers are registered at the same time and published together to promote interoperability. The Electronic Mail Association is playing a key role in publishing these identifiers.

Internet messaging offers the ability to send messages that request files to be automatically retrieved through its MIME content architecture. However, X.400 does not define features to enable a user agent or message store to automatically return files or messages in reply to a message requesting them.

Security. All PEM services are defined outside the message transfer service, and are independent of the transfer service being used. The current PEM specifications are oriented, both in design and encoding algorithm, toward the Internet 822 message format and SMTP transfer. PEM messages can be carried within X.400 messages, as well as within SMTP, although gateways between X.400 security services and PEM services may break down because of encryption and encoding of security elements within the messages themselves. In contrast, X.400 defines security services at the level of a user agent (e.g., EDI user agent) and a message transfer agent.

PEM was designed to offer a base set of security services needed in messaging between end users. The X.400 set of security services is more comprehensive. Additional user agent services that are supported by X.400, but not by PEM, include proof of delivery, security access management, message flow confidentiality, message sequence integrity, nonrepudiation of delivery, and message security labeling. In the future, PEM could be enhanced to support these services. In addition, PEM does not support the message transfer agent security services of report origin authentication, proof of submission, or nonrepudiation of submission.

As security is incorporated in messaging, particularly for EDI, a single security infrastructure of trusted authorities and certificate authorities may begin to serve both the X.400 and Internet communities.

Management. The approaches to messaging management in the Internet and X.400 suffer from the same divisions as management of other applications, services, and devices. The Internet community is using a Simple Network Management Protocol (SNMP) approach in the definition of managed objects, and for interchange of management information. The X.400 community is following the lead of the ITU by using the Telecommunications Management Network framework with the OSI CMIP. The ITU-T and ISO/International Electrotechnical Commission (IEC's) scope for messaging management is limited to the management of messaging-defined application resources, while the Internet group's scope includes the management of messaging resources outside X.400 and Internet mail. Some technical issues need to be resolved as well. At present, it is uncertain whether CMIP-conformant MIBs tracking X.400 application resources will be manageable by SNMP processes, or if CMIP reporting and event threshold setting services will be effective on SMTP/MIME resources identified in SNMP MIBs. An effort within the International Federation for Information Processing Task Group 6.5 has been established to bridge this gap with a common management and information base model, and with the development of management requirements for ISO and the ITU. Although official channels of communication between the Internet Engineering Task Force (IETF) and the ITU/ISO/IEC are lacking, informal input and concerns expressed by Internet mail management experts have been considered where possible.

While the messaging experts are attempting to cooperate in this area, the work is difficult. Progress has been slow, and will rely on the efforts of SNMP and CMIP

experts to harmonize or bridge their technologies and services in a mutually effective, transparent way.

Future Standards Activities

Messaging technology continues to evolve, and the technology will continue to drive the standards process. The following paragraphs touch on future activities that may be standardized.

Additional Messaging Application Program Interfaces. Application program interfaces for messaging components have been developed by X/OPEN and by the X.400 Application Program Interface (API) Association. (X/OPEN is a registered trademark of X/OPEN Ltd.) It is likely that the XAPI Association's X.400 Common Message Call API[58] will become an ITU-T recommendation by 1996. Outside of messaging, however, frameworks for object interaction through application program interfaces are being developed in the Object Management Group and by several vendors. Additional APIs for messaging will likely be developed to fit these frameworks.

Alerting. As intelligent personal communicating devices become smaller and more portable, there could be increased demand for alerting standards, which would include the format of alerts, the registration of auto-alert actions with a messaging service, and auto-alert actions themselves. These standards would allow paging systems to interact with a larger set of applications.

Asynchronous Access. Traditionally, wide-area X.400 messaging has used synchronous communications protocols, primarily X.25. The MHS Alliance, a consortium of X.400 providers and system developers, has developed a set of profiles for X.400 messaging over previously standardized asynchronous, dialup communication protocols. This specification is now being reviewed by the ITU as Draft Recommendation X.445, and is likely to become an ITU recommendation by mid-1995.

