INTERNATIONAL TELECOMMUNICATION UNION

# CCITT

THE INTERNATIONAL
TELEGRAPH AND TELEPHONE
CONSULTATIVE COMMITTEE

**RED BOOK**

---

**VOLUME VIII – FASCICLE VIII.7**

# DATA COMMUNICATION NETWORKS
# MESSAGE HANDLING SYSTEMS

**RECOMMENDATIONS X.400-X.430**

---

**VIIITH PLENARY ASSEMBLY**
MALAGA-TORREMOLINOS, 8-19 OCTOBER 1984

Geneva 1985

INTERNATIONAL TELECOMMUNICATION UNION

# CCITT

THE INTERNATIONAL
TELEGRAPH AND TELEPHONE
CONSULTATIVE COMMITTEE

**RED BOOK**

---

**VOLUME VIII – FASCICLE VIII.7**



# DATA COMMUNICATION NETWORKS
# MESSAGE HANDLING SYSTEMS

**RECOMMENDATIONS X.400-X.430**

---

**VIIITH PLENARY ASSEMBLY**
MALAGA-TORREMOLINOS, 8-19 OCTOBER 1984

Geneva 1985

ISBN 92-61-02361-4

## CONTENTS OF THE CCITT BOOK
## APPLICABLE AFTER THE EIGHTH PLENARY ASSEMBLY (1984)

### RED BOOK

**Volume IV** – *(4 fascicles, sold separately)*

FASCICLE IV.1 – Maintenance; general principles, international transmission systems, international telephone circuits. Recommendations M.10-M.762 (Study Group IV).

FASCICLE IV.2 – Maintenance; international voice frequency telegraphy and fascimile, international leased circuits. Recommendations M.800-M.1375 (Study Group IV).

FASCICLE IV.3 – Maintenance; international sound programme and television transmission circuits. Series N Recommendations (Study Group IV).

FASCICLE IV.4 – Specifications of measuring equipment. Series 0 Recommendations (Study Group IV).

**Volume V** – Telephone transmission quality. Series P Recommendations (Study Group XII).

**Volume VI** – *(13 fascicles, sold separately)*

FASCICLE VI.1 – General Recommendations on telephone switching and signalling. Interface with the maritime mobile service and the land mobile services. Recommendations Q.1-Q.118 *bis* (Study Group XI).

FASCICLE VI.2 – Specifications of Signalling Systems Nos. 4 and 5. Recommendations Q.120-Q.180 (Study Group XI).

FASCICLE VI.3 – Specifications of Signalling System No. 6. Recommendations Q.251-Q.300 (Study Group XI).

FASCICLE VI.4 – Specifications of Signalling Systems R1 and R2. Recommendations Q.310-Q.490 (Study Group XI).

FASCICLE VI.5 – Digital transit exchanges in integrated digital networks and mixed analogue-digital networks. Digital local and combined exchanges. Recommendations Q.501-Q.517 (Study Group XI).

FASCICLE VI.6 – Interworking of signalling systems. Recommendations Q.601-Q.685 (Study Group XI).

FASCICLE VI.7 – Specifications of Signalling System No. 7. Recommendations Q.701-Q.714 (Study Group XI).

FASCICLE VI.8 – Specifications of Signalling System No. 7. Recommendations Q.721-Q.795 (Study Group XI).

FASCICLE VI.9 – Digital access signalling system. Recommendations Q.920-Q.931 (Study Group XI).

FASCICLE VI.10 – Functional Specification and Description Language (SDL). Recommendations Z.101-Z.104 (Study Group XI).

FASCICLE VI.11 – Functional Specification and Description Language (SDL), annexes to Recommendations Z.101-Z.104 (Study Group XI).

FASCICLE VI.12 – CCITT High Level Language (CHILL). Recommendation Z.200 (Study Group XI).

FASCICLE VI.13 – Man-Machine Language (MML). Recommendations Z.301-Z.341 (Study Group XI).

**Volume VII** – *(3 fascicles, sold separately)*

FASCICLE VII.1 – Telegraph transmission. Series R Recommendations (Study Group IX). Telegraph services terminal equipment. Series S Recommendations (Study Group IX).

FASCICLE VII.2 – Telegraph switching. Series U Recommendations (Study Group IX).

FASCICLE VII.3 – Terminal equipment and protocols for telematic services. Series T Recommendations (Study Group VIII).

**Volume VIII** – *(7 fascicles, sold separately)*

FASCICLE VIII.1 – Data communication over the telephone network. Series V Recommendations (Study Group XVII).

FASCICLE VIII.2 – Data communication networks: services and facilities. Recommendations X.1-X.15 (Study Group VII).

FASCICLE VIII.3 – Data communication networks: interfaces. Recommendations X.20-X.32 (Study Group VII).

FASCICLE VIII.4 – Data communication networks: transmission, signalling and switching, network aspects, maintenance and administrative arrangements. Recommendations X.40-X.181 (Study Group VII).

FASCICLE VIII.5 – Data communication networks: Open Systems Interconnection (OSI), system description techniques. Recommendations X.200-X.250 (Study Group VII).

FASCICLE VIII.6 – Data communication networks: interworking between networks, mobile data transmission systems. Recommendations X.300-X.353 (Study Group VII).

FASCICLE VIII.7 – Data communication networks: message handling systems. Recommendations X.400-X.430 (Study Group VII).

**Volume IX** – Protection against interference. Series K Recommendations (Study Group V). Construction, installation and protection of cable, and other elements of outside plant. Series L Recommendations (Study Group VI).

**Volume X** – *(2 fascicles, sold separately)*

FASCICLE X.1 – Terms and definitions.

FASCICLE X.2 – Index of the Red Book.

PAGE INTENTIONALLY LEFT BLANK

PAGE LAISSEE EN BLANC INTENTIONNELLEMENT

## CONTENTS OF FASCICLE VIII.7 OF THE RED BOOK

**Recommendations X.400 to X.430**

**Data communication networks**
**Message handling systems**



---

PRELIMINARY NOTES

1 The Questions entrusted to each Study Group for the Study Period 1985-1988 can be found in Contribution No. 1 to that Study Group.

2 In this fascicle, the expression "Administration" is used for shortness to indicate both a telecommunication Administration and a recognized private operating agency.

3 The status of annexes and appendices attached to the Series X Recommendations should be interpreted as follows:

- an *annex* to a Recommendation forms an integral part of the Recommendation;
- an *appendix* to a Recommendation does not form part of the Recommendation and only provides some complementary explanation or information specific to that Recommendation.

FASCICLE VIII.7

**Recommendations X.400 to X.430**

# DATA COMMUNICATION NETWORKS: MESSAGE HANDLING SYSTEMS

PAGE INTENTIONALLY LEFT BLANK

PAGE LAISSEE EN BLANC INTENTIONNELLEMENT

**Recommendation X.400**

## MESSAGE HANDLING SYSTEMS: SYSTEM MODEL-SERVICE ELEMENTS

*(Malaga-Torremolinos, 1984)*

The establishment in various countries of telematic services and computer-based store-and-forward message services in association with public data networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT,

*considering*

(a) the need for interpersonal messaging and message transfer services;

(b) the need to transfer messages of different types having a large variety of formats;

(c) that Recommendation X.1 includes specific user classes of service and X.10 categories of access; and Recommendation X.2 defines optional user facilities for public data network;

(d) that Recommendations X.20, X.20 *bis*, X.21, X.21 *bis*, X.22, X.25, X.28 and X.29 define DTE/DCE interfaces to public data networks;

(e) that Recommendations X.300 and X.121 permit international connections between public data networks;

(f) that the F-series of Recommendations defines telematic services and that the T-series of Recommendations defines terminal equipment and control procedures for telematic services;

(g) that the V-series of Recommendations provides the means for data communication over the telephone network;

(h) that Recommendation X.200 defines the reference model of open systems interconnection for CCITT applications;

(i) that a set of Recommendations describes various aspects of message handling: X.400, X.401, X.408, X.409, X.410, X.411, X.420, and X.430,

*unanimously declares*

(1) that the general aspects of message handling systems are described in § 1;

(2) that the functional model for message handling systems is specified in § 2;

(3) that the naming and addressing principles are specified in § 3;

(4) that the service elements are specified in § 4;

(5) that the layered representation of the message handling model is specified in § 5.

CONTENTS

## 1 Purpose and scope

This Recommendation is one of a series of Recommendations and describes the system model and service elements of the message handling system (MHS). This Recommendation defines the *message handling (MH) services* that Administrations provide to enable subscribers to exchange messages on a store-and-forward basis. Two MH services are provided. The interpersonal messaging (IPM) service supports interpersonal communication, including communication with existing CCITT Telex and Telematic services. The message transfer (MT) service supports general, application-independent message transfer.

Other Recommendations in the series contain additional information essential to a thorough understanding of the MHS. Recommendation X.401 lists the basic service elements and optional user facilities. Recommendation X.411 specifies the protocol aspects of the MT service. Recommendation X.420 specifies the protocol aspects of the IPM service. Recommendation X.409 defines the notation and representational technique used to specify and encode MHS protocols. Recommendation X.408 specifies the algorithms that the MHS uses when converting between different types of encoded information. Recommendation X.430 describes how Teletex terminals access the MHS. Recommendation X.410 describes the general techniques used for MHS protocols and the way in which standard Open Systems Interconnection (OSI) protocols are used to support MHS applications.

The MH services and provided by means of the Message Handling System (MHS). § 2 of this Recommendation introduces a functional model of message handling systems and its applicability to a variety of physical and organizational configurations. § 3 describes the system's provisions for naming and addressing. § 4 describes the MH Service elements in detail. § 5 presents a layered model of the MHS, using the Reference Model of Open Systems Interconnection for CCITT Applications (Recommendation X.200) hereafter called the *OSI Reference Model.*

## 2 Message handling system model

### 2.1 *Purpose*

The MHS model serves as a tool to aid in the development of Recommendations on MH services. It comprises several different functional components that work together to provide those services. The model can be applied to a number of different physical and organizational configurations.

The MHS model uses the techniques of the OSI Reference Model to formally define the layered communication structure used between the model's functional components. The resulting layered representation determines the requirements for Recommendations on protocols, and clarifies the relationship between those protocols and the defined service elements. This aspect of the model is described in § 5.

## 2.2 *Description of the MHS model*

### 2.2.1 *Overview*

A functional view of the MHS model is shown in Figure 1/X.400. In this model, a *user* is either a person or a computer application. A user is referred to as either an *originator* (when sending a message) or a *recipient* (when receiving one). MH Service elements define the set of message types and the capabilities that enable an originator to transfer messages of those types to one or more recipients.

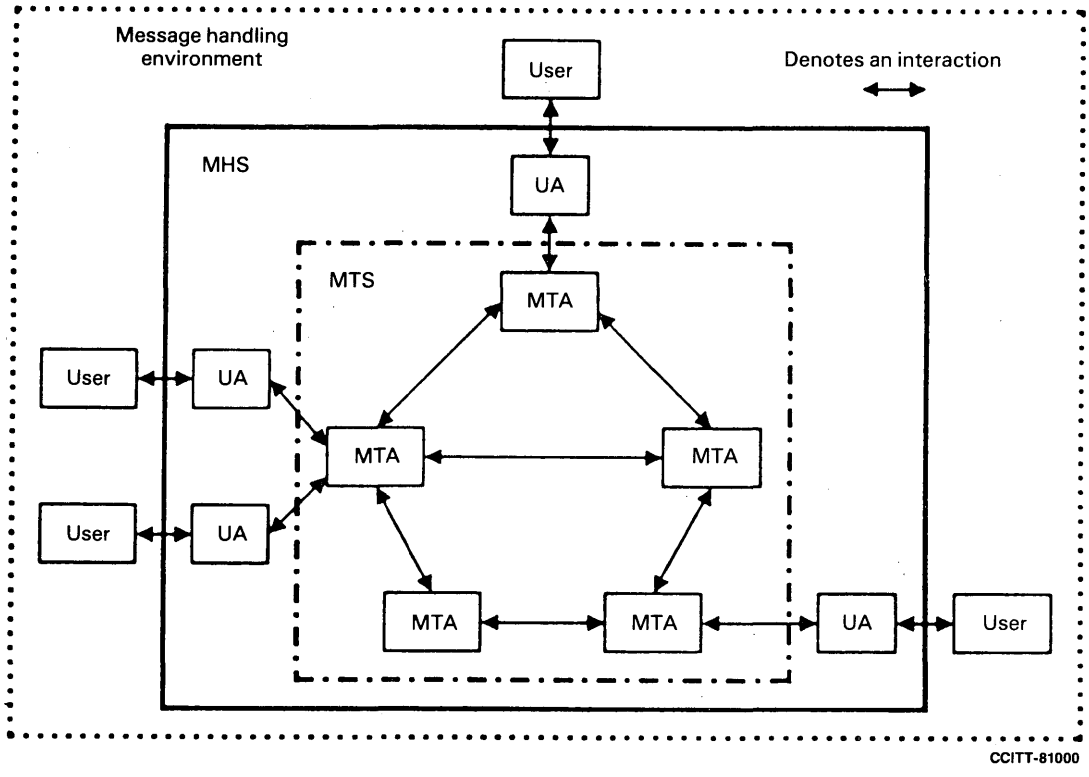



FIGURE 1/X.400

**Functional view of the MHS model**

An originator prepares messages with the assistance of his User Agent. A user agent (UA) is an application process that interacts with the message transfer system (MTS) to submit messages. The MTS delivers to one or more recipient UAs the messages submitted to it. Functions performed solely by the UA and not standardized as part of the MH Service elements are called local UA functions.

The MTS comprises a number of message transfer agents (MTAs). Operating together, the MTAs relay messages and deliver them to the intended recipient UAs, which then make the messages available to the intended recipients.

The collection of UAs and MTAs is called the message handling system (MHS). The MHS and all of its users are collectively referred to as the message handling environment.

The basic structure of messages is shown in Figure 2/X.400. The *envelope* carries information to be used when transferring the message. The three types of envelope are identified in § 2.2.2. The *content* is the piece of information that the originating UA wishes delivered to one or more recipient UAs.

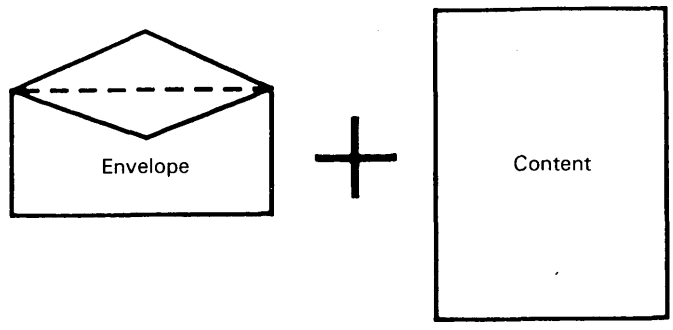



FIGURE 2/X.400

**Basic message structure**

2.2.2 *Message transfer system and its users*

The MTS provides the general, application-independent, store-and-forward message transfer service. The service elements are described in § 4.1.

2.2.2.1 *Submission and delivery*

The MTS provides the means by which UAs can exchange messages. There are two basic interactions between MTAs and UAs:

1) The *submission* interaction is the means by which an originating UA transfers to an MTA the content of a message plus the submission envelope. The *submission envelope* contains the information the MTS requires to provide the requested service elements.
2) The *delivery* interaction is the means by which the MTA transfers to a recipient UA the content of a message plus the delivery envelope. The *delivery envelope* contains information related to delivery of the message.

In the submission and delivery interactions, responsibility for the message is passed between the MTS and the UA. The submission and delivery interactions are specified in Recommendation X.411.

2.2.2.2 *Relaying*

Each MTA relays the message to another MTA until the message reaches the recipient's MTA, which then delivers it to the recipient UA using the delivery interaction.

The *relaying* interaction is the means by which one MTA transfers to another the content of a message plus the relaying envelope. The *relaying envelope* contains information related to the operation of the MTS plus the service elements requested by the originating UA.

MTAs transfer messages containing any type of binary coded information. MTAs neither interpret nor alter the content of messages except when a UA requests that they do so by means of a specific service element.

2.2.2.3 *User agent*

The UA uses the MT Service provided by the MTS. A UA is a set of computer application processes that, as a minimum, contain the functions necessary to interact with the MTS using the submission and delivery procedures defined in Recommendation X.411.

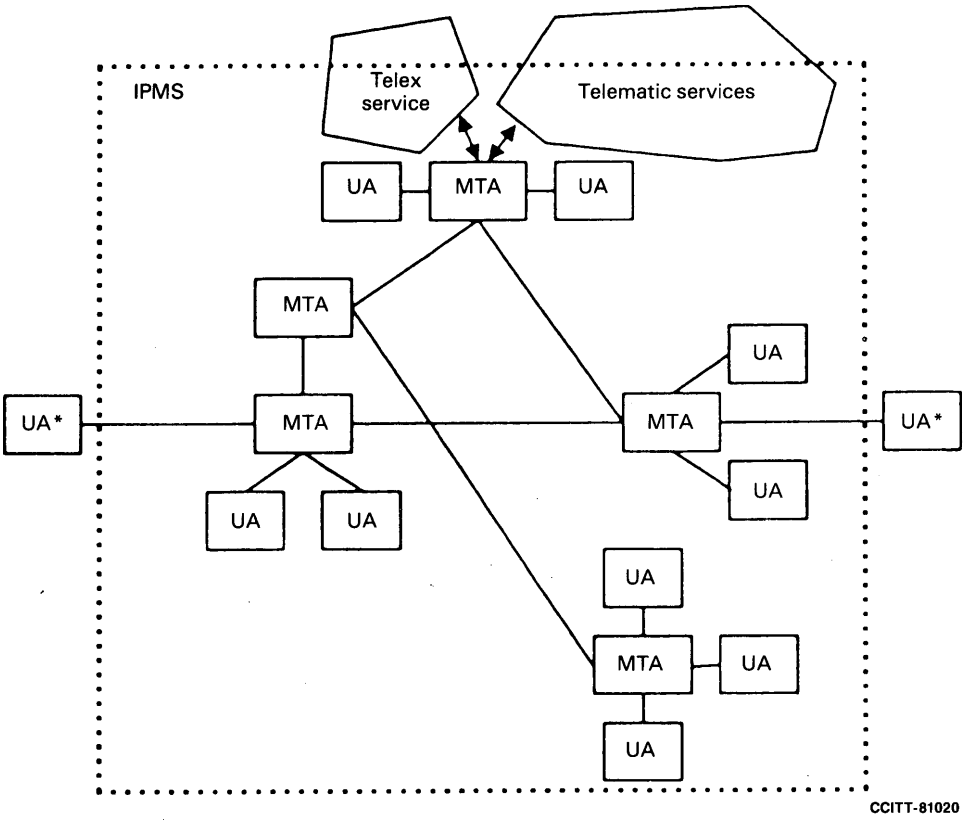
UAs are grouped into classes based on the type of the content of the messages they can handle. The MTS provides a UA with the ability to identify its class when sending messages to other UAs. UAs within a given class are referred to as cooperating UAs since they cooperate with one another to enhance the communication among their respective users.

2.2.3 *Interpersonal messaging system*

2.2.3.1 *System overview*

The Interpersonal messaging system (IPMS) provides an individual with services to assist him in communicating with other individuals. The service elements are defined in § 4.2.

As shown in Figure 3/X.400, the IPMS comprises the MTS, a specific class of cooperating UAs, and access to Telex and CCITT Telematic services. It provides its users (originators and recipients) with the Interpersonal Messaging (IPM) service. The users of the IPMS are typically people and constitute a single community.




* These UAs use the services of the MTS but are not in the class of UAs that support the IPM service. Therefore, they cannot meaningfully communicate with IPM UAs.

FIGURE 3/X.400

**The Interpersonal messaging system**

*Note* – Other classes of UAs may be defined, however, these UAs would form a separate community unable to communicate meaningfully with IPM UAs.

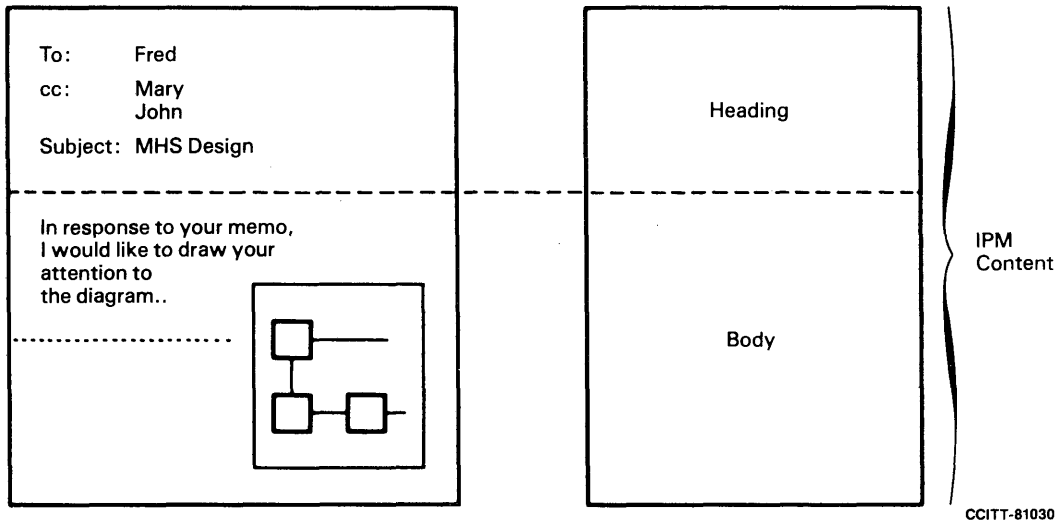
To provide the IPM service, UAs use service elements provided by the MTS. An IPM UA should therefore:

- provide the functions necessary to prepare messages;
- perform the submission interaction with the MTS;
- perform the delivery interaction with the MTS;
- perform the functions necessary to present messages to its user;
- provide functions to cooperate with other UAs in order to help its user deal with messages;
- perform additional message preparation and manipulation functions.

Optionally, the IPM UA may provide local UA functions not subject to standardization by CCITT. For example, it might provide word processing facilities, or database facilities for storing and retrieving previously received messages.

2.2.3.2 *Content type within the IPMS*

The IPM class of UAs create messages containing a content of the IPMS content type. The specific content that is sent from one IPM UA to another as a result of an originator composing and sending a message is called an *IP-message.* A simple example of the relationship between a typical office memorandum, or memo, and the corresponding IP-message structure is shown in Figure 4/X.400.




*Note* – The IP-message is conveyed with an envelope when being transferred through the MTS.

FIGURE 4/X.400

**Relationship between a memo and an IP-message**

In this example, the memo contains information (for example, to:, cc:, and subject:, etc.) which are provided by the user and transformed by the IPM UA to the part of the IP-message called the *heading.*

The information the user wishes to communicate (that is, the body of the memo) is contained within that part of the IP-message called the *body.* In this example, the body contains two types of encoded information: text and facsimile. In general, a body may consist of a number of different encoded information types such as voice, text, facsimile and graphics.

The protocol for the IPM content type is specified in Recommendation X.420. The procedures for converting between different types of encoded information are contained in Recommendation X.408.

2.2.4 *Telematic services and message handling*

Users of telex and the telematic services and users of the IPM service must be able to communicate with one another.

To accommodate telex and telematic terminals, additional functionality is provided within the message handling system for the following reasons:

a) To ensure that there exists across the whole user community, a service that is consistent with the message handling, telex and telematic service definitions.

b) To perform the appropriate protocol translation in order that users can communicate freely.

c) To provide, where requested by the subscriber, additional functionality that is under the user's control and which supplements the functionality of the connected terminal (for example, by providing long-term message storage).

The functionality within the MHS for teletex terminals is defined in Recommendation X.430. The functionality for telex and other telematic services is for further study.

*Note* – Videotex is viewed as an interactive, rather than a messaging terminal. As such, it is seen as an I/O device for interacting with a UA to display the contents of messages containing videotex graphics. The IPM service permits the inclusion of videotex graphics in the content of a message.

## 2.3 *Application of the MHS model*

### 2.3.1 *Physical mapping*

Users access UAs for message processing purposes, for example, to create, present, or file messages. Certain UA implementations will provide storage in which users can manage outgoing and incoming messages. While processing messages, the user interacts with his UA via an input/output device or process (for example, keyboard, display, printer, or facsimile equipment). A UA can be implemented as a set of computer processes in a processing system or as an intelligent terminal.

A UA and MTA may be implemented in the same system (co-resident UA and MTA). In this case, the UA accesses the MT service elements by interacting directly with the MTA in the same system. Alternatively, a UA may be implemented in a physically separate system (stand-alone UA). In this case, the UA must communicate with the MTA in the other system via standardized protocols specified for message handling. It is also possible for an MTA to be implemented in a system without UAs (stand-alone MTA).

Some possible physical configurations are shown in Figures 5/X.400 to 7/X.400. The different physical systems may be connected by means of dedicated lines or switched network connections.

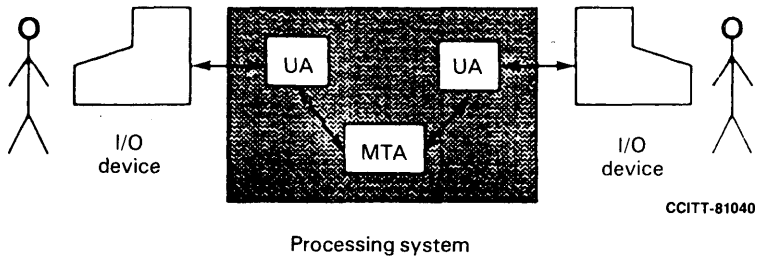



FIGURE 5/X.400

**Co-resident UA and MTA**

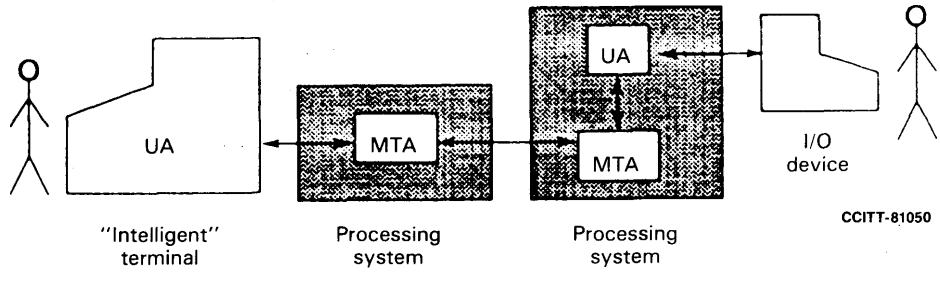



FIGURE 6/X.400

**Co-resident and stand-alone UA and MTA**

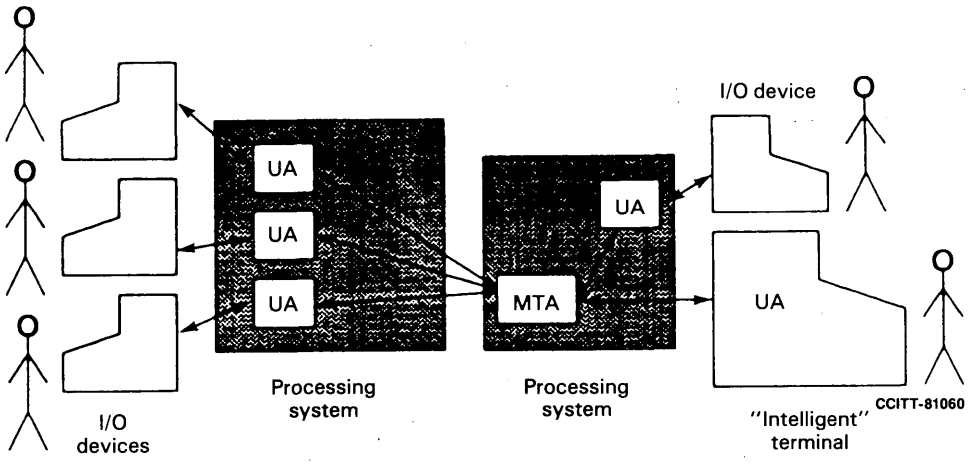



FIGURE 7/X.400

**Combination of co-resident and stand-alone UA and MTA**

2.3.2 *Organizational mapping*

This subsection defines the various roles that an Administration or organization may play in providing MH services. An organization may be a company or a non-commercial organization.

2.3.2.1 *Message handling boundaries*

The collection consisting of at least one MTA and zero or more UAs owned by an Administration or organization constitutes a management domain (MD). The MD managed by an Administration is called an Administration management domain (ADMD). The MD managed by an organization is called a Private Management Domain (PRMD). An MD provides the MT and IPM services in accordance with the classification of optional user facilities found in Recommendation X.401.

An Administration may provide access for its subscribers to the ADMD at one or more of the following boundaries, each of which is described below:

- user to Administration supplied UA
- private UA to Administration MTA
- private MTA to Administration MTA

2.3.2.2 *Administration-supplied UA*

Two cases are possible as shown in Figure 8/X.400. In the first case, the subscriber has only an I/O device (for example, a teletypewriter or telephone) and interacts with the Administration-supplied UA via a man/machine dialogue. In addition, an Administration may choose to supply intelligent stand-alone terminals containing UA functionality providing the IPM service.

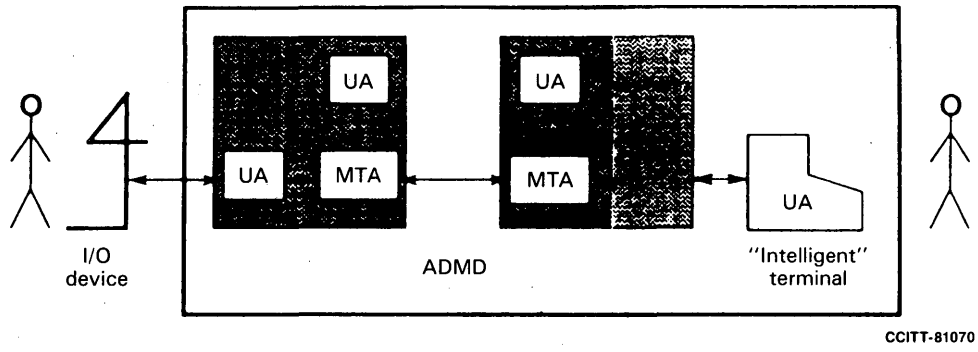



FIGURE 8/X.400

**Administration-supplied UA**

2.3.2.3 *Private UA to Administration MTA*

In this case, the subscriber has a private stand-alone UA (for example, a workstation or personal computer). The UA interacts with the Administration-supplied MTA using the submission and delivery procedures to obtain MT Service. If the subscriber wishes to be part of the IPM community, his UA must conform to that class of UA.

A private, stand-alone UA is not in a MD but is *associated* with an MD, as shown in Figure 9/X.400.

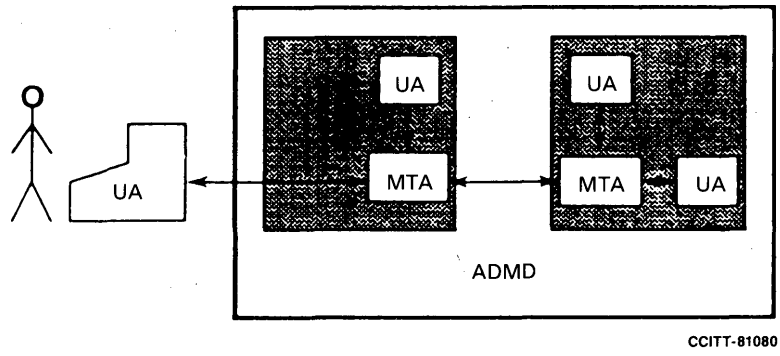



FIGURE 9/X.400

**Private UA to Administration MTA**

2.3.2.4 *Private MTA to Administration MTA*

An Administration's subscriber may have an MTA (or a number of MTAs) and one or more User Agents. These subscriber UAs may be integrated into the same system as one of its MTAs or may be associated stand-alone UAs. The subscriber's MTA(s) and UA(s) form a Private Management Domain (PRMD) and may interact with the Administration Management Domain (ADMD) on an MD-to-MD (MTA-to-MTA) basis.

A Private MD is considered to exist entirely within one country. Within that country, a PRMD may have access to one or more ADMDs, as shown in Figure 10/X.400. However, with respect to a specific interaction between a PRMD and an ADMD (such as when a message is transferred between MDs), the PRMD is considered to be associated only with the single ADMD. A PRMD will not act as a relay between two ADMDs.

When an ADMD interacts with a PRMD, the ADMD takes responsibility for the actions of the PRMD which are related to the interaction. In addition to ensuring that the PRMD properly provides the Message Transfer Service, the ADMD is responsible for ensuring that the accounting, logging, quality of service, and other related operations of the PRMD are correctly performed.

*Note* – It should be recognized that the provision of support for private messaging systems by CCITT members falls within the framework of national regulations. Thus, the technical possibilities described in this section may or may not be offered by an Administration which provides Message Handling services.

## 2.4 *Directory function*

The specification of directories in the Message Handling functional model is for further study.

# 3 Naming and addressing

The intent of this section is to explain the concept of originator/recipient (O/R) name, present the selection of O/R name attributes and forms, and introduce the concepts of routing, distribution list, and directory with respect to names and addresses.

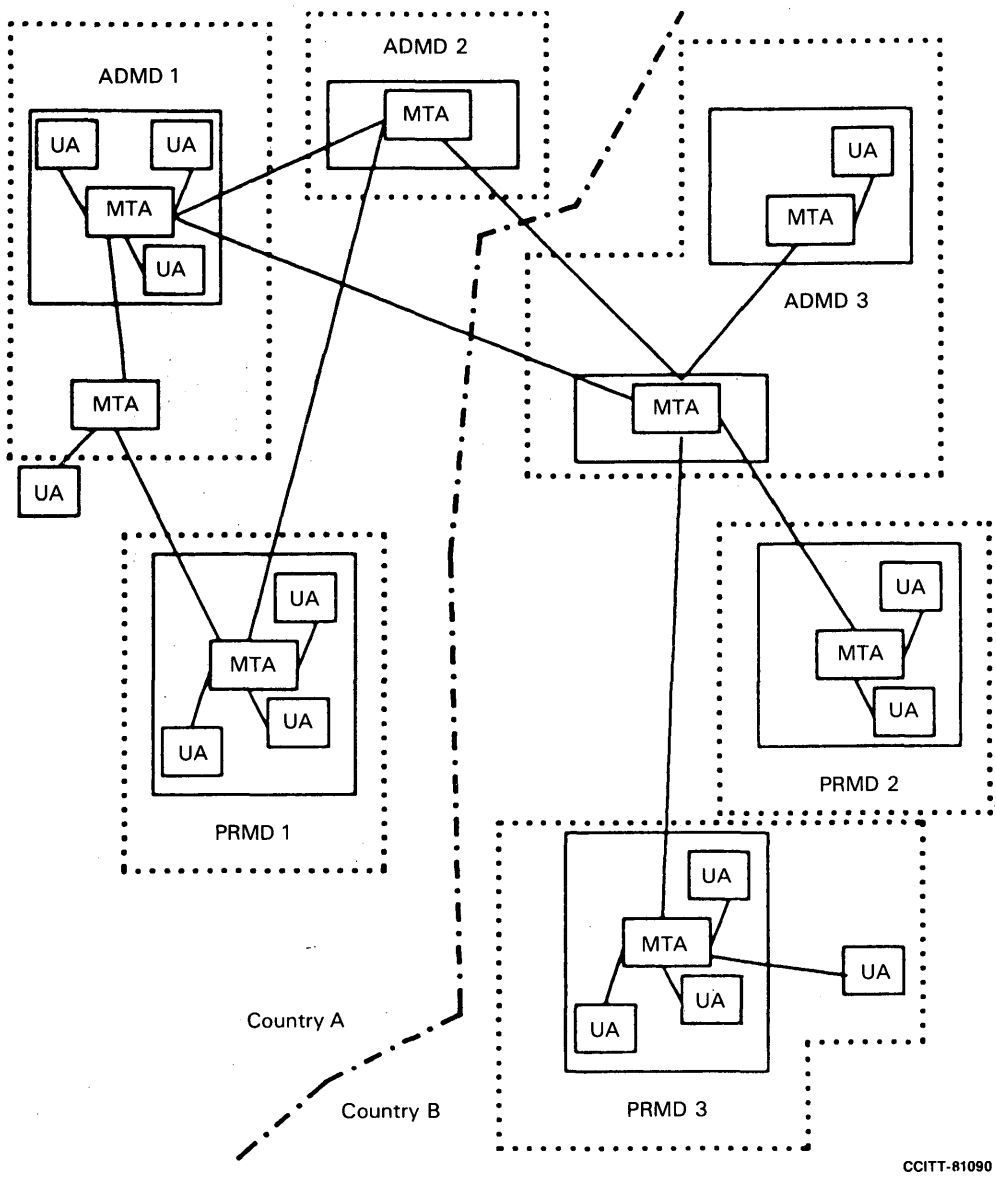



FIGURE 10/X.400

**Administration and private management domains**

## 3.1 *Concepts and terms*

Two kinds of names, primitive and descriptive, are identified in the discussion of MHS.

### 3.1.1 primitive names

A primitive name is a name assigned by a naming authority to a specific entity. A naming authority is simply a source of names. The only constraint imposed upon the naming authority is that it may never hand out the same name twice. For example, an "employee number" is a primitive name where the employing organization is the naming authority.

### 3.1.2 **descriptive names**

A descriptive name denotes exactly one user in the MHS. For example, consider the following description relevant to MHS:

The Vice President of Marketing of the ABC Corporation

If there are no such users or several, the description is not a descriptive name. However, if there is exactly one such user, the description denotes precisely that user and, therefore, is a descriptive name for him.

Primitive names typically figure as components of descriptive names. Since primitive names are only unambiguous within a limited context (that is, that of the naming authority), a single primitive name is in general not sufficient for globally unique identification.

### 3.1.3 *Attributes and attribute lists*

A descriptive name identifies an entity by specifying one or more of its attributes. The set of attributes and their associated values form an attribute list.

In the MHS context, particular multi-valued attributes are permitted. That is, an attribute within the attribute list that constitutes a particular descriptive name may have more than one value. Which attribute types may have multiple values is decided by the individual MD responsible for the operation of the naming authority.

### 3.1.4 *Originator/Recipient names*

In the MHS context, the principal entity that requires naming is the user (the originator and recipient of messages). However, users are outside the MHS and therefore a descriptive name of a user is applied to the user's UA. This is called the O/R name.

To submit a message for delivery, the originating UA must provide the MTS with the name of the UA for each of the message's intended recipients.

For the present, an O/R name is restricted to be an attribute list.

### 3.1.5 *Originator/Recipient addresses*

Addresses are generally used to specify the geographical locations of buildings, the logical locations of components of information processing systems, and so forth. That is, an address usually identifies an entity by specifying where it is, rather than what it is.

An O/R address is a descriptive name for a UA that has certain characteristics that help the MTS to locate the UA's point of attachment. Thus, every O/R address is an O/R name, but not every O/R name is an O/R address.

Since an O/R address is also an O/R name, it could be used, for example, by the originator to identify the intended recipient of a message. Doing so has both advantages and disadvantages. Its principal advantage is that the directory lookup operation is bypassed, which may reduce the delay and/or cost associated with message delivery. In addition, the use of O/R addresses as names will be useful during the early phases of MHS implementation, before international directories capable of mapping names into addresses are in place. The principal disadvantage of using O/R addresses in lieu of O/R names is that the former are much harder for human originators to discover and remember. It should also be noted that O/R addresses are more subject to change than other forms of O/R names.

## 3.2 *Originator/Recipient (O/R) attributes*

### 3.2.1 *Construction of O/R names*

It is an objective that an originator be able to provide a descriptive name for each recipient of a message using information commonly known about that user. This Recommendation specifies a set of standard attributes from which these O/R names can be constructed.

Each MD must ensure that every UA in the MD has at least one name. An MD does not necessarily have to utilize all attribute types in its MD when creating O/R names. However, it must allow its users to construct names using attributes utilized by other MDs.

Four categories of standard attributes have been identified. These categories, along with examples of possible attributes in each category, are given below. Those attributes marked with an asterisk (*) are CCITT-defined attributes and are supported by the MHS protocols. Other possible attributes and perhaps further categories of attributes are for further study:

3.2.1.1 *Personal attributes*

*Examples:* Personal name *
(Surname, Given name(s), Initials, Generational qualifier (for example, "Jr."))

3.2.1.2 *Geographical attributes*

*Examples:* Street name and number
Town name
Region name
Country name *

3.2.1.3 *Organizational attributes*

*Examples:* Organization name *
Organizational unit *
Position or role

3.2.1.4 *Architectural attributes*

*Examples:* X.121 address *
Unique UA identifier (only numeric values) *
Administration management domain name *
Private management domain name *

The MHS protocols also support *domain-defined attributes* whose types are not defined by CCITT. The use of standard attributes will facilitate the identification of recipients by originators. It is recognized that existing message systems do not currently use all of the existing CCITT-defined attributes. The domain-defined attribute has been defined specifically to accommodate the naming conventions of these existing systems and to facilitate interworking with them while they evolve to the use of the CCITT-defined attributes. It is intended that domain-defined attributes be used only for an interim period.

3.2.2 *Base attribute sets*

A base attribute set is a minimum set of attributes whose values unambiguously identify a particular MD. Every O/R name must specify values for all of the attributes in some base attribute set. This requirement enables the MTS to determine the MD that serves the designated user. Depending upon the particular base attribute set included in an O/R name, the MTS may require a directory service – possibly spanning the MHS – to determine the MD.

*Note* – Values for at least one attribute in each base attribute set are assigned by an MHS-wide naming authority.