Messaging As a General Protocol Mechanism. One recent trend in messaging enables applications to send and, possibly, receive an application document directly, without exiting the application. These applications are termed *mail-enabled.* With applications and messaging being linked, it would not be surprising if the applications began to use mail not only for interpersonal messaging, but also as a general protocol mechanism to exchange application-specific information. This is happening, to some extent, in the X.400 API Association, which is using messaging as a means to synchronize directory information among distributed directories and to circulate calendaring information among users. For information update and retrieval, as well as other applications, this trend is likely to continue.

Smart Messaging. Smart messaging is an extension of messaging as a general protocol mechanism, pioneered by General Magic and Bell Communications Research (Bellcore). Smart messages are programs that can be executed or interpreted by messaging platforms as they traverse the network. Messaging protocols will not need to be defined by static sets of messaging headers, but rather by a language that allows the user to compose an infinite variety of messages that are interpreted on a remote device for the message recipient.

Universal Mailbox. AT&T has long been a proponent of the universal mailbox, with its unified messaging architecture. Adding multimedia capabilities to the desktop, as well as application program interfaces like Microsoft and Intel's Telephone API, will likely consolidate voice- and text-messaging systems. This will require messaging standards to support access through telephone terminals and, possibly, simplified numeric addressing.

Wireless Messaging. As the technology and implementations of wireless messaging become more widespread, standards for wireless messaging will be promoted. The ITU plans to include wireless messaging in its X.400 series of recommendations in the future.

Conclusion

The basic messaging structures, protocols, and functions have been defined to create a flexible framework for the store-and-forward movement of messages, supporting a wide range of capabilities in both the X.400 and Internet worlds. The technological challenge in standardizing these messaging architectures will be whether to converge, harmonize, or continue to develop them along separate, parallel paths. X.400 was developed to exchange messages among public service providers, MOTIS among computer systems, and Internet messaging among researchers. These distinctions have become increasingly fuzzy, and the need to work globally across these standards will gain importance. AT&T will continue to participate in standardization activities in these forums to facilitate meeting its stated desire to be the preeminent provider of public and private messaging services and products. The addition of standardized

applications and services, provided through messaging, will lead to revolutionary new services that will further extend the breadth of AT&T's messaging offerings into the 21st century. It will also continue to encourage use of AT&T's supporting network technologies. AT&T's strategy of being the leader in messaging standardization will protect the network-operator investments in both Internet and X.400 services and products.