3.3 *Initial selection of O/R names for MHS*

3.3.1 *Choosing base attribute sets*

As suggested in § 3.2 above, several base attribute sets may be specified for MHS. Each set will be chosen to meet a particular need, and each MD may support those sets most suited to its users. The following candidate base attribute sets are of special interest:

Commercial: organization and country names

Residential: region and country names

Architectural: country and MD names

Terminal-oriented: X.121 address, telex address, or telematic terminal identification

MDs serving commercial users might support the commercial base attribute set and the following additional attributes: given name, surname, position or role, organization name, and organizational unit. MDs serving residential users might support the residential base attribute set and the following: given name, surname, street name and number, and town name. In addition, some MDs might support the architectural base attribute set and the unique UA identifier attribute.

For initial service, two base attribute sets (architectural and terminal-oriented) will be supported by every MD. Support means the following:

- A relaying MD must be able to accept a message using either of the base attribute sets and relay the message to another MD.
- The MD of a recipient UA will be identified by at least one base attribute set of the MD's choosing.
- Users must be able to designate recipients using either of the two base attribute sets.

The terminal-oriented base attribute set provides unambiguous identification of a UA – in the context of a telematic service, for instance. Thus, it completely fulfills the requirements of a base attribute set, and in fact goes beyond those requirements.

Initially, two forms of O/R names are included in this Recommendation. Both forms of names are, in fact, O/R addresses. The first form is intended to identify users of the MHS, while the second form is intended primarily to identify users of teletex or other telematic services. It is anticipated that the Recommendation will be extended in the future to allow for other O/R name forms.

### 3.3.2 *O/R names: Form 1*

The first form of MHS O/R name specifies the originator or recipient of a message by means of the country and Administration management domain to which he belongs (or, in the case of a UA belonging to a private MD, the Administration MD with which his PRMD is associated), and a subset of other attributes.

An Administration MD name, a country name, and other attributes form an O/R name. Three variant representations are defined. The first variant consists of a collection of one or more attributes chosen from personal name, organization name, organizational unit names, and private domain names. The second variant consists of an identifier that distinguishes the user's UA from all other UAs served by the same MD. Its purpose is to allow O/R names to be entered from terminals equipped only with numeric keypads. The third variant consists of an X.121 address. Its purpose is to allow telex terminals to be identified in the context of store-and-forward telex by use of the escape digit defined for telex in that Recommendation. Only one of these representations shall be specified for a given recipient by an originator.

*Note* – Attributes enclosed in square brackets are optional

*Variant 1*

O/R name consists of:
Country name
Administration domain name
[Private domain name]
[Personal name]
[Organization Name]
[Organizational Unit Names]
[Domain-defined attributes]

*Note* – At least one of private domain name, personal name, organization name and organizational unit names must be selected.

*Variant 2*

O/R name consists of:
Country name
Administration domain name
UA unique numeric identifier
[Domain-defined attributes]

*Variant 3*

O/R name consists of:
Country name
Administration domain name
X.121 address
[Domain-defined attributes]

User attributes are selected by the MD and convey domain specific information required to identify an individual UA. Standard user attributes are defined in this Recommendation. Domain-defined attributes are left to the discretion of individual MDs.

*Note* – Without additional support, a user whose input device is a numeric keypad will be unable to address messages to other users unless they use the unique UA identifier. Procedures to overcome this limitation are for further study.

This choice of O/R name reflects two pragmatic considerations. First, it recognizes that a directory service spanning all MDs cannot be put in place immediately, and avoids the need for such a service. This is accomplished primarily by the choice of Administration domain name and, where appropriate, private domain name, as part of the base attribute set.

Second, in avoiding the complete standardization of user attributes, this name form recognizes that the initial MHS will be created by linking existing MDs, most of which have already developed user attribute sets to meet their internal naming needs. The initial Recommendation allows the continued use of these domain-defined attributes until MHS-wide standards can be developed.

The reliance on architectural attributes, Administration domain name, and, where appropriate, private domain name, will diminish as directory services are introduced and can provide a more user-friendly form of O/R name.

### 3.3.3 *O/R names: Form 2*

The second form of O/R name specifies the originator or recipient of a message by identifying his telematic terminal.

O/R name consists of: X.121 address,

[Telematic Terminal Identifier]

This form comprises the X.121 address and, optionally, a telematic terminal identifier. The telematic terminal identifier might be, for instance, a telex answer back string or a teletex terminal identifier.

## 3.4 *Routing*

A *route* is information that describes the path to be taken by a message as it progresses through the MTS from originating to recipient UA. O/R addresses provide the MTS with a basis for route selection. Given an O/R address, the MTS can determine a route to the UA denoted by that address.

This Recommendation considers only the routing of messages between the originating and recipient MDs; routing within an MD is beyond the scope of the Recommendation.

The Recommendation initially specifies an incremental routing scheme. That is, each MD along the route of a message determines the MTA in the next MD to which that message should be transferred, but makes no decisions beyond that determination. In particular, no attempt is made to establish the full route for a message, either in the originating MD or in any other MD that acquires store-and-forward responsibility for the message.

The base attribute set in a message's O/R address provides the required input to the incremental routing decisions. An MD charged with routing a message to a recipient examines that recipient's O/R address, extracts the base attribute set, and uses it as a basis for relaying the message. In general, the message will be relayed either directly to an MTA within the recipient MD, or else to an MTA in some intermediate MD that can relay the message toward the recipient MD. In each case, the precise choice of the next MD may depend on policy or on technical, economic, or other concerns, all of which are beyond the scope of this Recommendation.

Generally, the routing of a message will be based solely on the base attribute set in its O/R address and the concerns suggested above. However, the need for special services, such as content conversion, might require some other routing decision. This is for further study.

In the case of an O/R address for a telematic terminal device that is represented by an X.121 address, the message is transferred to the ADMD that serves it. If the telematic terminal device is not served by an ADMD, then it may be delivered directly to the terminal.

An O/R address must be correctly formed – especially with regard to its base attribute set – if the message is to be routed. In particular, an O/R address must contain exactly one base attribute set. O/R addresses with less than a complete base attribute set or with two or more base attribute sets or parts thereof will be rejected.

Messages are relayed by each MD on the basis of the base attribute sets in their O/R addresses until arrival at the recipient MD. At this point, the user attributes in an O/R address are interpreted to support further routing to the recipient UA. The recipient MD has responsibility for checking the correctness of the user attributes. In the normal case, the attributes must identify only one UA, and all of the values for the attributes specified in the address must be identical to the values of the corresponding attributes associated with the recipient UA.

In the case of a message being transferred to a recipient in a private MD, the base attribute set provides for the message to be routed to the specified ADMD to which the PRMD is attached. The ADMD then uses the user attributes in the O/R address to determine the correct PRMD. In some cases, the ADMD may require that the private MD name attribute be included in the user attribute set of any UAs belonging to a PRMD which accesses it. Alternatively the ADMD might require that some other specific user attribute (such as organization name), or set of user attributes (standard or domain-defined) be supplied in order to identify a PRMD UA. The method for designating UAs in an attached PRMD may be individually determined by each ADMD in a country, or may be done uniformly on a national basis. This shall be a national matter within countries where PRMDs exist.

### 3.5 *Directory*

The need of MHS users for a number of directory service elements is recognized. Directory service elements are intended to assist users and their UAs in obtaining information to be used in submitting messages for delivery by the MTS. The MTS may also use directory service elements to obtain information to be used in routing messages. Some functional requirements of directories have been identified and are listed below:

1) Verify the existence of an O/R name.
2) Return the O/R address that corresponds to the O/R name presented.
3) Determine whether the O/R name presented denotes a user or a distribution list.
4) Return a list of the members of a distribution list.
5) When given a partial name, return a list of O/R name possibilities.
6) Allow users to scan directory entries.
7) Allow users to scan directory entries selectively.
8) Return the capabilities of the entity referred to by the O/R name.
9) Provide maintenance functions to keep the directory up-to-date.

In addition to functionality, a number of operational aspects must be considered. These include user-friendliness, flexibility, availability, expandability and reliability.

Currently, these aspects of directory service elements and procedures are for further study.

## 4 Service elements

### 4.1 *Message transfer service*

The basic MT service enables UAs to access and be accessed by the MTS in order to exchange messages. Each message is assigned a unique message reference identification. If a message cannot be delivered, the originating UA is informed. To facilitate meaningful communication, a UA may specify the types of encoded information that can be contained in messages delivered to it. The content type, the original encoded information types, the time of submission and delivery and whether conversion occurred are indicated for each message.

In addition to the basic service, optional service elements may be selected on a per-message basis or for a contractual period of time. These service elements are grouped under the following headings: submission and delivery, conversion, query, and status and inform. The international availability of these service elements is given in Recommendation X.401.

The features of the MT service are listed in Table 1/X.400.

TABLE 1/X.400

**Message transfer service elements**

| Service group | Service elements | Section |
|---|---|---|
| Basic | Access management | 4.1.1.8 |
| | Content type indication | 4.1.1.9 |
| | Converted indication | 4.1.1.5 |
| | Delivery time stamp indication | 4.1.1.7 |
| | Message identification | 4.1.1.1 |
| | Non-delivery notification | 4.1.1.2 |
| | Original encoded information types indication | 4.1.1.4 |
| | Registered encoded information types | 4.1.1.3 |
| | Submission time stamp indication | 4.1.1.6 |
| Submission and delivery | Alternate recipient allowed | 4.1.2.4 |
| | Deferred delivery | 4.1.2.5 |
| | Deferred delivery cancellation | 4.1.2.8 |
| | Delivery notification | 4.1.2.6 |
| | Disclosure of other recipients | 4.1.2.3 |
| | Grade of delivery selection | 4.1.2.1 |
| | Multi-destination delivery | 4.1.2.2 |
| | Prevention of non-delivery notification | 4.1.2.7 |
| | Return of contents | 4.1.2.9 |
| Conversion | Conversion prohibition | 4.1.3.1 |
| | Explicit conversion | 4.1.3.3 |
| | Implicit conversion | 4.1.3.2 |
| Query | Probe | 4.1.4.1 |
| Status and inform | Alternate recipient assignment | 4.1.5.1 |
| | Hold for delivery | 4.1.5.2 |

4.1.1 *Basic message transfer service*

To transfer a message, the originating UA submits the message to the MTS. The name or address of the recipient as specified in § 3 is supplied by the UA along with the message. The MTS completes delivery when the message is given to the recipient's UA along with the originator's name.

The following service elements are part of the basic service:

4.1.1.1 *Message identification*

This service element enables the MTS to provide a UA with a unique identifier for each message submitted to or delivered by the MTS. UAs and the MTS use this identifier to refer to a previously submitted message in connection with service elements such as delivery and non-delivery notification.

4.1.1.2 *Non-delivery notification*

This service element enables the MTS to notify an originating UA if a submitted message was not delivered to the specified recipient UA(s). The reason the message was not delivered is included as part of the notification. For example, the recipient UA may be unknown to the MTS.

In the case of a multi-destination message, a non-delivery notification may refer to any or all of the recipient UAs to which the message could not be delivered.

4.1.1.3 *Registered encoded information types*

This service element enables a UA to inform the MTS of the encoded information type(s) that can be delivered to it.

4.1.1.4 *Original encoded information types indication*

This service element enables an originating UA to specify to the MTS the encoded information types of a message being submitted. When a message is delivered, it also indicates to the recipient UA the encoded information types of the message specified by the originating UA.

4.1.1.5 *Converted indication*

This service element enables the MTS to indicate to a recipient UA that the MTS performed encoded information type conversion on a delivered message. The recipient UA is informed of the resulting types.

4.1.1.6 *Submission time stamp indication*

This service element enables the MTS to indicate to an originating UA and the recipient UA the date and time at which a message was submitted to the MTS.

4.1.1.7 *Delivery time stamp indication*

This service element enables the MTS to indicate to a recipient UA the date and time at which the MTS delivered a message.

4.1.1.8 *Access management*

This service element enables a UA and MTA to establish access to one another and to manage information associated with access establishment.

The service element permits the UA and MTA to identify and validate the identity of the other. It provides a capability for the UA to specify its O/R name and to maintain access security. When access security is achieved through passwords, these passwords can be periodically updated. It should be noted that these passwords are distinct from any local security mechanisms used by the UA to authenticate its user.

The Access Management service element permits the UA to inform the MTS of the network address to be used by the MTS if the UA is accessed through a switched network.

4.1.1.9 *Content type indication*

This service element enables an originating UA to indicate the content type for each submitted message. A recipient UA may have one or more content types delivered to it. An example of a content type is the contents generated by the IPM class of cooperating UAs.

4.1.2 *Submission and delivery service elements*

The following submission and delivery elements may be selected:

4.1.2.1 *Grade of delivery selection*

This service element enables an originating UA to request that transfer through the MTS be *urgent* or *non-urgent*, rather than *normal*. The time periods defined for non-urgent and urgent transfer are longer and shorter, respectively, than that defined for normal transfer. The expected time period for delivery for each of these three grades of delivery is for further study.

4.1.2.2 *Multi-destination delivery*

This service element enables an originating UA to specify that a message being submitted is to be delivered to more than one recipient UA. Simultaneous delivery to all specified UAs is not implied by the service. The number of recipient UAs on a submitted multi-recipient message is unlimited.

4.1.2.3 *Disclosure of other recipients*

This service element enables an originating UA to instruct the MTS, when submitting a multi-recipient message, to disclose the O/R names of all other recipient(s) to each recipient UA upon delivery of the message. The O/R names disclosed are as supplied by the originating UA.

4.1.2.4 *Alternate recipient allowed*

This service element enables an originating UA to specify that the message being submitted may be delivered to an alternate recipient as described below.

A destination MD will interpret all of the user attributes in order to select a recipient UA. Three cases may be distinguished:

- All the attributes match precisely those of a subscriber UA. The message is delivered to that UA.
- Either insufficient attributes are supplied or those supplied match those of more than one subscriber UA. The message cannot be delivered.
- At least the minimum set of attributes required by the destination MD is supplied. Nevertheless, taking all of the attributes into account, the attributes match those of no UA.

In the last case, an MD that supports the alternate recipient assignment service element may deliver the message to a UA that has been assigned to receive such messages. This UA will be notified of the O/R name of the intended recipient as specified by the originator. Delivery to this UA will be reported in a delivery notification if requested by the originator.

*Note 1* – This service element does not allow the delivery of a message if any part of the base attribute set is absent or unrecognized.

*Note 2* – The availability of this service element to the originator does not imply that any alternate recipient UA exists or has been assigned to receive the message.

4.1.2.5 *Deferred delivery*

This service element enables an originating UA to instruct the MTS that a message being submitted should be delivered no sooner than a specified date and time. Delivery will take place as close to the date and time specified as possible, but not before. The date and time specified for deferred delivery is subject to a limit which is defined by the originator's management domain.

4.1.2.6 *Delivery notification*

This service element enables an originating UA to request that an explicit notification be returned to the originating UA when a submitted message has been successfully delivered to a recipient UA. The notification is related to the submitted message by means of the message identifier and includes the date and time of delivery. In the case of a multi-destination message, a delivery notification may refer to any or all of the recipient UAs to which the message was delivered.

Delivery notification carries no implication that any UA or user action, such as examination of the message's content, has taken place.

4.1.2.7 *Prevention of non-delivery notification*

This service element enables an originating UA to instruct the MTS not to return a non-delivery notification to the originating UA in the event that the message being submitted is judged undeliverable.

4.1.2.8 *Deferred delivery cancellation*

This service element enables an originating UA to instruct the MTS to cancel a previously successfully submitted deferred delivery message. The cancellation attempt may not always succeed. Possible reasons for failure are: deferred delivery time has passed, or the message has already been forwarded within the MTS.

4.1.2.9 *Return of contents*

This service element enables an originating UA to request that the content of a submitted message be returned with any non-delivery notification. This will not be done, however, if any encoded information type conversion has been performed on the message's content.

4.1.3 *Conversion service elements*

The following encoded information type conversion elements may be selected.

*Note* – The specific encoded information types and conversions currently defined in the MHS Recommendations are applicable to the IPM content type. Additional information types and conversions may be required in the future to support other content types.

4.1.3.1 *Conversion prohibition*

This service element enables an originating UA to instruct the MTS that encoded information type conversion(s) should not be performed for a particular submitted message.

4.1.3.2 *Implicit conversion*

This service element enables a UA to have the MTS perform for a period of time any necessary conversion on submitted messages. Neither the originating nor recipient UA explicitly requests this service element. If the encoded information type capabilities of the recipient UA are such that more than one type of conversion can be performed, the most appropriate conversion is performed. When a message is delivered after conversion has been performed, the recipient UA is informed of the original encoded information types as well as the current encoded information types in the message.

The possibility of the MTS performing several conversions on the same message is for further study.

The need for the recipient UA to control implicit conversion, and the means by which any MD other than the recipient's can perform implicit conversion, are for further study.

4.1.3.3 *Explicit conversion*

This service element enables an originating UA to request the MTS to perform a specified conversion, such as required when interworking between different telematic services. When a message is delivered after conversion has been performed, the recipient UA is informed of the original encoded information types as well as the current encoded information types in the message.

*Note* – The combination of conversion services is for further study. This service element is intended to support interworking with telematic terminals/services.

4.1.4 *Query service elements*

The query service elements permit UAs to request information related to the control and operation of the MTS.

4.1.4.1 *Probe*

This service element enables a UA to establish before submission whether a particular message could be deliverable. The MTS provides the submission information and generates delivery and/or non-delivery notifications indicating whether a message with the same submission information could be delivered to the specified recipient UAs.

The Probe service element includes the capability of checking whether the message size, content type, and/or encoded information types would render it undeliverable. The significance of the result of a Probe depends upon the recipient UA(s) having registered with the MTS the encoded information types, content type and maximum message size that it can accept. This service element is subject to the same delivery time targets as for the urgent class.

4.1.5 *Status and inform service elements*

4.1.5.1 *Alternate recipient assignment*

This service element enables a UA to be given the capability to have certain messages delivered to it for which there is not an exact match between the recipient attributes specified and the descriptive name(s) of the UA. Such a UA is specified in terms of one or more attributes for which an exact match is required, and one or more attributes for which any value is acceptable. For example, an organization may establish a UA to receive all messages for which country name, Administration management domain name and organization name (for example, company name) are an exact match but the personal name of the recipient does not correspond to an individual known by the MHS in that organization. This permits the organization to manually handle the messages to these individuals.

In order for a message to be assigned to an alternative recipient, the originator must have requested the Alternative Recipient Allowed service element.

Messages will be delivered to the UA if all of the following conditions hold:

1) The originator requested the alternate recipient allowed service element.
2) The recipient O/R name contains a full set of attributes.
3) Those attributes that are required to match do.
4) Those attributes that are not required to match do not match those of any UA in that MD.

#### 4.1.5.2 *Hold for delivery*

This service element enables a recipient UA to request that the MTS hold its messages for delivery and returning notifications until a later time. The UA can indicate to the MTS when it is unavailable to take delivery of messages and notifications, and also, when it is again ready to accept delivery of messages and notifications from the MTS. The MTS may indicate to the UA that messages are waiting due to the criteria the UA established for holding messages. Responsibility for the management of this service element lies with the recipient MTA.

Criteria for requesting a message to be held for delivery are: encoded information type, maximum content length, and priority. The message will be held until the maximum delivery time for that message expires.

*Note 1* – The possible impact of this service element on the expected time period for delivery of a message is for further study.

*Note 2* – The maximum time that a returning notification will be held is for further study.

*Note 3* – It is recognized that stand-alone UAs will generally require the Hold for Delivery service element.

*Note 4* – The Hold for Delivery service element is distinguished from the document storage facility described in Recommendation X.430. The hold for delivery service element provides temporary storage to facilitate delivery and only after a message has been transferred to the recipient's UA, is delivery notification returned. The document storage facility augments the storage of a teletex terminal and may be used to store messages for an extended period of time. Unlike the hold for delivery service element, delivery notifications are returned as soon as the message is placed in (that is, delivered to) the document storage.

### 4.2 *Interpersonal messaging service*

The basic IPM service, which is built upon the MT service, is provided by the class of *cooperating UAs* called IPM UAs, along with the telex and telematic services. This service enables a user to send an IP-message to one or more recipients and to have it received by those recipients. IPM UAs make use of the basic MT service elements of the MTS described in § 4.1.1. They also permit users to request any of the optional MT service elements that are available, as defined in §§ 4.1.2-4.1.5. In addition, IPM UAs provide other capabilities that are basic elements of the IPM Service. For example, they uniquely identify each IP-message and the nature and attributes of its body.

Optional service elements, which may be selected on a per IP-message basis or for a contractual period of time, are also defined. The international availability of these service elements is given in Recommendation X.401. The features of the IPM Service are listed in Table 2/X.400.

An IPM UA may perform many local IPM UA functions for its user in addition to providing access to MT and IPM service elements. Before defining the IPM service elements, some examples of local IPM UA service elements are described.

To assist users in preparing and editing IP-messages, an IPM UA may provide an editing capability. This editor could operate only on a single line of text at a time, or it could permit the display and alteration of a page at a time. A user may have to frequently contact his IPM UA to see if new IP-messages have arrived. Alternatively, the IPM UA could alert him when new IP-messages have been delivered (for example, by setting the message light on his telephone, or by displaying on his desktop terminal the originator's name and subject of all unread IP-messages). An IPM UA may provide local database controls to help the user find previously received and filed IP-messages (for example, to find the IP-message from Mr. Jones, delivered sometime in August on the subject of "teleconferencing"). A user on vacation may request his IPM UA to automatically forward all his IP-messages to his delegate or defines rules for which IP-messages his IPM UA should not forward (for example, personal IP-messages).

TABLE 2/X.400

**Interpersonal messaging service elements**

| Service group | Service elements | Section |
|---|---|---|
| Basic | Basic MT service elements<br>IP-message identification<br>Typed body | 4.1.1<br>4.2.1.2<br>4.2.1.3 |
| Submission and delivery and conversion (MT service elements) | *See Table 1/X.400* | 4.1.2<br>4.1.3 |
| Cooperating IPM UA action | Blind copy recipient indication<br>Non-receipt notification<br>Receipt notification<br>Auto-forwarded indication | 4.2.3.1<br>4.2.3.2<br>4.2.3.3<br>4.2.3.4 |
| Cooperating IPM UA information conveying | Originator indication<br>Authorizing users indication<br>Primary and copy recipients indication<br>Expiry date indication<br>Cross-referencing indication<br>Importance indication<br>Obsoleting indication<br>Sensitivity indication<br>Subject indication<br>Replying IP-message indication<br>Reply request indication<br>Forwarded IP-message indication<br>Body part encryption indication<br>Multi-part body | 4.2.4.1<br>4.2.4.2<br>4.2.4.3<br>4.2.4.4<br>4.2.4.5<br>4.2.4.6<br>4.2.4.7<br>4.2.4.8<br>4.2.4.9<br>4.2.4.10<br>4.2.4.11<br>4.2.4.12<br>4.2.4.13<br>4.2.4.14 |
| Query | *See Table 1/X.400* | 4.1.4 |
| Status and inform (MT service elements) | *See Table 1/X.400* | 4.1.5 |

Local services like those above, while perhaps using some of the MT and IPM service elements, do not require coordination or cooperation with other users and their IPM UAs. Thus, they do not impact the communication protocol standards associated with Message Handling. Therefore, local IPM UA service elements that may be provided by an Administration are not subject to standardization by the CCITT.

### 4.2.1 *Basic interpersonal messaging service*

To send an IP-message, the originating user makes a request of his IPM UA, specifying the O/R name of the recipient that is to receive the IP-message. The originator's IPM UA submits a message containing the IP-message to the MTS for delivery to the specified recipient. Following delivery by the MTS to the recipient's IPM UA, the IP-message can be received by the recipient. The following service elements are part of the basic IPM Service.

#### 4.2.1.1 *Basic message transfer service elements*

The service elements that are part of the basic MT Service and that are used to transfer an IP-message are available as part of the basic IPM Service (see § 4.1.1).

4.2.1.2 *IP-message identification*

This service element permits cooperating IPM UAs to convey an identifier for each IP-message sent or received. The particular method by which this identifier is actually generated is not specified and is a local matter. However, it is important to be aware that IPM UAs and users use this identifier to refer to a previously sent or received IP-message (for example, in receipt notifications).

The IP-message identifier can convey with it the O/R name of the originator (or originating IPM UA) which generated it. It is suggested that the O/R name be included with the identifier under any circumstances where identifiers generated by multiple users are involved, in order to prevent ambiguity.

4.2.1.3 *Typed body*

This service element permits the nature and attributes of the body of the IP-message to be conveyed along with the body. A CCITT-defined, ISO-defined, nationally-defined, or private-use type may be designated. Because the body may undergo conversion, the body type may change over time.

Examples of a typed body include:

1) Unstructured IA No. 5 text.
2) A teletex document.
3) A G3 facsimile page.

4.2.2 *Submission and delivery, and conversion message transfer service elements*

The submission and delivery and conversion service elements of the MT Service may be selected and invoked to transfer an IP-message as options of the IPM Service (see §§ 4.1.2-4.1.3).

4.2.3 *Cooperating IPM UA action service elements*

The following service elements may be selected:

4.2.3.1 *Blind copy recipient indication*

This service element allows the originator to provide the names of one or more additional users who are intended recipients of the IP-message being sent. These names are not disclosed to either the primary or copy recipients. Whether or not these additional recipients are disclosed to one another is a local matter.

It should be noted that the O/R name(s) of the blind copy recipient(s) must be available to the originator's IPM UA to accomplish this service. (See also the Note in § 4.2.4 concerning the use of O/R names and "user-friendly" names in certain IPM service elements.)

4.2.3.2 *Non-receipt notification*

This service element allows the originator to request that he be notified that an IP-message was not received by the intended recipient. For multi-recipient IP-messages, this service element can be specified on a per-recipient basis.

The recipient's IPM UA is expected to return a non-receipt notification in the following circumstances:

a) The IP-message was auto-forwarded to another recipient.
b) The recipient's subscription was terminated before receipt could occur.
c) The IP-message was discarded by the recipient's IPM UA before receipt could take place.

Non-receipt notifications are generated automatically by a recipient's IPM UA – the recipient himself is not involved in generating the notification.

It should be recognized that, in certain cases, a non-receipt notification will not occur even though the IP-message has not been received. In particular, no time limit is used in defining when non-receipt has occurred. Consequently, the failure of the recipient to access his IPM UA and receive his IP-messages over a protracted period of time is not sufficient to cause a non-receipt notification to be generated.

Besides an indication of non-receipt, the notification includes the following:

1) The identifier of the IP-message to which the notification applies.
2) The reason for non-receipt (for example, auto-forwarded, termination of subscription before receipt).
3) The O/R name of the recipient IPM UA generating the notification, and in the event that the IP-message was delivered to an alternate recipient IPM UA, the O/R name of the originally intended recipient as well.
4) An indication of any conversion performed on the IP-message.
5) Optionally, the return of the IP-message, if requested by the originator. However, the IP-message is not returned if conversion has taken place.
6) In the case of auto-forwarding, optionally, a comment supplied by the intended recipient.

#### 4.2.3.3 *Receipt notification*

This service element allows the originator to request that he be notified of the receipt, by a recipient, of an IP-message being sent. The service element also implicitly requests notification of non-receipt as defined in § 4.2.3.2 above.

The recipient's IPM UA can return the receipt notification automatically as, for example, in the case where the IP-message is automatically printed on a terminal. Alternatively, the recipient's IPM UA can require positive action from the recipient to initiate the return of the notification. In this case, the recipient can elect not to have a receipt notification returned, even though the IP-message has been received.

Besides the indication of receipt, the notification includes the following:

1) The identifier of the IP-message to which the notification applies.
2) The time at which receipt occurred.
3) Information describing the manner in which notification was initiated (for example, automatically by the IPM UA or upon explicit action by the recipient).
4) An indication of any conversion performed on the IP-message.
5) The O/R name of the recipient of the IP-message and, in the event that the IP-message was received by an alternate recipient's IPM UA, the O/R name of the originally intended recipient as well.

No legal significance can be attached to a receipt notification.

#### 4.2.3.4 *Auto-forwarded indication*

This service element allows a recipient to determine that a body of an incoming IP-message contains an IP-message that has been auto-forwarded. Thus, the recipient can distinguish from that where an incoming IP-message contains a forwarded IP-message (as described in § 4.2.4.12) in the body. As with a forwarded IP-message, an auto-forwarded IP-message may be accompanied by information (for example, time stamps, indication of conversion) associated with its original delivery.

*Note* – The indication that auto-forwarding of an IP-message has occurred enables a recipient IPM UA, should it so choose, to prevent further auto-forwarding and thus the possibility of loops. In addition, a recipient IPM UA can choose whether or not to auto-forward based on other criteria (for example, sensitivity classification).

When an IPM UA auto-forwards an IP-message, it designates it as auto-forwarded. If receipt notification has been requested for the IP-message being auto-forwarded, the IPM UA generates a non-receipt notification informing the originator of the auto-forwarding of the IP-message. The notification optionally includes a comment supplied by the originally intended recipient. No further notification applying to the auto-forwarded IP-message is generated by any IPM UA.

### 4.2.4 *Cooperating IPM UA information conveying service elements*

The following service elements may be selected.

*Note* – As part of the originator indication, authorizing users indication, primary and copy recipients indication and blind copy recipient indication service elements described in this section and § 4.2.3 above, it is possible for a "user-friendly" form of name (or nickname) to be specified without an O/R name. This should occur only when no requirement is envisaged for recipients of the IP-message to be informed of the O/R name(s) in order to further communicate with the relevant parties by means of the IPM Service. It is always necessary to specify the O/R name of a recipient to whom some specific request for further action (for example, receipt notification or reply) applies.

4.2.4.1 *Originator indication*

This service element allows the identity of the originator to be conveyed to the recipient. The MTS provides to the recipient the authenticated O/R name of the originator. In contrast, the intent of this IPM service element is to identify the originator in a user-friendly way (see the Note in § 4.2.4).

4.2.4.2 *Authorizing users indication*

This service element enables the originator to indicate to the recipient names of the one or more persons who authorized its sending. For example, an individual may authorize a particular action which is subsequently communicated to those concerned by another person such as a secretary. The former person is said to authorize its sending while the latter person is the one who sent the message (originator). This does not imply signature-level authorization.

4.2.4.3 *Primary and copy recipients indication*

This service element allows the originator to provide the names of the one or more users who are the intended primary recipients of the IP-message, and the names of the zero or more users who are the intended copy recipients of the IP-message. It is intended to enable a recipient to determine the category in which each of the specified recipients (including the recipient himself) was placed. The exact distinction between these two categories of recipients is unspecified. However, the primary recipients, for example, might be expected to act upon the IP-message, while the copy recipients might be sent the IP-message for information only.

*Note* – As an example of this service element in a typical memorandum, the primary recipients are normally designated by the directive "to:" while "cc:" identifies the copy recipients.

4.2.4.4 *Expiry date indication*

This service element allows the originator to indicate to the recipient the date and time after which he considers the IP-message to be invalid. The intent of this service element is to state the originator's assessment of the current applicability of an IP-message. The particular action on behalf of a recipient by his IPM UA, or by the recipient himself, is unspecified. Possible actions might be to file or delete the IP-message after the expiry date has passed.

4.2.4.5 *Cross-referencing indication*

This service element allows the originator to associate with the IP-message being sent the identifiers of one or more other IP-messages. This allows the recipient's IPM UA, for example, to retrieve from storage a copy of the referenced IP-messages.

4.2.4.6 *Importance indication*

This service element allows the originator to indicate to the recipients his assessment of the importance of the IP-message being sent. Three levels of importance are defined: low, normal and high.

This service element is not related to the grade of delivery selection service element provided by the MTS. The particular action taken by the recipient or his IPM UA based on the importance categorization is unspecified. It is the intent to allow the recipient IPM UA, for example, to present IP-messages in order of their importance or to alert the recipient of the arrival of IP-messages of high importance.

4.2.4.7 *Obsoleting indication*

This service element allows the originator to indicate that one or more IP-messages he sent previously are obsolete. The IP-message that carries this indication supersedes the obsolete IP-message.

The action to be taken by the recipient or his IPM UA is a local matter. The intent, however, is to allow the IPM UA or the recipient to, for example, remove or file obsolete IP-messages.

4.2.4.8 *Sensitivity indication*

This service element allows the originator of an IP-message to specify guidelines for the relative security of the IP-message upon its receipt. It is the intent that the sensitivity indication should control such items as:

1) Whether the recipient should have to prove his identity to receive the IP-message.
2) Whether the IP-message should be allowed to be printed on a shared printer.
3) Whether an IPM UA should allow the recipient to forward the received IP-message.
4) Whether the IP-message should be allowed to be auto-forwarded.

The sensitivity level can be indicated to the recipient or interpreted directly by the recipient's IPM UA.

If no sensitivity level is indicated, it should be assumed that the IP-message's originator has advised no restriction on the recipient's further disposition of the IP-message. The recipient is free to forward, print, or otherwise do as he chooses with the IP-message.

Three specific levels of sensitivity above the default are defined:

- *Personal:* The IP-message is sent to the recipient as an individual, rather than to him in his role. There is no implication that the IP-message is private, however.
- *Private:* The IP-message contains information that should be seen (or heard) only by the recipient and not by anyone else. The recipient's IPM UA can provide services to enforce this intent on behalf of the IP-message's originator.
- *Company-confidential:* The IP-message contains information that should be handled according to company-specific procedures.

*Note* – Other levels of sensitivity indication are for further study.

#### 4.2.4.9 *Subject indication*

This service element enables the originator to indicate to the recipient(s) the subject of the IP-message being sent. The subject information is to be made available to the recipient.

#### 4.2.4.10 *Replying IP-message indication*

This service element allows the originator of an IP-message to indicate to the recipient(s) that this IP-message is being sent in reply to another IP-message. If, by means of the Reply Request Indication service element, the originator of the original message specified the intended recipients of the reply, the originator of the reply should address it to those users. Otherwise the reply should be sent only to the originator. The originator of the reply may also specify the O/R names of additional users who are to receive copies of the reply for information.

The recipients of the reply receive it as a regular IP-message, together with an indication of to which IP-message it is a reply.

#### 4.2.4.11 *Reply request indication*

This service element enables the originator to request that a recipient send an IP-message in reply to the IP-message that carries the request. The originator can also specify the date by which the reply should be sent, and the O/R names of the one or more users to whom the reply should be sent. The recipient is informed of the date and names, but it is up to the recipient to decide whether or not to reply. A blind copy recipient should consider carefully to whom he sends a reply, in order that the meaning of the blind copy designation service element is preserved.

#### 4.2.4.12 *Forwarded IP-message indication*

This service element allows a forwarded IP-message, or a forwarded IP-message plus its "delivery information", to be sent as the body (or as one of the body parts) of an IP-message, as shown in Figure 11/X.400. An indication that the body part is forwarded is conveyed along with the body part. In a multi-part body, forwarded body parts can be included along with body parts of other types. "Delivery information" is information which is conveyed from the MTS when an IP-message is delivered (for example, time stamps and indication of conversion). However, inclusion of this delivery information along with a forwarded IP-message in no way guarantees that this delivery information is validated by the MTS.

The receipt and non-receipt notification service elements are not affected by the forwarding of an IP-message.

#### 4.2.4.13 *Body part encryption indication*

This service element allows the originator to indicate to the recipient that any body part of the IP-message being sent that has been encrypted. Encryption can be used to prevent unauthorized inspection or modification of the body part. This service element can be used by the recipient to determine that some body part(s) of the IP-message must be decrypted. However, this service element does not itself encrypt or decrypt any body part.

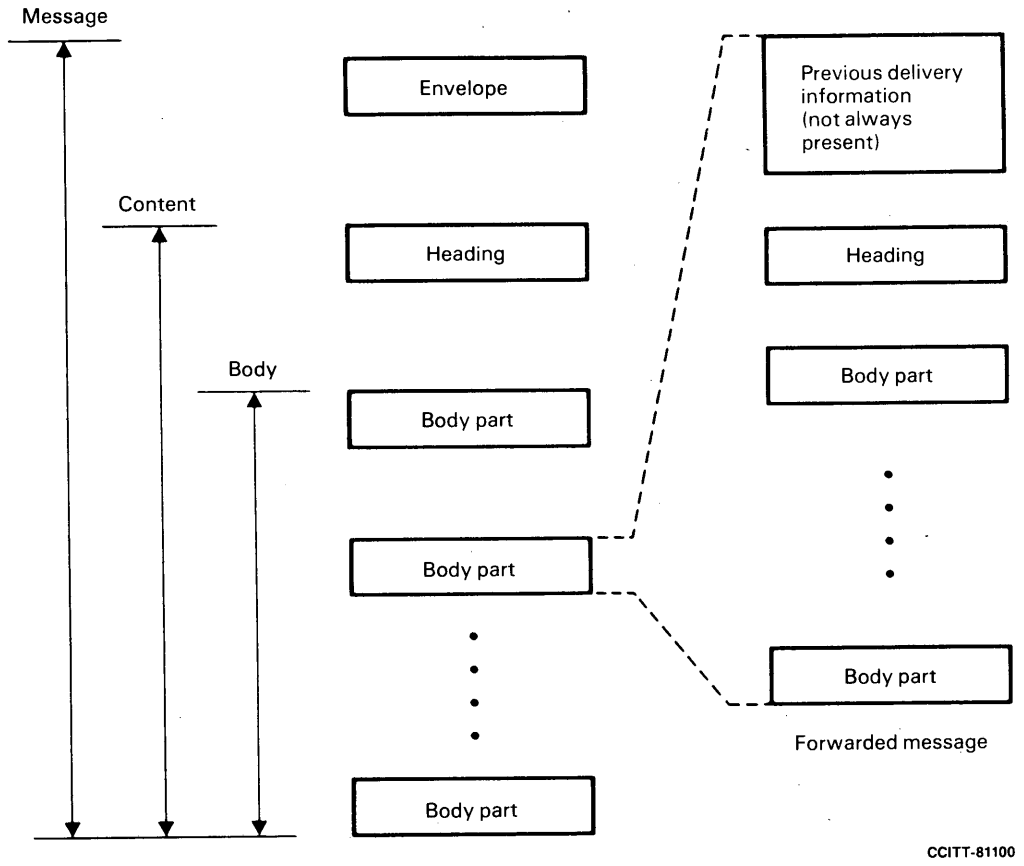



FIGURE 11/X.400

**Forwarded IP-message**

4.2.4.14 *Multi-part body*

This service element allows an originator to send to a recipient or recipients an IP-message with a body that is partitioned into several parts. The nature and attributes, or type, of each body part is conveyed along with the body part.

4.2.5 *Query message transfer service elements*

These service elements enable a user to instruct its IPM UA to invoke the query service elements of the MT Service (see § 4.1.4).

4.2.6 *Status and inform message transfer service elements*

These service elements enable a user to instruct his IPM UA to invoke the status and inform service elements of the MT Service (see § 4.1.5).

4.3 *Directory service*

This topic is for further study.

## 5 Layered representation of the Message Handling System (MHS) model

5.1 *Principles and concepts*

In this section, a layered representation of the MHS model is described. The representation is entirely compatible with the functional model described in § 2, and serves the following purposes:

- it allows principles and techniques developed for the OSI Reference Model to be applied to MHS;
- it identifies the protocols, entities, and service interfaces required for MHS;
- it allows the MHS protocols to be related to specific layers of the OSI Reference Model.

In developing the layered model for MHS, the principles of layering developed for the OSI Reference Model were used. Briefly, these are as follows:

1) Layers are defined so as to group similar functions together.

2) Layers are defined so as to minimize interactions across layer boundaries.

3) Layers are defined so that different protocols may be used within a layer without affecting the layer service definition.

4) The number of layers is kept to the minimum consistent with the above principles.

5) Each layer adds to the services provided by the layer below.

6) Each layer requires one or more distinct peer protocols.

## 5.2 *Layered description of the message handling system*

The message handling entities and protocols are located in the application layer of the OSI Reference Model. This is done to permit the MH applications to use the underlying layers to accomplish the following:

1) establish connections between individual systems using a variety of network types (for example, circuit, packet, telephone, and local area);

2) establish session-connections to permit the message handling applications to reliably transfer messages between open systems;

3) signal the use of the standardized message handling presentation transfer syntax defined in Recommendation X.409.

The message handling functions in the application layer can be considered to be divided into two layers:

1) The user agent layer (UAL) contains the UA functionality associated with the contents of messages.

2) The message transfer layer (MTL) contains the MTA functionality and provides the MT Service defined in § 4.1.

The layered description of MHS can be related to the functional model of MHS which is described in § 2 of this Recommendation. Three types of systems can be distinguished, based on the functional model, as follows:

1) Systems that contain only UA functions (*S1*),

2) Systems that contain only MTA functions (*S2*),

3) Systems that contain both UA and MTA functions (*S3*).

Applying the layered description, these systems and the protocols that operate between them are illustrated in Figure 12/X.400.

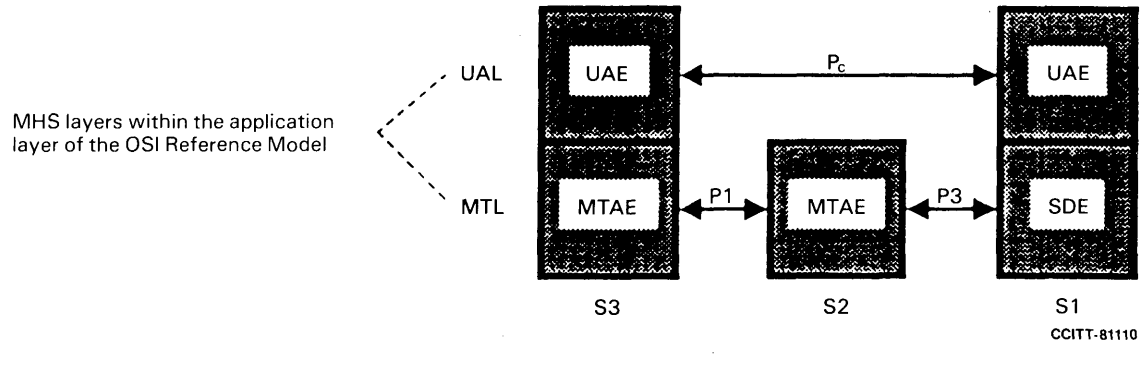



FIGURE 12/X.400

**Layered description of the message handling system**

Three functional entities appear in the layered model:

1) The UA entity (UAE) embodies those aspects of UA functionality having to do with representation of the message's content and other cooperating UA layer functionality as required. An originator/recipient above the UAL can make use of the capabilities provided by the cooperation of the UAEs and the MTL below. Local UA functionality is beyond the scope of this model.