References

1. International Telegraph and Telephone Consultative Committee, Recommendations X.400-X.420, *Message Handling Systems*, 1984.
2. International Telegraph and Telephone Consultative Committee, Recommendations X.400-X.420, *Message Handling Systems*, 1988.
3. ISO 7498, *Information Processing Systems — Open Systems Interconnection — Basic Reference Model*, 1984.
4. International Telegraph and Telephone Consultative Committee, Recommendation F.400, *Message Handling and Directory Services — Operations and Definition of Service*, 1992.
5. International Telegraph and Telephone Consultative Committee, Recommendation X.400, *Message Handling Services — Message Handling System and Service Overview*, 1993.
6. International Telegraph and Telephone Consultative Committee, Recommendation X.402, *Message Handling Systems — Overall Architecture*, 1992.
7. International Telegraph and Telephone Consultative Committee, Recommendation X.411, *Message Handling Systems — Message Transfer System: Abstract Service Definition and Procedures*, 1992.
8. International Telegraph and Telephone Consultative Committee, Recommendation X.413, *Message Handling Systems — Message Store: Abstract Service Definition*, 1992.
9. International Telegraph and Telephone Consultative Committee, Recommendation X.419, *Message Handling Systems — Protocol Specifications*, 1992.
10. International Telegraph and Telephone Consultative Committee, Recommendation X.420, *Message Handling Systems — Interpersonal Messaging System*, 1992.
11. International Telegraph and Telephone Consultative Committee, Recommendation X.435, *Message Handling Systems — Electronic Data Interchange Messaging System*, 1991.
12. International Telegraph and Telephone Consultative Committee, Recommendation X.440, *Message Handling Systems — Voice Messaging System*, 1992.
13. International Telegraph and Telephone Consultative Committee, Recommendation X.480, *Message Handling Systems and Directory Services — Conformance Testing*, 1992.
14. International Telegraph and Telephone Consultative Committee, Recommendation X.481, *P2 Protocol: Protocol Implementation Conformance Statement (PICS) Proforma*, 1992.
15. International Telegraph and Telephone Consultative Committee, Recommendation X.482, *P1 Protocol: Protocol Implementation Conformance Statement (PICS) Proforma*, 1992.
16. International Telegraph and Telephone Consultative Committee, Recommendation X.483, *P3 Protocol: Protocol Implementation Conformance Statement (PICS) Proforma*, 1992.
17. International Telegraph and Telephone Consultative Committee, Recommendation X.484, *P7 Protocol: Protocol Implementation Conformance Statement (PICS) Proforma*, 1992.
18. International Telegraph and Telephone Consultative Committee, Recommendation X.485, *Voice Messaging Protocol Implementation Conformance Statement (PICS) Proforma*, 1992.
19. International Telegraph and Telephone Consultative Committee, Recommendations X.208, *Specification of abstract syntax notation one (ASN.1)*, 1988.
20. International Telegraph and Telephone Consultative Committee, Recommendation X.209, *Specification of basic encoding rules for abstract syntax notation one (ASN.1)*, 1988.
21. International Telegraph and Telephone Consultative Committee, Recommendation F.401, *Message Handling Services: Naming and Addressing for Public Message Handling Services*, 1992.
22. International Telegraph and Telephone Consultative Committee, Recommendations X.500-X.521, *Directory*, 1988.
23. International Telegraph and Telephone Consultative Committee, Recommendation X.228, *Reliable Transfer: Protocol Specifications*, 1988.
24. Data Interchange Standards Association, *X12: American National Standards for Electronic Data Interchange*, Version 003000-1992, Data Interchange Standards Association.
25. International Telegraph and Telephone Consultative Committee, Recommendation G.721, *32 kbit/s Adaptive Differential Pulse Code Modulation*, 1992.
26. International Organization for Standardization, ISO Committee Draft 10918-1, *Digital compression and coding of continuous still images — Part 1: Requirements and guidelines*, 1991.
27. International Organization for Standardization, ISO Committee Draft 11172, *Information Technology — Coding of moving pictures and associated video for digital store media up to about 1.5 Mbit/s*, 1990.
28. International Telegraph and Telephone Consultative Committee, Recommendation F.435, *Message Handling: Electronic Data Interchange Messaging Service*, 1991.
29. International Telegraph and Telephone Consultative Committee, Recommendation F.440, *Message Handling Services: The Voice Messaging Service*, 1992.
30. ISO/IEC 9506-1, International Organization for Standardization, *Information Technology — Open Systems Interconnection — Common Management Information Protocol, Part 1: Specification*, 1991.
31. Jon Postel, Request for Comments 821, *Simple Mail Transfer Protocol*, 1982.
32. M. T. Rose, Request for Comments 1463, *Post Office Protocol: Version 3*, 1993.
33. M. R. Crispin, Request for Comments 1176, *Interactive Mail Access Protocol: Version 2*, 1990.
34. J. Rice, Request for Comments 1203, *Interactive Mail Access Protocol: Version 3*, 1991.
35. M. L. Lambert, Request for Comments 1056, *PCMAIL: A Distributed Mail System for Personal Computers*, 1988.
36. D. Crocker, Request for Comments 822, *Standard for the Format of ARPA Internet Text Messages*, August 1982.
37. International Organization for Standardization, ISO 3166, *Codes for the representation of names of countries*, 1993.
38. D. Crocker, et al., Request for Comments 724, *Proposed official standard for the format of ARPA Network messages*, 1977.
39. J. Klensin, et al., Request for Comments 1426, *SMTP Service Extension for 8-bit MIME Transport*, February 1993.
40. S. T. Kent, "Internet Privacy Enhanced Mail," *Communications of*