   *Note* – Two similar terms "UA" and "UAE" are used throughout the description of message handling systems. The term UA refers to the functionality described in the functional model in § 2 [for example, as represented by (S1) Systems]. The term "UAE" is used to denote strictly the UA functionality which is associated with the operation of some UA to UA protocol as identified in the layered model ($P_c$ as defined below).

2) The MTA entity (MTAE) provides the functionality required to support the layer services of the MTL in cooperation with other MTAEs.

3) The submission and delivery entity (SDE) makes the services of the MTL available to a UAE through the MTL boundary. The SDE does not itself provide message transfer services, but rather interacts with the peer MTAE to provide access to the MTAE.

Three peer protocols appear in the layered model:

1) The message transfer protocol (P1) defines the relaying of messages between MTAs and other interactions necessary to provide MTL services. While being transferred by means of P1, a message comprises a message content, as submitted by the UAE, and a relaying envelope. The envelope contains the information the MTAEs require to provide the services of the MTL. This protocol is specified in Recommendation X.411. The use of the OSI layers below the application layer to reliably relay messages between MTAs is specified in Recommendation X.410.

2) The submission and delivery protocol (P3) allows the SDE in an S1 system to provide its UAE with access to MTL services. This protocol is specified in Recommendation X.411. The use of the OSI layers below the application layer to reliably exchange messages between SDEs and MTAEs and the remote operation procedures for coordinating their interactions are specified in Recommendation X.410.

3) $P_c$ is potentially a range of protocols defining the syntax and semantics of the message content being transferred. Each instance of a $P_c$ protocol is associated with a class of cooperating UA.

## 5.3 *Layered description of the interpersonal messaging system*

The IPMS provides originators and recipients with the IPM service described in § 4.2. As shown in Figure 13/X.400, the class of of UAs that act on behalf of users in the IPM community cooperate with one another using a specific instance of $P_c$ called the Interpersonal Messaging Protocol (P2) which is specified in Recommendation X.420.

This protocol essentially consists of:

a) The definition of a set of protocol elements, or components, each having a standardized syntax and semantics. These elements are used to form the contents of the messages exchanged between IPM UAEs.

b) The operations relating to the exchange of these protocol elements which an IPM UAE must perform.

c) The rules which an IPM UAE must follow in utilizing the MTL service in providing the IPM service.

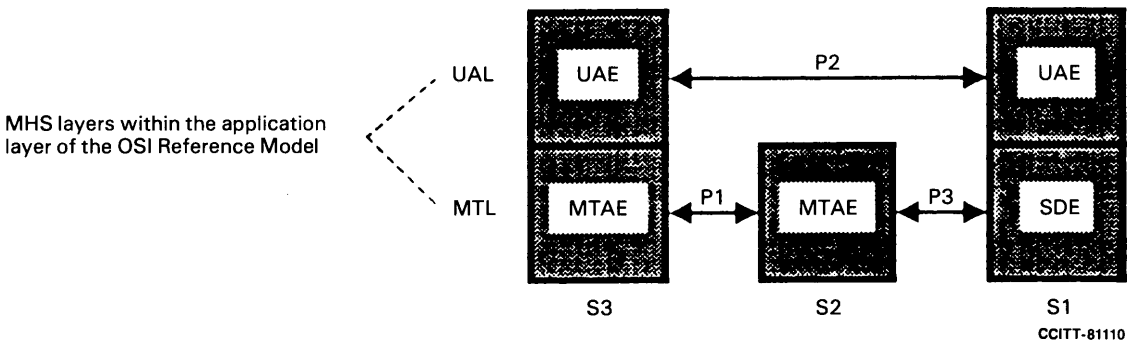



FIGURE 13/X.400

**Layered description of the interpersonal messaging system**

### 5.3.1 *User interaction with the UAL and local UA functions*

From above the IPM UAL, a UAE is driven by the actions of the originator/recipient who uses it. In many cases, there will be locally provided UA functions which assist the user in accessing the IPM services and possibly provide additional features (see Figure 14/X.400). These functions are not the subject of the MHS Recommendations, however, and are not described by this layered model. The way in which the IPM service elements are provided is described as if the originator/recipient always directly interacted with his UAE.

To permit remote users to interact with their UAs, the Interactive Terminal to System Protocols ($P_t$) are identified. $P_t$ is a family of interactive protocols (for example, based on Recommendation X.29). The selection of $P_t$ is not part of this Recommendation and is a national matter.

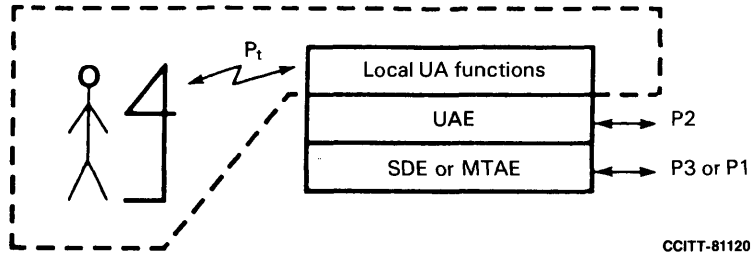



FIGURE 14/X.400

**Representation of local UA functions**

### 5.3.2 *Interpersonal messaging content*

The high level structure of the message content exchanged by IPM UAEs is illustrated in Figure 15/X.400.

Two types of IPM contents are defined: IP-message and IPM-status-report. An *IP-message* carries information generated by and transferred between users. The *heading* of the IP-message carries control information that adds further meaning to the IP-message or helps support cooperating IPM UA service elements. For example, the heading contains fields such as "subject:", "to:", and "cc:" that are associated with memoranda, as defined in § 4.2. The *body* of the IP-message is the information the originating user wishes to communicate, for example, paragraphs of text, voice comments, facsimile diagrams, and combinations of the above.

An *IPM-status-report* content consists of information generated by an IPM UAE (e.g., receipt notification), and is transferred to another IPM UAE, either to be used by that UAE or to be conveyed to a user.

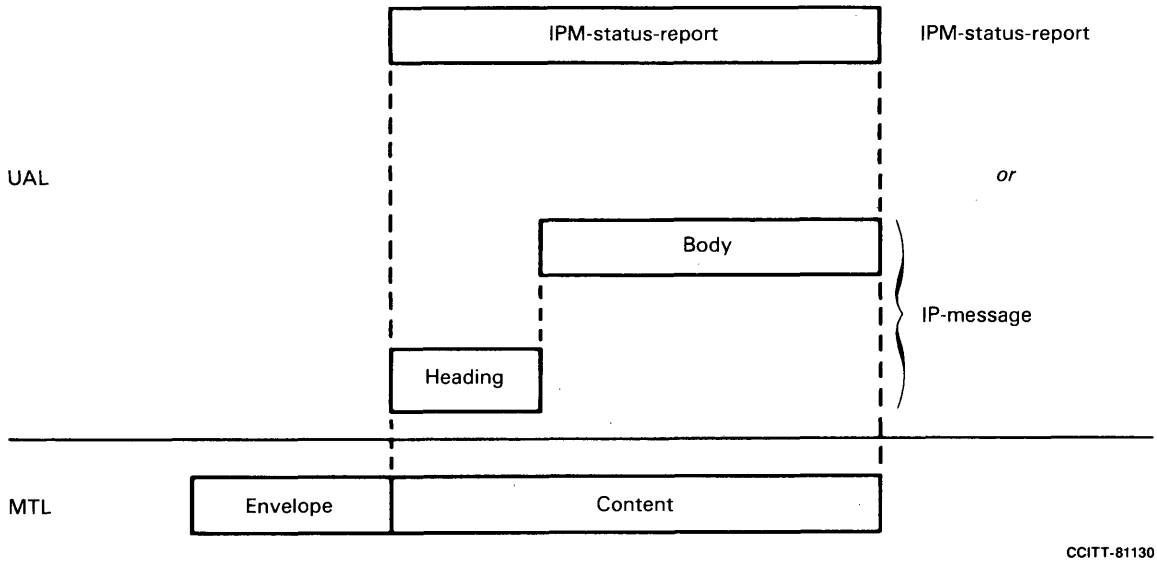



FIGURE 15/X.400

**Interpersonal messaging contents structure**

ANNEX A

(to Recommendation X.400)

## Glossary of terms

The following terms are implicitly defined in the body of this Recommendation. They are listed in this Annex in an explanatory way which does not establish definitions in the strict sense.

**Administration management domain (ADMD)**

A management domain managed by an Administration.

**base attribute set**

A minimum set of attributes whose values unambiguously identify a particular management domain.

**body**

The body of the IP-message is the information the user wishes to communicate.

**content**

The piece of information that the originating UA wishes delivered to the recipient UA. For IPM UAs, the content consists of either an IP-message or an IPM-status-report.

**cooperating user agent**

A UA that cooperates with another recipient's UA in order to facilitate the communication between the originator and recipient.

**delivery**

The interaction by which the Message Transfer Agent transfers to a recipient User Agent the content of a message plus the delivery envelope.

**delivery envelope**

The envelope which contains the information related to the delivery of the message.

**descriptive name**

A name that denotes exactly one user in the MHS.

**encoded information type**

It is the code and format of information that appears in the body of an IP-message. Examples of encoded information types are Teletex, TIFO (Group 4 facsimile) and voice.

**envelope**

A place in which the information to be used in the submission, delivery and relaying of a message is contained.

**heading**

The Heading of an IP-message is the control information that characterizes the IP-message.

**interpersonal messaging (IPM)**

Communication between persons by exchanging messages.

**interpersonal messaging service**

The set of service elements which enable users to exchange interpersonal messages.

**interpersonal messaging system (IPMS)**

The collection of UAs and MTAs, which provide the Interpersonal Messaging Service.

**IP-message**

An IP-message carries information generated by and transferred between IPM UAs. An IP-message contains a Heading and a Body.

**IPM-status-report**

The pieces of information generated by an IPM UAE and transferred to another IPM UAE, either to be used by that UAE or to be conveyed to a user.

**management domain (MD)**

The set of MHS entities managed by an Administration or organization that includes at least one MTA.

**message**

In the context of Message Handling Systems, the unit of information transferred by the MTS. It consists of an envelope and a content.

**message handling address**

An O/R address which is comprised of an Administration Management Domain name, a country name, and a set of user attributes.

**message handling system (MHS)**

The set of UAs plus the MTS.

**message transfer agent (MTA)**

The functional component that, together with other MTAs, constitutes the MTS. The MTAs provide message transfer service elements by: (1) interacting with originating UAs via the submission dialogue, (2) relaying messages to other MTAs based upon recipient designations, and (3) interacting with recipient UAs via the delivery dialogue.

**message transfer agent entity (MTAE)**

An entity, located in an MTA, that is responsible for controlling the MTL. It controls the operation of the protocol to other peer entities in the MTL.

**message transfer layer (MTL)**

A layer in the Application Layer that provides MTS service elements. These services are provided by means of the services of the layer below plus the functionality of the entities in the layer, namely, the MTAEs and the SDEs.

**message transfer protocol (P1)**

The protocol which defines the relaying of messages between MTAs and other interactions necessary to provide MTL services.

**message transfer service**

The set of optional service elements provided by the Message Transfer System.

**message transfer system (MTS)**

The collection of MTAs, which provide the Message Transfer Service elements.

**open systems interconnection (OSI)**

A term referring to the capability of interconnecting different systems.

**O/R address**

A descriptive name for a UA that contains certain characteristics which help the MTS to locate the UA's point of attachment. An O/R address is a type of O/R name.

**O/R name**

The descriptive name for a User Agent.

**originating UA**

A UA that submits to the MTS a message to be transferred.

**originator**

A user, a human being or computer process, from whom the MHS accepts a message.

**primitive name**

A name assigned by a naming authority. Primitive names are components of descriptive names.

**private management domain (PRMD)**

A management domain managed by a company or non-commercial organization.

**recipient**

A user, a human being or computer process, who receives a message from the MHS.

**recipient UA**

A UA to which a message is delivered or that is specified for delivery.

**relaying**

The interaction by which one Message Transfer Agent transfers to another the content of a message plus the relaying envelope.

**relaying envelope**

The envelope which contains the information related to the operation of the MTS plus the service elements requested by the originating UA.

**submission**

The interaction by which an originating User Agent transfers to a Message Transfer Agent the contents of a message plus the submission envelope.

**submission and delivery entity (SDE)**

An entity that is located in the MTL, co-resident with a UA and not with an MTA, and responsible for controlling the submission and delivery interactions with an MTAE.

**submission and delivery protocol (P3)**

The protocol which governs communication between an SDE and an MTAE.

**submission envelope**

The envelope which contains the information the MTS requires to provide the requested service elements.

**user**

A person or computer application or process who make use of MHS.

**user agent (UA)**

Typically, a set of computer processes (for example, an editor, a file system, a word processor) that are used to create, inspect, and manage the storage of messages. There is typically one user per UA. During message preparation, the originator communicates with his UA via an input/output (I/O) device (for example, a keyboard, display, printer, facsimile machine, and/or telephone). Also by means of these devices, the UA communicates to its user messages received from the MTS. To send and receive messages, the UA interacts with the MTS via the submission and delivery protocol.

**user agent entity (UAE)**

An entity in the UAL of the Application Layer that controls the protocol associated with cooperating UAL services. It exchanges control information with the MTAE or SDE in the layer below. The control information is the information the MTL requires to create the appropriate envelope and thus provide the desired message transfer service elements.

**user agent layer (UAL)**

The layer that contains the UAEs.

## ANNEX B

(to Recommendation X.400)

### Abbreviations

The following abbreviations are used in this Recommendation.

| | |
|---|---|
| ADMD | Administration management domain |
| cc | Courtesy copy |
| ID | Identifier/Identification |
| I/O | Input/Output |
| IPM | Interpersonal messaging |
| IPMS | Interpersonal messaging system |
| MD | Management domain |
| MH | Message handling |
| MHS | Message handling system |
| MT | Message transfer |
| MTA | Message transfer agent |
| MTAE | Message transfer agent entity |
| MTL | Message transfer layer |
| MTS | Message transfer system |
| O/R | Originator/Recipient |
| OSI | Open systems interconnection |
| P1 | Message transfer protocol |
| P2 | Interpersonal messaging protocol |
| P3 | Submission and delivery protocol |
| $P_c$ | Family of content protocols |
| PRMD | Private management domain |
| $P_t$ | Family of interactive terminal to system protocols |
| SDE | Submission and delivery entity |
| TIF0 | Text interchange format 0 |
| UA | User agent |
| UAE | User agent entity |
| UAL | User agent layer |

APPENDIX I

(to Recommendation X.400)

**List of terms**

The following terms are contained in this Recommendation. References to definitions, implicit or explicit, are included where appropriate and available.

## A

**Access Management**
cont. in X.400; impl. def. in X.400

**Address**
cont. in X.400; impl. def. in X.400

**Administration**
cont. in X.400

**Administration Management Domain**
cont. in X.400; impl. def. in X.400

**Alternate Recipient Allowed**
cont. in X.400; impl. def. in X.400

**Alternate Recipient Assignment**
cont. in X.400; impl. def. in X.400

**Application Layer**
cont. in X.400; impl. def. in X.200

**Architectural Attributes**
cont. in X.400; impl. def. in X.400

**Attributes**
cont. in X.400; impl. def. in X.400

**Attribute Lists**
cont. in X.400; impl. def. in X.400

**Authorizing Users Indication**
cont. in X.400; impl. def. in X.400

**Auto-forwarded Indication**
cont. in X.400; impl. def. in X.400

## B

**Base Attribute Set**
cont. in X.400; impl. def. in X.400

**Blind Copy Recipient Indication**
cont. in X.400; impl. def. in X.400

**Body**
cont. in X.400; impl. def. in X.400

**Body Part**
cont. in X.400; impl. def. in X.400

**Body Part Encryption Indication**
cont. in X.400; impl. def. in X.400

## C

**CCITT-defined Attribute**
cont. in X.400; impl. def. in X.400

**Content**
cont. in X.400; impl. def. in X.400

**Content Type**
cont. in X.400; impl. def. in X.400

**Content Type Indication**
cont. in X.400; impl. def. in X.400

**Conversion Prohibition**
cont. in X.400; impl. def. in X.400

**Converted Indication**
cont. in X.400; impl. def. in X.400

**Cooperating User Agent**
cont. in X.400; impl. def. in X.400

**Courtesy Copy**
cont. in X.400; impl. def. in X.400

**Cross-referencing Indication**
cont. in X.400; impl. def. in X.400

## D

**Deferred Delivery**
cont. in X.400; impl. def. in X.400

**Deferred Delivery Cancellation**
cont. in X.400; impl. def. in X.400

**Delivery**
cont. in X.400; impl. def. in X.400

**Delivery Dialogue**
cont. in X.400; impl. def. in X.400

**Delivery Envelope**
cont. in X.400; impl. def. in X.400

**Delivery Notification**
cont. in X.400; impl. def. in X.400

**Delivery Time Stamp Indication**
cont. in X.400; impl. def. in X.400

**Descriptive Name**
cont. in X.400; impl. def. in X.400

**Directory**
cont. in X.400

**Directory Function**
cont. in X.400

**Directory Service**
cont. in X.400

**Disclosure of Other Recipients**
cont. in X.400; impl. def. in X.400

**Domain-defined Attribute**
cont. in X.400; impl. def. in X.400

## E

**Encoded Information Type**
cont. in X.400; impl. def. in X.400

**Envelope**
cont. in X.400; impl. def. in X.400

**Expiry Date Indication**
cont. in X.400; impl. def. in X.400

**Explicit Conversion**
cont. in X.400; impl. def. in X.400

## F

**Forwarded IP-message Indication**
cont. in X.400; impl. def. in X.400

## G

**Geographical Attribute**
cont. in X.400; impl. def. in X.400

**Grade of Delivery Selection**
cont. in X.400; impl. def. in X.400

## H

**Heading**
cont. in X.400; impl. def. in X.400

**Hold for Delivery**
cont. in X.400; impl. def. in X.400

## I

**Implicit Conversion**
cont. in X.400; impl. def. in X.400

**Importance Indication**
cont. in X.400; impl. def. in X.400

**Interactive Terminal to System Protocol**
cont. in X.400; impl. def. in X.400

**Interpersonal Messaging**
cont. in X.400; impl. def. in X.400

**Interpersonal Messaging Content**
cont. in X.400; impl. def. in X.400

**Interpersonal Messaging Protocol**
cont. in X.400

**Interpersonal Messaging Service**
cont. in X.400; impl. def. in X.400

**Interpersonal Messaging System**
cont. in X.400

**IP-message**
cont. in X.400; impl. def. in X.420

**IP-message Identification**
cont. in X.400; impl. def. in X.400

**IPM Content Type**
cont. in X.400; impl. def. in X.400

**IPM-status-report**
cont. in X.400

## L

**Layer**
cont. in X.400; expl. def. in X.200

## M

**Management Domain**
cont. in X.400; impl. def. in X.400

**Message**
cont. in X.400; impl. def. in X.400

**Message Handling**
cont. in X.400; impl. def. in X.400

**Message Handling Address**
cont. in X.400; impl. def. in X.400

**Message Handling Environment**
cont. in X.400; impl. def. in X.400

**Message Handling System**
cont. in X.400; impl. def. in X.400

**Message Handling System Model**
cont. in X.400; impl. def. in X.400

**Message Identification**
cont. in X.400; impl. def. in X.400

**Message Transfer Agent**
cont. in X.400; impl. def. in X.400

**Message Transfer Agent Entity**
cont. in X.400; impl. def. in X.400

**Message Transfer Layer**
cont. in X.400; impl. def. in X.400

**Message Transfer Protocol**
cont. in X.400

**Message Transfer Service**
cont. in X.400; impl. def. in X.400

**Message Transfer System**
cont. in X.400; impl. def. in X.400

**MTL Envelope**
cont. in X.400; impl. def. in X.400

**Multi-destination Delivery**
cont. in X.400; impl. def. in X.400

**Multi-part Body**
cont. in X.400; impl. def. in X.400

## N

**Naming and Addressing**
cont. in X.400; impl. def. in X.400

**Non-delivery Notification**
cont. in X.400; impl. def. in X.400

**Non-receipt Notification**
cont. in X.400; impl. def. in X.400

## O

**O/R Address**
cont. in X.400; impl. def. in X.400

**O/R Name**
cont. in X.400; impl. def. in X.400

**Obsoleting Indication**
cont. in X.400; impl. def. in X.400

**Open Systems Interconnection**
cont. in X.400

**Optional User Facility**
cont. in X.400; cont in X.2; expl. def. in X.15

**Organizational Mapping**
cont. in X.400; impl. def. in X.400

**Original Encoded Information Types Indication**
cont. in X.400; impl. def. in X.400

**Originating User Agent**
cont. in X.400; impl. def. in X.400

**Originator**
cont. in X.400; impl. def. in X.400

**Originator Indication**
cont. in X.400; impl. def. in X.400

## P

**Peer Entity**
cont. in X.400; impl. def. in X.400

**Personal Attribute**
cont. in X.400; impl. def. in X.400

**Physical Mapping**
cont. in X.400; impl. def. in X.400

**Prevention of Non-delivery Notification**
cont. in X.400; impl. def. in X.400

**Primary and Copy Recipients Indication**
cont. in X.400; impl. def. in X.400

**Primitive Name**
cont. in X.400; impl. def. in X.400

**Private Management Domain**
cont. in X.400; impl. def. in X.400

**Probe**
cont. in X.400; impl. def. in X.400

**Protocol**
cont. in X.400

## R

**Receipt Notification**
cont. in X.400; impl. def. in X.400

**Recipient**
cont. in X.400; impl. def. in X.400

**Recipient User Agent**
cont. in X.400; impl. def. in X.400

**Registered Encoded Information Types**
cont. in X.400; impl. def. in X.400

**Relaying**
cont. in X.400

**Relaying Envelope**
cont. in X.400; impl. def. in X.400

**Reply Request Indication**
cont. in X.400; impl. def. in X.400

**Replying IP-message Indication**
cont. in X.400; impl. def. in X.400

**Return of Contents**
cont. in X.400; impl. def. in X.400

## S

**Sensitivity Indication**
cont. in X.400; impl. def. in X.400

**Service Element**
cont. in X.400; impl. def. in X.400

**Subject Indication**
cont. in X.400; impl. def. in X.400

**Submission**
cont. in X.400; impl. def. in X.400

**Submission and Delivery Entity**
cont. in X.400; impl. def. in X.400

**Submisssion and Delivery Protocol**
cont. in X.400

**Submission Envelope**
cont. in X.400; impl. def. in X.400

**Submission Time Stamp Indication**
cont. in X.400; impl. def. in X.400

## T

**Telematic**
cont. in X.400

**Telematic Service**
cont. in X.400

**Teletex Terminal**
cont. in X.400

**Teletex Service**
cont. in X.400; impl. def. in X.400

**Teletex Terminal Identifier**
cont. in X.400

**Telex Service**
cont. in X.400

**Typed Body**
cont. in X.400; impl. def. in X.400

## U

**User**
cont. in X.400; impl. def. in X.400

**User Agent**
cont. in X.400; impl. def. in X.400

**User Agent Entity**
cont. in X.400; impl. def. in X.400

**User Agent Layer**
cont. in X.400; impl. def. in X.400

# MESSAGE HANDLING SYSTEMS BASIC SERVICE ELEMENTS AND OPTIONAL USER FACILITIES

*(Malaga-Torremolinos, 1984)*

The establishment in various countries of telematic services and computer-based store-and-forward message services in association with public data networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT,

*considering*

(a) the need for interpersonal messaging and message transfer services;

(b) the need to transfer messages of different types having a large variety of formats;

(c) that Recommendation X.1 specifies user classes of service and Recommendation X.10 categories of access; Recommendation X.2 defines optional user facilities for public data networks;

(d) that Recommendations X.20, X.20 *bis*, X.21, X.21 *bis*, X.22, X.25, X.28 and X.29 define DTE/DCE interfaces to public data networks;

(e) that Recommendations X.300 and X.121 permit international connections between public data networks;

(f) that the F-series of Recommendations defines telematic services, that the T-series of Recommendations defines terminal equipment and control procedures for telematic services;

(g) that the V-series of Recommendations provides the means for data communication over the telephone network;

(h) that Recommendation X.200 defines the reference model of open systems interconnection for CCITT applications;

(i) that a set of Recommendations describes various aspects of message handling: X.400, X.401, X.408, X.409, X.410, X.411, X.420 and X.430,

*unanimously declares*

that the basic service elements and optional user facilities described in this Recommendation are for use in message handling systems.

## 1 Purpose and scope

This Recommendation is one of a series of Recommendations and describes the basic service elements and the optional user facilities of the Message Handling System (MHS). MHS service elements are provided by means of the Interpersonal Messaging (IPM) and Message Transfer (MT) Services. This Recommendation overviews those services and categorizes the optional user facilities of each. Locally provided service elements, for which communication with other users is not required, are not covered by this Recommendation.

Other Recommendations in the series contain additional information essential to a thorough understanding of the IPM and MT Services. Recommendation X.400 describes the MHS model and defines the IPM and MT service elements. Recommendation X.408 specifies the algorithms that MHS uses when converting between different types of encoded information. Recommendation X.409 defines the notation and the representational technique used to specify and encode MHS protocols. Recommendation X.410 describes general techniques used for MHS protocols and the way in which Standard Open Systems Interconnection (OSI) protocols are used to support MHS applications. Recommendation X.411 specifies the protocol aspects of the MT Service. Recommendation X.420 specifies the protocol aspects of the IPM Service. Recommendation X.430 describes how teletex terminals access the MHS.

Some service elements are inherent in the MHS; these constitute the *basic* IPM and MT services. Additional service elements, called *optional user facilities*, may be selected by the subscriber or user, either on a per-message basis or for an agreed contractual period of time. Each optional user facility visible to the user is classified as either essential or additional. *Essential (E)* optional user facilities are to be made available internationally by Administrations. *Additional (A)* optional user facilities may be made available by some Administrations for national use and may also be available internationally on the basis of bilateral agreement.

The MHS is designed in accordance with the principles of the Reference model of open systems interconnection for CCITT applications (Recommendation X.200), and thus can be constructed using any physical network. The IPM and/or MT Services may be offered separately or in combination with various telematic services (Recommendations F.160, F.200, F.300) or data communication services (Recommendations X.1, X.2, X.10 + V Series). They can be obtained by making appropriate arrangements.

## 2 Interpersonal messaging service

### 2.1 *Basic service*

A set of service elements comprises the basic service. The basic IPM service enables a user to send and receive IP-messages. A user prepares IP-messages with the assistance of his User Agent (UA). User Agents, which are a set of computer application processes, cooperate with each other to facilitate communication between their respective users. To send an IP-message, the originating user makes a request of his UA, specifying the name or address of the recipient who is to receive the IP-message. The IP-message, which has an identifier conveyed with it, is then sent by the originator's UA to the recipient's UA via the message transfer service (see § 3). Following a successful delivery to the recipient's UA, the IP-message can be received by the recipient. To facilitate meaningful communication, a user may specify the encoded information type(s) that can be contained in IP-messages delivered to him. The original encoded information type(s) and an indication of any conversions that may have been performed and the resulting encoded information type(s) are supplied with each delivered IP-message. In addition, the submission time and delivery time are supplied with each IP-message. Non-delivery notification is provided with the basic service. Other capabilities are included.

Users of the IPM service are able to exchange IP-messages with subscribers of the teletex service and eventually with subscribers to telex and other telematic services.

An IPM UA may perform many local UA functions for its user in addition to providing access to MT and IPM service elements. For example, to assist users in preparing and editing IP-messages, a UA may provide an editing capability. This editor could operate only on a single line of text at a time, or it could permit the display and alteration of a page at a time. A user may have to frequently contact his UA to see if new messages have arrived. Alternatively, the UA could alert him when new IP-messages have been delivered (for example, by setting the message light on his telephone, or by displaying on his desktop terminal the originator's name and subject of all unread messages). A UA may provide local database controls to help the user find previously received and filed IP-messages (for example, to find the message from Mr. Jones, delivered sometime in August on the subject of "teleconferencing"). A user on vacation may request his UA to automatically forward all his IP-messages to his delegate or define rules for which IP-messages his UA should not forward (for example, personal messages).

Local services like those above, while perhaps using some of the MT and IPM service elements, do not require coordination or cooperation with other users and their UAs. Thus they do not impact the communication protocol standards associated with Message Handling. Therefore, local UA service elements that may be provided by an Administration are not subject to standardization by the CCITT.

The access to User Agents provided by an Administration is outside the scope of this Recommendation.

### 2.2 *IPM optional user facilities*

A set of the service elements of the IPM service are optional user facilities. The optional user facilities of the IPM service, which may be selected on a per-message basis or for an agreed contractual period of time, are listed in Tables 1/X.401 and 2/X.401, respectively. Local user facilities may be usefully provided in conjunction with some of these optional user facilities.

TABLE 1/X.401

**IPM optional user facilities selectable on a per-message basis**

| IPM optional user facilities | Origination by UAs | Reception by UAs |
|---|---|---|
| Alternate recipient allowed | A | A |
| Authorizing users indication | A | E |
| Auto-forwarded indication | A | E |
| Blind copy recipient indication | A | E |
| Body part encryption indication | A | E |
| Conversion prohibition | E | E |
| Cross-referencing indication | A | E |
| Deferred delivery | E | N/A |
| Deferred delivery cancellation | A | N/A |
| Delivery notification | E | N/A |
| Disclosure of other recipients | A | E |
| Expiry date indication | A | E |
| Explicit conversion | A | N/A |
| Forwarded IP-message indication | A | E |
| Grade of delivery selection | E | E |
| Importance indication | A | E |
| Multi-destination delivery | E | N/A |
| Multi-part body | A | E |
| Non-receipt notification | A | A |
| Obsoleting indication | A | E |
| Originator indication | E | E |
| Prevention of non-delivery notification | A | N/A |
| Primary and copy recipients indication | E | E |
| Probe | A | N/A |
| Receipt notification | A | A |
| Reply request indication | A | E |
| Replying IP-message indication | E | E |
| Return of contents | A | N/A |
| Sensitivity indication | A | E |
| Subject indication | E | E |

E: Essential optional user facility
A: Additional optional user facility
N/A: Not applicable

TABLE 2/X.401

**IPM optional user facilities agreed for a contractual period of time**

| IMP optional user facilities | Categorization |
|---|---|
| Alternate recipient assignment | A |
| Hold for delivery | A |
| Implicit conversion | A |

The optional user facilities of the IPM service that are selected on a per-message basis are categorized for both origination and reception by UAs. If an Administration provides the IPM service and offers these optional user facilities for origination by UAs, then a user is able to create and send IP-messages according to the procedures defined for the associated service element. If an Administration provides the IPM service and offers these optional user facilities for reception by UAs, then the receiving UA will be able to receive and recognize the indication associated with the corresponding service element and to inform the user of the requested optional user facility. Each optional user facility is categorized as additional or essential for UAs from these two perspectives.

*Note* – With the Access Protocol described in Recommendation X.430, teletex terminals are able to make use of the basic IPM service as well as of the optional user facilities provided by the message handling system.

## 3 Message transfer service

### 3.1 *Basic service*

The basic MT service enables a UA to submit and to have messages delivered to it. If a message cannot be delivered, the originating UA is so informed. Each message is uniquely and unambiguously identified. To facilitate meaningful communication, a UA may specify the encoded information type(s) that can be contained in messages which are delivered to it. The content type and original encoded information type(s) of a message and an indication of any conversions, that may have been performed, and the resulting encoded information type(s) are supplied with each delivered message. In addition, the submission time and delivery time are supplied with each message. Non-delivery notification is provided with the basic service.

### 3.2 *MT optional user facilities*

The optional user facilities of the MT service, which may be selected on a per-message basis or for an agreed contractual period of time, are listed in Tables 3/X.401 and 4/X.401, respectively. Local user facilities may be usefully provided in conjunction with some of these facilities.

TABLE 3/X.401

**MT optional user facilities provided to the UA selectable on a per-message basis**

| MT optional user facilities | Categorization |
|---|---|
| Alternate recipient allowed | E |
| Conversion prohibition | E |
| Deferred delivery | E |
| Deferred delivery cancellation | E |
| Delivery notification | E |
| Disclosure of other recipients | E |
| Explicit conversion | A |
| Grade of delivery selection | E |
| Multi-destination delivery | E |
| Prevention of non-delivery notification | A |
| Probe | E |
| Return of contents | A |

TABLE 4/X.401

**MT optional user facilities provided to the UA agreed for a contractual period of time**

| MT optional user facilities | Categorization |
|---|---|
| Alternate recipient assignment | A |
| Hold for delivery | A |
| Implicit conversion | A |

E: Essential optional user facility
A: Additional optional user facility

ANNEX A

(to Recommendation X.401)

**Abbreviations**

The following abbreviations are used in this Recommendation.

IPM Interpersonal Messaging
MHS Message Handling System
MT Message Transfer
OSI OSI Systems Interconnection
UA User Agent

APPENDIX I

(to Recommendation X.401)

**List of terms**

The following terms are contained in this Recommendation. References to definitions, implicit or explicit, are included where appropriate and where available.

## A

**Additional Optional User Facility**
cont. in X.401

**Administration**
cont. in X.401

**Alternate Recipient Allowed**
cont. in X.401; impl. def. in X.400

**Alternate Recipient Assignment**
cont. in X.401; impl. def. in X.400

**Authorizing Users Indication**
cont. in X.401; impl. def. in X.400

**Auto-forwarded Indication**
cont. in X.401; impl. def. in X.400

## B

**Basic Service**
cont. in X.401

**Basic Service Element**
cont. in X.401

**Blind Copy Recipient Indication**
cont. in X.401; impl. def. in X.400

**Body Part Encryption Indication**
cont. in X.401; impl. def. in X.400

## C

**Conversion Prohibition**
cont. in X.401; impl. def. in X.400

**Cross-referencing Indication**
cont. in X.401; impl. def. in X.400

## D

**Deferred Delivery**
cont. in X.401; impl. def. in X.400

**Deferred Delivery Cancellation**
cont. in X.401; impl. def. in X.400

**Delivery Notification**
cont. in X.401; impl. def. in X.400

**Disclosure of Other Recipients**
cont. in X.401; impl. def. in X.400

## E

**Encoded Information Type**
cont. in X.401; impl. def. in X.400

**Essential Optional User Facility**
cont. in X.401; impl. def. in X.2

**Expiry Date Indication**
cont. in X.401; impl. def. in X.400

**Explicit Conversion**
cont. in X.401; impl. def. in X.400

## F

**Forwarded IP-message Indication**
cont. in X.401; impl. def. in X.400

## G

**Grade of Delivery Selection**
cont. in X.401; impl. def. in X.400

## H

**Hold for Delivery**
cont. in X.401; impl. def. in X.400

## I

**Implicit Conversion**
cont. in X.401; impl. def. in X.400

**Importance Indication**
cont. in X.401; impl. def. in X.400

**Interpersonal Messaging Service**
cont. in X.401; impl. def. in X.400

**IP-message**
cont. in X.401; impl. def. in X.400

## M

**Message**
cont. in X.401; impl. def. in X.400

**Message Attachment Designation**
cont. in X.401; impl. def. in X.400

**Message Handling**
cont. in X.401; impl. def. in X.400

**Message Handling System**
cont. in X.401; impl. def. in X.400

**Message Transfer Service**
cont. in X.401; impl. def. in X.400

**Multi-destination Delivery**
cont. in X.401; impl. def. in X.400

**Multi-part Body**
cont. in X.401; impl. def. in X.400

## N

**Non-delivery Notification**
cont. in X.401; impl. def. in X.400

**Non-receipt Notification**
cont. in X.401; impl. def. in X.400

## O

**Obsoleting Indication**
cont. in X.401; impl. def. in X.400

**Optional User Facility**
cont. in X.401; cont. in X.2; expl. def. in X.15

**Originator Indication**
cont. in X.401; impl. def. in X.400

## P

**Prevention of Non-delivery Notification**
cont. in X.401; impl. def. in X.400

**Primary and Copy Recipients Indication**
cont. in X.401; impl. def. in X.400

**Probe**
cont. in X.401; impl. def. in X.400

## R

**Receipt Notification**
cont. in X.401; impl. def. in X.400

**Reply Request Indication**
cont. in X.401; impl. def. in X.400

**Replying User-message Indication**
cont. in X.401; impl. def. in X.400

**Return of Contents**
cont. in X.401; impl. def. in X.400

## S

**Sensitivity Indication**
cont. in X.401; impl. def. in X.400

**Service Element**
cont. in X.401; impl. def. in X.400

**Subject Indication**
cont. in X.401; impl. def. in X.400

## U

**User Agent**
cont. in X.401; impl. def. in X.400

**User-message**
cont. in X.401; impl. def. in X.420

**Recommendation X.408**

### MESSAGE HANDLING SYSTEMS: ENCODED INFORMATION TYPE CONVERSION RULES

*(Malaga-Torremolinos, 1984)*

The establishment in various countries of telematic services and computer-based store-and-forward message services in association with public data networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT,

*considering*

(a) the need for interpersonal messaging and message transfer services;

(b) the need to transfer messages of different types having a large variety of formats;

(c) that the F-series of Recommendations defines telematic services and that the T-series of Recommendations defines terminal equipment and control procedures for telematic services;

(d) that the V-series of Recommendations provides the means for data communication over the telephone network;

(e) that Recommendation X.200 defines the reference model of open systems interconnection for CCITT applications;

(f) that a set of Recommendations describes various aspects of message handling: X.400, X.401, X.408, X.409, X.410, X.411, X.420 and X.430;

(g) that CCITT and ISO have drawn up a suitable set of conversion rules;

(h) that, for specific user applications, some variations may be developed and applied by bilateral agreement,

*unanimously declares*

(1) that the purpose and scope of this Recommendation are described in § 1;

(2) that general aspects of the rules for converting between encoded information types are described in § 2;

(3) that conversion rules for particular encoded information types are defined in subsequent sections.

CONTENTS

## 1 Purpose and scope

This Recommendation is one of a series of Recommendations and defines the encoded information type conversion rules for the message handling system (MHS). This Recommendation specifies the algorithms the MHS uses when converting between different types of encoded information.

Other Recommendatations in the series contain information essential to a proper understanding of the IPM service and the MT service. Recommendation X.401 outlines its basic service elements and optional user facilities. Recommendation X.400 describes the MHS model and defines the MHS service elements. Recommendation X.411 specifies the protocol aspects of the Message Transfer Layer (MTL). Recommendation X.420 specifies the protocol aspects of the Interpersonal Messaging Service. Recommendation X.409 defines the notation and representational technique used to specify and encode MHS protocols. Recommendation X.430 describes how Teletex terminals access the MHS. Recommendation X.410 describes general techniques used for MHS protocols and the way in which standard open systems interconnection (OSI) protocols are used to support MHS applications.

§ 2 of this Recommendation describes the general aspects of conversion for the MHS. In §§ 3 through 11, particular conversion rules are defined. § 12 lists documents that supplement this Recommendation. Annex A lists the abbreviations used.

## 2 General aspects of conversion

Among the data elements subject to conversion are the subject and body of a message. The conversion has two aspects, format and code. The aspect of control is described elsewhere in the accompanying Recommendations.

### 2.1 *Encoded information types*

This Recommendation defines the conversion rules for the nine types of encoded information utilized in the MHS. In order to refer to each type, the following terms are used:

TLX: The code is defined in Recommendation F.1. The format is defined in Recommendation S.5.

IA5 Text: The code is defined in Recommendation T.50.

TTX: The format is defined in Recommendations F.200 and T.60, and the code is defined in Recommendation T.61.

G3 Fax: The encoding scheme is defined in Recommendation T.4, and the singalling of the encoding scheme is defined in Recommendation T.30.

TIF0: The format and encoding scheme are defined in Recommendation T.73.

Videotex: The code is defined in Recommendations T.100 and T.101.

Voice: The encoding scheme is for further study.

SFD: The format is defined in Recommendation X.420, and the code is defined in Recommendation T.61.

TIF1: The format and encoding scheme are defined in Recommendation T.73.

*Note* – TLX: Telex, TTX: Teletex, SFD: Simple Formattable Document, TIF: Text Interchange Format.

The TTX and G3 Fax types have two subtypes: basic and optional. The TIF0 type and the TIF1 type have two subtypes: basic and non-basic.

Table 1/X.408 depicts all conceivable conversions between the above types and subtypes. It characterizes each as: (1) unnecessary, (2) possible without loss of information, (3) possible but with loss of information, or (4) impractical. This Recommendation defines the rules for format and code conversion for conversions in the second and third categories.

### 2.2 *Two aspects of conversion*

The conversion rules have two aspects:

- the format aspect,
- the code aspect.