the ACM 36(6): 48-60, August 1993.
41. J. Linn, Request for Comments 1421, *Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures*, February 1993.
42. S. T. Kent, Request for Comments 1422, *Privacy Enhancement for Internet Electronic Mail: Part II: Certificate-Based Key Management*, February 1993.
43. D. Balenson, Request for Comments 1423, *Privacy Enhancement for Internet Electronic Mail: Part III: Algorithms, Modes, and Identifiers*, February 1993.
44. B. S. Kaliski, Request for Comments 1424, *Privacy Enhancement for Internet Electronic Mail: Part IV: Key Certification and Related Services*, February 1993.
45. U.S. Government, U.S. Federal Information Processing Standard 46-1, *Data Encryption Standard*, 1988.
46. B. S. Kaliski, Request for Comments 1319, *The MD2 Message-Digest Algorithm*, April 1992.
47. R. Rivest, Request for Comments 1321, *The MD5 Message-Digest Algorithm*, April 1992.
48. N. Borenstein and N. Freed, Request for Comments 1521, *MIME (Multipurpose Internet Mail Extensions): Mechanisms for specifying and describing the format of Internet message bodies*, September 1993.
49. J. B. Postel and J. K. Reynolds, Request for Comments 959, *File Transfer Protocol*, 1985.
50. K. R. Sollins, Request for Comments 783, *Trivial File Transfer Protocol (TFTP) Protocol (Revision 2)*, 1981.
51. International Telegraph and Telephone Consultative Committee, Recommendation D.36, *General Accounting Principles Applicable to Message Handling Systems*, 1991.
52. G. Vaudreuil, Internet Draft, "An Extensible Message Format for Delivery Notifications," January 1994.
53. N. Borenstein, Request for Comments 1523, *The text/enriched MIME content type*, September 1993.
54. *Graphics Interchange Format (Version 89a)*, Compuserve, Inc., Columbus, Ohio, 1990.
55. ISO/IEC 8859, International Organization for Standardization, *Information processing — 8-bit single-byte coded graphic character sets*, 1987.
56. International Telegraph and Telephone Consultative Committee, Recommendation G.711, *Pulse Code Modulation (PCM) of Voice Frequencies*, 1972.
57. Adobe Systems, Inc., *PostScript Language Reference Manual*, Addison-Wesley, 1985.
58. X.400 API Association, *Common Messaging Call API*, Version 1.0, June 1993.

(Manuscript approved April 1994)

Stephen J. Griesmer *is a technical manager in the Applications Architecture Department of AT&T Bell Laboratories in Holmdel, New Jersey, where he is responsible for developing distributed processing architectures and application standards. Mr. Griesmer joined AT&T in 1982 after receiving a B.S.E.E. in electrical engineering and computer science from Princeton University, New Jersey, and an M.Sc. in interdisciplinary studies from the University of British Columbia, Vancouver, Canada.*

Richard W. Jesmajian *is a member of technical staff in the Applications Architecture Department of AT&T Bell Laboratories in Holmdel, New Jersey. He is responsible for messaging, directory standards, and financial electronic data interchange (EDI) standards. He is also Chair of the International Telecommunications Union Telecommunications Standardization Sector X.400 Protocol Group (Question 14 of Study Group 7). Mr. Jesmajian received a B.S. in mathematics and education from Union College, Barbourville, Kentucky, and an M.S. in computer science from Fairleigh Dickinson University, Teaneck, New Jersey. He joined AT&T in 1979.*