TABLE 1/X.408

**Encoded information type conversions**

| | From \ To | TLX | IA5 Text | TTX | | G3 Fax | | TIF0 | | Videotex | Voice | SFD | TIF1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Basic | Optional [1] | Basic | Optional [1] | Basic | Non basic [1] | | | | Basic | Non basic [1] |
| | TLX [5] | – | a | a | a | a | a | a | a | b | b | a | a | a |
| | IA5 Text | b | – | b | b | a | a | a | a | b | b | a | b | b |
| TTX | Basic | b | b | – | a | a[4] | a[4] | a[4] | a[4] | a | b | b | a | a |
| | Optional [1] | b | b | b | b [2], [3] | a[4] | a[4] | a[4] | a[4] | a | b | b | a | b [2], [3] |
| G3 Fax | Basic | c | c | c | c | – | a | a | a | c [6] | c | c | a | a |
| | Optional [1] | c | c | c | c | b | b [2], [3] | b | b | c [6] | c | c | b | b |
| TIF0 | Basic | c | c | c | c | a [4] | a [4] | – | a | c [6] | c | c | a | a |
| | Non basic [1] | c | c | c | c | b | b | b | b [2], [3] | c [6] | c | c | b | b [2], [3] |
| | Videotex | b | b | b | b | a [7] | a [7] | a [7] | a [7] | **FS** | **FS** | **FS** | **FS** | **FS** |
| | Voice | c | c | c | c | c | c | c | c | c | **FS** | c | c | c |
| | SFD | b | b | a | a | a | a | a | a | b | **FS** | – | a | a |
| TIF1 | Basic | b | b | b | b | a [4] | a [4] | a | a | b | b | b | – | a |
| | Non basic [1] | b | b | b | b [2], [3] | a [4] | a [4] | b | b [2], [3] | b | b | b | b | b [2], [3] |

– No conversion.

a Possible without loss of information.

b Possible with loss of information.

c Impratical.

**FS** For further study.

[1] Specified in the relevant Recommendations.

[2] No information is lost if the originating and recipient terminals have the same optional functions.

[3] Information may be lost if the originating terminal uses optional functions that the recipient terminal lacks.

[4] Information may be lost due to the difference between the printable and reproducible areas.

[5] The WHO ARE YOU character is assumed to be a protocol element used for communicating with the telex terminal and not part of the message's content.

[6] It may be possible with loss of information, if the recipient terminal has the capability of the photographic type of information.

[7] When converting videotex, color information may be lost.

2.2.1 *Ground rules*

If there is an existing standard on the conversion between different types, it should be referred to without any modifications unless required. If there is not, the following ground rules are applied:

1) If there are standards on the subject types, the conversion rule should be defined such that the intersecting part of the standards is preserved. The creation of new rules for non-intersecting parts should be based on clear requirements; otherwise they should not be created.
2) When either the subject or object type has no standard, the conversion rules should be defined such that standard types can be accommodated as much as possible for both directions of the conversion.
3) When neither type has a standard, the definition of the rules is for further study.

2.2.2 *Format aspect*

The format aspect represents the dimensional attributes of the Presentation space of user messages.

The two-dimensional (X and Y) aspect of the conversion is to be specified for a message being transferred. Following are the parameters to be defined for this aspect. Whether Voice must consider this aspect is for further study.

The X-direction of the Presentation space is defined by means of either:

a) the size of a character and the number of characters to be presented,
b) length.

The Y-direction of the Presentation space is defined by means of either:

a) the number of lines per Presentation space or per unit length,
b) length.

2.2.3 *Code aspect*

With respect to the code aspect, the following classification is used to specify the conversion between different types:

a) Intersecting repertoires;
b) Non-intersecting repertoires.

For b) above, further definition should be based on the purpose for which they are to be used and the ground rules in § 2.2.1:

a) Graphic repertoires;
b) Control repertoires.

For simplicity and clarity, rules should be tabulated.

## 3 Conversion from TLX

3.1 *Conversion from TLX to IA5 Text*

3.1.1 *Format conversion*

This requires further study.

3.1.2 *Code conversion*

This conversion rule is defined in Recommendation S.18.

3.2 *Conversion from TLX to TTX*

3.2.1 *Format conversion*

This requires further study.

3.2.2 *Code conversion*

This conversion rule is defined in Recommendation T.60.

3.3 *Conversion from TLX to G3 Fax*

This requires further study.

3.4 *Conversion from TLX to TIF0*

This requires further study.

3.5 *Conversion from TLX to Videotex*

This requires further study.

3.6 *Conversion from TLX to Voice*

This requires further study.

3.7 *Conversion from TLX to SFD*

This requires further study.

3.8 *Conversion from TLX to TIF1*

This requires further study.

## 4 Conversion from IA5 text

4.1 *Conversion from IA5 Text to TLX*

4.1.1 *Format conversion*

This requires further study.

4.1.2 *Code conversion*

This conversion rule is defined in Recommendation S.18.

4.2 *Conversion from IA5 Text to TTX*

4.2.1 *Format conversion*

An IA5 String is converted into one or more teletex pages in the format defined by the following "default condition of basic teletex". A form feed in the IA5 String provokes a new teletex page:

Paper size and orientation: vertical basic page format.
Character spacing: 2.54 mm.
Line feed spacing: 4.23 mm.
Rendition: default rendition.

This implies that the maximum number of characters per line is 72 and the maximum number of lines per page is 55.

4.2.2 *Code conversion*

4.2.2.1 *Graphic character conversion*

Every IA5 character is represented by seven bits ($b_7$-$b_1$). Most graphic characters of IA5 are converted into the corresponding primary graphic characters of T.61 by adding 0 as the eighth bit ($b_8$).

Table 2/X.408 shows the IA5 graphic characters that are not subject to this rule. They are converted into supplementary graphic characters of T.61.

*Note* – In the case of upward arrow head, grave accent, and overline, whether diacritical marks of T.61 can be chosen as converted codes is for further study.

TABLE 2/X.408

**Conversion to supplementary graphic characters of T.61**

| Name | IA5 | | T.61 | |
|---|---|---|---|---|
| | Graphic | Position | Graphic | Position |
| Number sign | # | 2/3 | # | 10/6 |
| Currency sign | ¤ | 2/4 | ¤ | 10/8 |
| Reverse solidus | \ | 5/12 | **FS** | **FS** |
| Upward arrow head | ^ | 5/14 | **FS** | **FS** |
| Grave accent | ` | 6/0 | **FS** | **FS** |
| Left curly bracket | { | 7/11 | « | 10/11 |
| Right curly bracket | } | 7/13 | » | 11/11 |
| Overline or tilde | ~ | 7/14 | **FS** | **FS** |

**FS**: for further study.

4.2.2.2 *Format effector conversion*

The conversion of IA5 format effectors is shown in Table 3/X.408.

TABLE 3/X.408

**Conversion of IA5 format effectors**

| Meaning | IA5 | | T.61 | |
|---|---|---|---|---|
| | Abbreviation | Position | Abbreviation | Position |
| Space | SP | 2/0 | SP | 2/0 |
| Backspace | BS | 0/8 | BS | 0/8 |
| Line feed | LF | 0/10 | LF | 0/10 |
| Carriage return | CR | 0/13 | CR | 0/13 |
| Form feed | FF | 0/12 | CR, FF (on new page) | 0/13, 0/12 |
| Horizontal tabulation | HT | 0/9 | **FS** | **FS** |
| Vertical tabulation | VT | 0/11 | **FS** | **FS** |

**FS**: for further study.

4.2.2.3 *Other control character conversion*

DEL (7/15) is discarded.

A transmission control character, a device control character, or an information separator in the original IA5 String is represented by one T.61 graphic character whose code is for further study.

4.3 *Conversion from IA5 Text to G3 Fax*

Requires further study.

4.4 *Conversion from IA5 Text to TIF0*

Requires further study.

4.5 *Conversion from IA5 Text to Videotex*

Requires further study.

4.6 *Conversion from IA5 Text to Voice*

Requires further study.

4.7 *Conversion from IA5 Text to SFD*

Requires further study.

4.8 *Conversion from IA5 Text to TIF1*

Requires further study.

## 5 Conversion from TTX

5.1 *Conversion from TTX to TLX*

5.1.1 *Format conversion*

Requires further study.

5.1.2 *Code conversion*

This conversion rule is defined in T.60.

5.2 *Conversion from TTX to IA5 Text*

5.2.1 *Format conversion*

A teletex document is converted into an IA5 String assuming the vertical orientation and a maximum of 77 characters per line (a line may be constructed by placing 72 characters to the right of the left margin and an additional 5 characters to the left of the left margin) and an unlimited number of lines per page. Teletex information in the horizontal orientation will result in loss of information.

A page boundary in the original teletex document is represented by two IA5 control characters: Carriage Return (CR, 0/13) and Form Feed (FF, 0/12).

5.2.2 *Code conversion*

5.2.2.1 *Graphic character conversion*

Every graphic character in the primary set of T.61 is converted into the corresponding graphic character of IA5 by deleting bit $b_8$. The conversion from graphic characters in the supplementary set of T.61 to graphic characters of IA5 is shown in Table 4/X.408.

TABLE 4/X.408

**Conversion from supplementary graphic characters of T.61**

| Name | T.61 | | IA5 | |
|---|---|---|---|---|
| | Graphic | Position | Graphic | Position |
| Cent sign | ¢ | 10/2 | ¤ | 2/4 |
| Pound sign | £ | 10/3 | ¤ | 2/4 |
| Dollar sign | $ | 10/4 | ¤ | 2/4 |
| Yen sign | ¥ | 10/5 | ¤ | 2/4 |
| General currency sign | ¤ | 10/8 | ¤ | 2/4 |
| Number sign | # | 10/6 | # | 2/3 |
| Angle quotation mark left | « | 10/11 | { | 7/11 |
| Angle quotation mark right | » | 11/11 | } | 7/13 |
| *Diacritical marks* | | 12/0-12/15 | **FS** | **FS** |
| *Others* | *Others* | *Others* | ? | 3/15 |

**FS**: for further study.

*Note* – Other conversion rules for the currency signs are for further study.

5.2.2.2 *Control function conversion*

*Format effectors*

The conversion of format effectors is as shown in Table 5/X.408.

*Presentation control functions*

A control sequence that represents Page Format Selection (PFS), Select Graphic Rendition (SGR), Select Horizontal Spacing (SHS), or Select Vertical Spacing (SVS) is discarded.

*Miscellaneous control functions*

A control sequence that represents Substitute Character (SUB) or Identify Graphic Subrepertoire (IGS) is discarded.

*Non-basic control functions*

Reverse Line Feed (RLF) is represented by one IA5 graphic character whose code is for further study.

TABLE 5/X.408

**Conversion of T.61 format effectors**

| Meaning | T.61 | | IA5 | |
|---|---|---|---|---|
| | Abbreviation | Position | Abbreviation | Position |
| Space | SP | 2/0 | SP | 2/0 |
| Backspace | BS | 0/8 | BS | 0/8 |
| Line feed | LF | 0/10 | LF | 0/10 |
| Form feed | FF | 0/12 | FF | 0/12 |
| Carriage return | CR | 0/13 | CR | 0/13 |
| Partial line down | PLD | 8/12 | **FS** | **FS** |
| Partial line up | PLU | 8/13 | **FS** | **FS** |

**FS**: for further study.

5.3 *Conversion from TTX to TTX*

Requires further study.

5.4 *Conversion from TTX to G3 Fax*

Requires further study.

5.5 *Conversion from TTX to TIF0*

Requires further study.

5.6 *Conversion from TTX to Videotex*

Requires further study.

5.7 *Conversion from TTX to Voice*

Requires further study.

5.8 *Conversion from TTX to SFD*

5.8.1 *Format conversion*

The teletex document is converted into an SFD by adding the following SFD descriptors to the beginning of the teletex document:

{**specificLogicalDescriptor** {**document,** {}},
**specificLogicalDescriptor** {**paragraph,** {**presentationDirectives** {**alignmentleft Aligned**}}},
**textUnit** {},

*The Teletex T.61 String follows without change.*}}

5.8.2 *Code conversion*

Not required. SFD uses T.61 String.

5.9 *Conversion from TTX to TIF1*

5.9.1 *Format conversion*

Requires further study.

5.9.2 *Code conversion*

Not required. T.61 String is allowed in TIF1.

## 6 Conversion from G3 Fax

6.1 *Conversion from G3 Fax to G3 Fax*

Requires further study.

6.2 *Conversion from G3 Fax to TIF0*

Requires further study.

6.3 *Conversion from G3 Fax to TIF1*

Requires further study.

## 7 Conversion from TIF0

7.1 *Conversion from TIF0 to G3 Fax*

Requires further study.

7.2 *Conversion from TIF0 to TIF1*

Requires further study.

7.3 *Conversion from TIF0 to TIF1*

Requires further study.

## 8 Conversion from Videotex

8.1 *Conversion from Videotex to TLX*

Requires further study.

8.2 *Conversion from Videotex to IA5 Text*

Requires further study.

8.3 *Conversion from Videotex to TTX*

Requires further study.

8.4 *Conversion from Videotex to G3 Fax*

Requires further study.

8.5 *Conversion from Videotex to TIF0*

Requires further study.

8.6 *Conversion from Videotex to Videotex*

Requires further study.

8.7 *Conversion from Videotex to Voice*

Requires further study.

8.8 *Conversion from Videotex to SFD*

Requires further study.

8.9 *Conversion from Videotex to TIF1*

Requires further study.

## 9 Conversion from Voice

9.1 *Conversion from Voice to Voice*

Requires further study.

## 10 Conversion from SFD

10.1 *Conversion from SFD to TLX*

10.1.1 *Format conversion*

The SFD logical descriptors and logical attribute specifications must be removed from each text unit.

If a heading has been supplied, it will be output once, followed by two CR-LF pairs from ITA2. The text unit characters will follow. If the paragraph option of left-margin has been selected, the ITA2 SP will be output on each line the number of units requested. This will reduce the number of graphics on the line in order to maintain no more than 69 per line.

Each text unit will be organized into lines of no more than 69 characters each. A CR-LF pair from ITA2 will be inserted at these points unless the T.61 CR-LF pair is already there. In this case, they are converted to ITA2 code.

At SFD paragraph object boundaries, one extra CR-LF pair is inserted if the one-blank-line option was chosen.

The T.61 FF (Form Feed) is converted to an ITA2 CR-LF pair.

10.1.2 *Code conversion*

The code conversion between TLX and TTX has been specified in Recommendation T.60 and should be followed for SFD T.61 Strings.

10.2 *Conversion from SFD to IA5 Text*

10.2.1 *Format conversion*

The SFD logical descriptors and logical attribute specifications must be removed from each text unit. Each text unit will be organized into lines of no more than 72 characters each. A CR-LF pair from IA5 will be inserted at these points unless the T.61 CR-LF pair is already there. In this case, they are converted to IA5 code.

At SFD paragraph object boundaries, one extra CR-LF pair is inserted if the option to add one blank line has been chosen. If the left-margin option has been selected, the IA5 SP will be output at the beginning of each line until the end of the current paragraph object. The SP characters must be included in the count of no more than 72 characters per line.

When the SFD document object is removed, if a heading has been specified, it will be output once before any text unit characters. The heading text, if present, will be followed by two CR-LF pairs from IA5.

*Note* – Inclusion of 40, 69 or 80 characters per line or other figures is for further study.

10.2.2 *Code conversion*

10.2.2.1 *Graphic character conversion*

Every graphic character in the primary set of T.61 is converted into the corresponding graphic character of IA5 by deleting bit $b_8$. The conversion from graphic characters in the supplementary set of T.61 to graphic characters of IA5 is defined in § 5.2.2.1.

10.2.2.2 *Control function conversion*

The conversion rules are defined in § 5.2.2.2.

10.3 *Conversion from SFD to TTX*

10.3.1 *Format conversion*

The SFD logical descriptors and logical specifications must be removed from each text unit.

When the SFD document object is removed, if a heading has been specified, it will be output at the beginning of each TTX page followed by two CR-LF pairs. The heading text, when present, will reduce the number of lines per page by the number of heading lines plus two.

Paragraph objects with left-margin specification will have the appropriate number of SPs output at the beginning of each line of the paragraph ahead of any text unit characters on the line. The graphic rendition request for underlining will cause the SGR with a parameter value of 4 to be output. At the end of the paragraph, an SGR with parameter 0 will restore normal rendition.

An SFD string is converted into one or more Teletex pages in the format defined by the following "default condition of basic Teletex". A form feed in the string provokes a new Teletex page:

Paper size and orientation: vertical basic page format.

Character spacing: 2.54 mm.

Line feed spacing: 4.23 mm.

Rendition: default rendition.

This implies that the maximum number of characters per line is 72 and the maximum number of lines per page is 55.

10.3.2 *Code conversion*

No code conversion is necessary; both use T.61.

10.4 *Conversion from SFD to G3 Fax*

Requires further study.

10.5 *Conversion from SFD to TIF0*

Requires further study.

10.6 *Conversion from SFD to Videotex*

Requires further study.

10.7 *Conversion from SFD to Voice*

Requires further study.

10.8 *Conversion from SFD to TIF1*

Requires further study.

## 11 Conversion from TIF1

11.1 *Conversion from TIF1 to TLX*

Requires further study.

11.2 *Conversion from TIF1 to IA5 Text*

Requires further study.

11.3 *Conversion from TIF1 to TTX*

Requires further study.

11.4 *Conversion from TIF1 to G3 Fax*

Requires further study.

11.5 *Conversion from TIF1 to TIF0*

Requires further study.

11.6 *Conversion from TIF1 to Videotex*

Requires further study.

11.7 *Conversion from TIF1 to Voice*

Requires further study.

11.8 *Conversion from TIF1 to SFD*

Requires further study.

11.9 *Conversion from TIF1 to TIF1*

Requires further study.

ANNEX A

(to Recommendation X.408)

**Abbreviations**

The following abbreviations are used in this Recommendation:

| | |
|---|---|
| BS | Backspace |
| CR | Carriage return |
| FF | Form feed |
| G3 | Group 3 |
| G3 Fax | Group 3 facsimile type |
| HT | Horizontal tabulation |
| IA | International alphabet |
| IA5 | International alphabet No. 5 |
| IGS | Identify graphic subrepertoire |
| ITA | International telegraph alphabet |
| ITA2 | International telegraph alphabet No. 2 |
| LF | Line feed |
| MHS | Message handling system |
| OSI | Open systems interconnection |
| PFS | Page format selection |
| PLD | Partial line down |
| PLU | Partial line up |
| RLF | Reverse line feed |
| SFD | Simple formattable document type |
| SGR | Select graphic rendition |
| SHS | Select horizontal spacing |
| SP | Space |
| SUB | Substitute character |
| SVS | Select vertical spacing |
| TIF | Text interchange format |
| TLX | Telex type |
| TTX | Teletex type |
| VT | Vertical tabulation |

APPENDIX I

(to Recommendation X.408)

**List of Terms**

The following terms are contained in this Recommendation. References to definitions, implicit or explicit, are included where appropriate and available.

## C

**Character Spacing**
cont. in X.408

**Code**
cont. in X.408; expl. def. in T.61

**Code Conversion**
cont. in X.408; expl. def. in X.408

**Control Function Conversion**
cont. in X.408; impl. def. in X.408

**Conversion**
cont. in X.408

**Conversion Rules**
cont. in X.408; impl. def. in X.408

## D

**Default Condition of Basic Teletex**
cont. in X.408; impl. def. in T.62

## E

**Encoded Information Types**
cont. in X.408; impl. def. in X.400

## F

**Format**
cont. in X.408

**Format Conversion**
cont. in X.408; impl. def. in X.408

## G

**Group 3 Fascimile Apparatus**
cont. in X.408; impl. def. in T.4

**Graphic Character Conversion**
cont. in X.408; impl. def. in X.408

**Graphic Repertoires**
cont. in X.408; impl. def. in T.61

## I

**IA No. 5 Code**
cont. in X.408; impl. def. in T.50

**IA5 Text**
cont. X.408

**ITA No. 2 Code**
cont. in X.408; impl. def. in F.1

## M

**Message**
cont. in X.408; impl. def. in X.400

**Message Handling System**
cont. in X.408; impl. def. in X.400

## P

**Presentation Space**
cont. in X.408

## S

**Simple Formattable Document**
cont. in X.408; impl. def. in X.420

**Supplementary Graphic Characters**
cont. in X.408

## T

**Teletex**
cont. in X.408; impl. def. in F.200

**Telex**
cont. in X.408

**Text Interchange Format**
cont. in X.408; impl. def. in T.73

## V

**Videotex**
cont. in X.408; impl. def. in F.300

**Voice**
cont. in X.408

# MESSAGE HANDLING SYSTEMS: PRESENTATION TRANSFER SYNTAX AND NOTATION

*(Malaga-Torremolinos, 1984)*

The establishment in various countries of telematic services and computer-based store-and-forward message services in association with public data networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT,

*considering*

(a) the need for interpersonal messaging and message transfer services;

(b) the need to transfer messages of different types having a large variety of formats;

(c) that the T-series of Recommendations defines control procedures for telematic services;

(d) that Recommendation X.200 defines the reference model of open systems interconnection for CCITT applications;

(e) that a set of Recommendations describes various aspects of message handling: X.400, X.401, X.408, X.409, X.410, X.411, X.420 and X.430,

*unanimously declares*

(1) that general aspects of the presentation transfer syntax and notation are described in § 1;

(2) that the presentation notation is described in §§ 2 and 4;

(3) that the presentation transfer syntax is described in § 3;

(4) that generally useful data types are described in §§ 5 and 6.

CONTENTS

## 1 Purpose and scope

This Recommendation defines the presentation transfer syntax used by application layer protocols in message handling systems and by the document interchange protocol for the telematic services. In the architecture of open systems interconnection (OSI), a presentation transfer syntax is used to represent information exchanged between application entities.

This Recommendation defines a transfer syntax for various kinds of information. Each piece of information is considered to have a type as well as a value. A *data type* (or *type* for short) is a class of information (for example, numeric or textual). A *data value* (or *value* for short) is an instance of such a class (for example, a particular number or fragment of text). This Recommendation defines several generally useful types (for example, Boolean, Integer, and IA5 String) from which application-specific types are constructed in other Recommendations (for example, the Message Protocol Data Units defined in Recommendation X.411).

This Recommendation presents and gives an example of the intended use, standard notation and standard representation for each type. A type's *standard notation* (or *notation* for short) is the set of conventions employed in other Recommendations to denote either the type itself or a value of that type. The type notation is used to specify the structure of objects defined by the protocols. Use of this notation permits all of the encoding details to be confined to this Recommendation. The value notation is used to give examples of such objects and to specify distinguished values of them, again without the need to reference their encoding details. A type's *standard representation* (or *representation* for short) is the set of rules for encoding values of that type for transmission as a sequence of octets. The representation of a value also encodes its type and length.

§§ 2 and 3 of this Recommendation describe in general terms the standard notation and standard representation, respectively. § 4 introduces generally useful notation for defining types, values, macros, and modules. § 5 defines built-in types, useful in their own right and also as a foundation for defining further types. § 6 defines further generally useful types, using the built-in types of § 5.

## 2 Descriptive conventions for the standard notation

The standard notation for each type is defined in this Recommendation using Backus-Naur Form (BNF). The BNF description of any formal language comprises a series of replacement rules called *productions.* By following the replacement rules, all valid instances of the language can be produced. Three classes of symbols appear in a production: terminals, non-terminals, and operators. A *terminal* is a symbol that appears literally in the language being described. A *non-terminal* is a symbol that is defined to be equivalent to either a particular series of symbols – either terminal or non-terminal – or any of several such series; all but a few non-terminals are defined by means of productions. Two *operators* are used in productions. The *equivalence* operator (":: =") assigns a value to a non-terminal. The *alternative* operator ("|") distinguishes among several alternative series of symbols, one of which may be selected. For example, the notation for the Boolean type is the keyword **BOOLEAN**, and the notation for a Boolean value is either of two keywords, **TRUE** or **FALSE**. This is indicated in BNF as follows:

**BooleanType** :: = **BOOLEAN**

**BooleanValue** :: = **TRUE** | **FALSE**

Symbols rendered in **bold** are non-terminals; all other symbols are terminals. The terminals ":: =", "|", "string", "identifier", "number" and "empty" are quoted to distinguish them from the BNF operators, and the built-in non-terminals listed immediately below. Non-terminals whose first character is a capital letter are defined in the grammar; other non-terminals, of which there are four – **string**, **identifier**, **number** and **empty** – are defined below.

The non-terminal **string** is a sequence of zero or more characters.

The non-terminal **identifier** is a sequence of one or more characters chosen from the capital letters, the small letters, the decimal digits, and the hyphen; the first character must be a letter. The distinction between a small letter and the corresponding capital letter is significant and distinguishes one identifier from another.

The non-terminal **number** denotes a non-negative integer and has two forms. The first specifies the integer's value in decimal (that is, radix 10) notation; it is a sequence of one or more decimal digits. The second specifies the integer's value in hexadecimal (that is, radix 16) notation; it is a sequence of one or more hexadecimal digits (that is, decimal digits and capital letters in the range "A" through "F"), followed by the capital letter "H".

*Note* – For clarity of exposition, where the standard notation is not used in this document, binary values are subscripted with "2", hexadecimal values are subscripted with "16", and decimal values appear without subscript.

The non-terminal **empty** denotes the null or empty string of symbols, that is, a string of zero symbols. **empty** is most commonly used in conjunction with the alternative operator. For example (see § 5.7), the case of a Set with no members is allowed for as follows:

**MemberValues :: = NamedValueList | empty**

Comments may be embedded in the notation. They are preceded by two hyphens ("–") and terminated by either two hyphens or the end of a line (see IA5).

The standard notation for particular data types is specified in § 5.

*Note* – It is emphasized for clarity that two notations appear in this document: on the one hand the standard notation, and on the other the Backus-Naur Form used to *define* the standard notation. Care should be taken not to confuse the two. As an example the Choice data type (see § 5.9) provides a means for selecting between alternative data types, for example, in a protocol data unit. In contrast, the BNF alternative operator is used to select between alternative forms of standard notation. The alternative operator is used in the *definition* of the standard notation for Choice, but is not itself part of that standard notation.

## 3 Standard representation

The standard representation for a value of each type is a *data element* (or *element* for short) having three components, which always appear in the following order. The *Identifier* distinguishes one type from another and governs the interpretation of the Contents. The *Length* specifies the length of the Contents. The *Contents* is the substance of the element, containing the primary information the element is intended to convey.

The representation for a Boolean, for example, is an element whose Identifier is $01_{16}$ and whose Contents is a single octet that encodes the value of the Boolean. **FALSE** is encoded as all bits zero, **TRUE** as any other combination of bits.

An element or any of its components may be viewed as a sequence of an integral number, $n$, of octets numbered 0 through $n - 1$. Each octet may be viewed in turn as a sequence of eight bits numbered 8 through 1. Throughout this Recommendation, Octet $i - 1$ is shown to the left of Octet $i$ and Bit $j + 1$ to the left of Bit $j$ on the printed page, and the adjectives "first" and "last" appeal to these particular octet and bit orderings. The abbreviations "MSB" and "LSB" stand for "most significant bit" and "least significant bit", respectively. The situation is summarized in Figure 1/X.409.

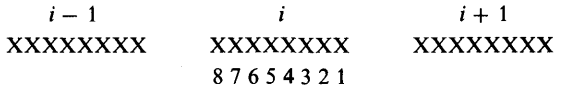



FIGURE 1/X.409

**Numbering of octets and bits**

### 3.1 *Identifier*

The Identifier distinguishes one type from another and governs interpretation of the Contents. It is one or more octets in length.

Four *classes* of types are distinguished by means of the Identifier: universal, application-wide, context-specific, and private-use. *Universal* types are generally useful, application-independent types; they are defined in §§ 5-6 of this Recommendation. *Application-wide* types are more specialized, being peculiar to a particular application; they are defined in companion Recommendations by means of the Tagged type (see § 5.8). *Context-specific* types, like application-wide types, are peculiar to an application and are defined using Tagged. However, they are used only within an even more limited context – for example, that of a Set (see § 5.7) – and their Identifiers are assigned so as to be distinct only within that limited context. *Private-use* types are reserved for private use; the assignment of specific private-use Identifiers is outside the scope of this Recommendation.

The four type classes are distinguished by means of the first and second bits (that is, Bits 8-7) of the first Identifier octet. Table 1/X.409 indicates the particular bit combination assigned to each class.

TABLE 1/X.409

**Assignment of identifier class bits**
**(Bits 8-7, cc)**

| Class | Bit assignment |
|---|---|
| Universal | $00_2$ |
| Application-wide | $01_2$ |
| Context-specific | $10_2$ |
| Private-use | $11_2$ |

Two *forms* of data elements are distinguished by means of the Identifier: primitive and constructor. A *primitive* element is one whose Contents is atomic, that is, has no further internal structure of data elements. A *constructor* element is one whose Contents is itself a data element, or a series of data elements. Constructor elements are thus recursively defined.

The two element forms are distinguished by means of the third bit (that is, Bit 6) of the first Identifier octet. Table 2/X.409 indicates the particular bit setting assigned to each form.

TABLE 2/X.409

**Assignment of identifier form bit**
**(Bit 6, f)**

| Form | Bit assignment |
|---|---|
| Primitive | 0 |
| Constructor | 1 |

The remaining five bits (that is, Bits 5-1) of the first Identifier octet encode a numeric *ID Code* that distinguishes one data type from another of the same class. These bits encode ID Codes in the range 0-30 as unsigned binary numbers whose MSB and LSB are Bit 5 and Bit 1, respectively. If the ID Code is greater than 30, Bits 5-1 have the value $11111_2$, and the ID Code is encoded in one or more *extension* octets. The situation is as shown in Figure 2/X.409.

$00000_2$ ID Code

...

$11110_2$ ID Code

$11111_2$ Extension octet(s) follow

FIGURE 2/X.409

**Values of Identifier Bits 5-1 (iiiii)**

Bit 8 of each extension octet indicates whether it is the last: Bit 8 of the last octet is zero; Bit 8 of each preceding extension octet is one. Bits 7-1 of the extension octets collectively encode ID Codes greater than 30. Conceptually, these groups of bits are concatenated to form an unsigned binary number whose MSB is Bit 7 of the first extension octet and whose LSB is Bit 1 of the last extension octet. The ID Code shall be encoded in the fewest possible octets, that is, with no leading octets with Bits 7-1 all 0.

The structure of the Identifier is summarized in Figure 3/X.409, in which the bits denoted by "c" encode the class of the data type, the bit denoted by "f" encodes the form of the element, and the bits denoted by "i" encode the ID Code assigned to the type.

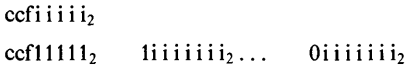



FIGURE 3/X.409

**Single- and multi-octet identifiers**

## 3.2 *Length*

The Length specifies the length in octets $L$ of the Contents and is itself variable in length.

The Length may take any of three forms: short, long, and indefinite. The *short* form is one octet long and shall be used, in preference to the long form, when $L$ is less than 128. Bit 8 has the value zero, and Bits 7-1 encode $L$ as an unsigned binary number whose MSB and LSB are Bit 7 and Bit 1, respectively.

Example: $L = 38$ is encoded as: $00100110_2$

The *long* form is from 2 to 127 octets long and is used when $L$ is greater than or equal to 128 and less than $2^{1008}$, except where the indefinite form is used. Bit 8 of the first octet has the value one. Bits 7-1 of the first octet encode a number one less than the size of the Length in octets as an unsigned binary number whose MSB and LSB are Bit 7 and Bit 1, respectively. $L$ itself is encoded as an unsigned binary number whose MSB and LSB are Bit 8 of the second octet and Bit 1 of the last octet, respectively. This binary number shall be encoded in the fewest possible octets, with no leading octets having the value 0.

Example: $L = 201$ is encoded as: $10000001_2\ 11001001_2$

In the case of the long form, this Recommendation makes no statement on the maximum length that must be supported. It is expected that each application using this Recommendation will establish its own maximum as appropriate.

The *indefinite* form is one octet long and may (but need not) be used in place of the short or long form, whenever the element is a constructor. It has the value $10000000_2$. When this form is employed, a special *end-of-contents (EOC)* element terminates the Contents.

There is no notation for the end-of-contents element. Although considered part of the Contents syntactically, the end-of-contents element has no semantic significance and is not mentioned in the descriptions of particular types that follow.

The representation for the end-of-contents element is an element whose class is universal, whose form is primitive, whose ID Code has the value 0, and whose Contents is unused and absent:

| EOC | *Length* | *Contents* |
|---|---|---|
| $00_{16}$ | $00_{16}$ | *Absent* |

*Note 1* – All three Length forms are valid for all applications. Where the above rules allow several forms to be used, the choice of form is left up to the entity that encodes the data element.

*Note 2* – No valid form of Length has the first (or only) octet equal to $11111111_2$. This value is reserved for possible future extension.

## 3.3 *Contents*

The Contents is the substance of the element and contains the primary information the element is intended to convey. It is variable in length, but always a multiple of eight bits, and is interpreted in a type-dependent way. If the element is a constructor, the Contents itself comprises zero or more elements; elements are thus recursively defined.

In some cases, the Contents comprises an ordered series of entities – the bits of a bit string, the octets of an octet string, or the elements of a sequence. In these cases, the order in which entities appear in the representation mirrors the order in which they appear in the notation. The entity occupying the left-most position on the printed page appears first in the Contents. The entity occupying the right-most position appears last.

## 4 Type, value, macro, and module definition

Standard notation is provided for the definition of types, values, macros, and collections of such definitions, as well as for types and values themselves. This additional notation, used in companion Recommendations and in § 6 of the present Recommendation, is presented below.

### 4.1 *Type definition*

Types are defined using the following notation. Such *defined types* are fabricated from the *built-in* types specified in § 5 and from other defined types. Defining a type assigns a reference name to it. By convention, each type name begins with a capital letter:

**TypeDefinition :: = identifier ":: =" Type**

Example:

**PrimaryColor :: = INTEGER {red(0), yellow(1), blue(2)}**

Wherever a type specification is called for by the standard notation, the reference name of a defined type may be supplied in its place. This aspect of the standard notation is not shown explicitly in the body of this Recommendation, but does appear in the notation summary of Annex B.

### 4.2 *Value definition*

Values are defined using the following notation. Defining a value assigns a reference name to it. By convention, each value name begins with a small letter:

**ValueDefinition :: = identifier Type ":: =" Value**

Example:

**defaultColor PrimaryColor :: = yellow**

Wherever a value specification is called for by the standard notation, the reference name of a defined value may be supplied in its place. Similarly, in most places where a number is required, the reference name of a defined Integer value may be supplied in its place. These two aspects of the standard notation are not shown explicitly in the body of this Recommendation, but do appear in the notation summary of Annex B.

### 4.3 *Macro definition*

It is occasionally useful to define *non* standard type and/or value notation for a particular data type. Such notation is defined by means of a *macro*. Macros are themselves defined using the notation specified below. Defining a macro also assigns a reference name to it. This name is always the first symbol in the type notation defined by the macro. By convention, each macro name is all capital letters:

**MacroDefinition :: = identifier MACRO ":: =" BEGIN MacroBody END**

The body of a macro specifies the desired non-standard notation using BNF (see § 2). Thus the body comprises a series of productions. The first production specifies the non-standard type notation, the second specifies the non-standard value notation, and the remaining productions define any non-terminals introduced by the first two productions:

**MacroBody :: = TypeProduction ValueProduction SupportingProductions**

**TypeProduction :: = TYPE NOTATION ":: =" AlternativeList**
**ValueProduction :: = VALUE NOTATION ":: =" AlternativeList**
**SupportingProductions :: = ProductionList | empty**

**ProductionList :: = Production | ProductionList Production**
**Production :: = identifier ":: =" AlternativeList**

**Alternative List :: = Alternative | AlternativeList "|" Alternative**
**Alternative :: = SymbolList**

**SymbolList :: = Symbol | SymbolList Symbol**
**Symbol :: = Terminal | Nonterminal | EmbeddedDefinitions**

As is partially revealed by the definition of **Symbol** given above, the BNF used within a macro definition differs somewhat from that described in § 2 (and used throughout this Recommendation). There are three differences, which are described below.

The first difference in the BNF used within macros is that *all* terminals (not just those that must be distinguished, for example, from BNF operators) are quoted:

**Terminal** :: = **"string"**

The second difference is that two additional non-terminals, **type** and **value**, are "built-in" and available for use within a macro definition. The non-terminal **type** is a data type. The non-terminal **value** is a data value of a particular type. As a side effect of either non-terminal's recognition, a reference name may (but need not) be assigned to the particular type or value that is found in the instance of the nonstandard notation being parsed. Both non-terminals are "parameterized" to indicate, in the case of **value**, the type of data value expected and, in either case, the reference name, if any, to be assigned:

**Nonterminal** :: = **ProductionName** |
**RegularBuiltinNonterminal** |
**SpecialBuiltinNonterminal**

**ProductionName** :: = **identifier** – – *of a supporting production*
**RegularBuiltinNonterminal** :: = **"string"** | **"identifier"** | **"number"** | **"empty"** – – *as in Section 2*
**SpecialBuiltinNonterminal** :: = **type** | – – *a type*
**type (identifier)** | – – *a type to which a name is assigned*
**value (Type)** | – – *a value of the specified type*
**value (identifier Type)** – – *a value to which a name is assigned*

The association between a reference name assigned by the **type** or **value** non-terminal and a particular data type or data value is made when the non-terminal is recognized. Reassigning the reference name later in the parse alters the association. The parse of the type notation is considered to immediately precede the parse of each instance of the value notation. Thus the associations that result from the former are in effect at the outset of the latter. All associations are strictly local to the macro definition and have no effect outside of it.

Exactly once during the parse of each instance of the non-standard value notation, the distinguished reference name VALUE must be assigned to a data value. *It is this value and its type that are implied by the non-standard notation defined by the macro.*

Other built-in non-terminals are for further study.

The third and final difference between the BNF described in § 2 and that under discussion here is that a collection of one or more type and value definitions may be embedded in the BNF at any point:

**EmbeddedDefinitions** :: = **<EmbeddedDefinitionList>**

**EmbeddedDefinitionList** :: = **EmbeddedDefinition** |
**EmbeddedDefinitionList EmbeddedDefinition**
**EmbeddedDefinition** :: = **TypeDefinition** | **ValueDefinition**

The association between a reference name and a particular type or value in this way takes effect at the point in the parse at which the definitions are encountered. No additional symbols are sought or recognized. In all other respects, such an association has precisely the same properties and effects as one made as a side effect of the recognition of the **type** or **value** non-terminal. The assignments denoted by such definitions can be thought of as the "behind the scenes" processing or bookkeeping required for the proper functioning or "execution" of the macro.

Example:

The macro defined and used below is designed to represent either a single attribute (for example, of a user) or a list of such attributes. In this example, the choice between the two forms must be made when the type is defined:

```
ATTRIBUTE MACRO      ::=
BEGIN
TYPE NOTATION        ::= "LIST" <Type ::= SEQUENCE OF IA5String> |
                         empty <Type ::= IA5String>
VALUE NOTATION       ::= value (VALUE Type)
END

recipient ATTRIBUTE LIST     ::= {"NAME = Ralph Smith", "AGE = 42"}
   or
recipient ATTRIBUTE          ::= "NAME = Ralph Smith"
```

Example:

The macro defined and used below is also designed to represent either a single attribute or a list of attributes. In this example, however, the choice between the two need not be made until the value is defined:

```
ATTRIBUTES MACRO    :: =
BEGIN
TYPE NOTATION       :: =  empty
VALUE NOTATION      :: =  value (VALUE SEQUENCE OF IA5String) | value (VALUE IA5String)
END

recipient ATTRIBUTES        :: =  {"NAME = Ralph Smith", "AGE = 42"}
    or
recipient ATTRIBUTES        :: =  "NAME = Ralph Smith"
```

Wherever a type or value specification is called by the standard notation, non-standard notation for a type or value may be supplied in its place. This aspect of the standard notation is shown explicitly in neither the body of this Recommendation nor in the notation summary of Annex B.

*Note* – A protocol specification that uses standard notation exclusively is more readily understood by someone familiar only with the present Recommendation than is a specification that uses non-standard notation. For this reason, non-standard notation should be defined and used only after careful deliberation.

### 4.4 *Module definition*

It is often convenient to group related type, value, and macro definitions, for example, those of a particular protocol specification. Collections – called *modules* – of zero or more type, value, and macro definitions are defined using the following notation. Defining a module assigns a reference name to it. By convention, each module name begins with a capital letter. The reference names of the types, values, and macros defined within a module must be distinct:

```
ModuleDefinition    :: =  identifier DEFINITIONS ":: =" BEGIN ModuleBody END

ModuleBody          :: =  DefinitionList | empty

DefinitionList      :: =  Definition | DefinitionList Definition
Definition          :: =  TypeDefinition | ValueDefinition | MacroDefinition
```

Example:

```
Color DEFINITIONS :: =
BEGIN
PrimaryColor :: = INTEGER {red(0), yellow(1), blue(2)}
defaultColor PrimaryColor :: = yellow
END
```

Within a module, wherever a type or value specification is called for by the standard notation, the reference name of a type, value, or macro defined in another or the same module, or the reference name of any of the universal types defined in § 6 of the present Recommendation, may be supplied in its place. If the type, value, or macro is defined in another module, its reference name is preceded by that of the module in which the type, value, or macro is defined; the two reference names are separated by a period (“.”). This aspect of the standard notation is not shown explicitly in the body of this Recommendation, but does appear in the notation summary of Annex B.

## 5 Built-in types

This Recommendation defines ten built-in types. A *built-in* type is one for which standard notation is provided:

```
Type    :: =  BooleanType | IntegerType | BitStringType | OctetStringType | NullType | SequenceType |
              SetType | TaggedType | ChoiceType | AnyType

Value   :: =  BooleanValue | IntegerValue | BitStringValue | OctetStringValue | NullValue | SequenceValue |
              SetValue | TaggedValue | ChoiceValue | AnyValue
```

The *Boolean* type models logical data. The *Integer* type models numeric data. The *Bit String* type models binary data. The *Octet String* type models textual and other data that is conveniently represented as sequences of octets. The *Null* type is a valueless placeholder. The *Sequence* type models ordered collections of data. The *Set* type models unordered collections of data. The *Tagged* type models data that is semantically, rather than just syntactically, tagged for identification. The *Choice* type models data whose type is chosen from a collection of types. The *Any* type models data whose type is unrestricted.

## 5.1 *Boolean*

A Boolean represents a logical quantity that can assume either of two values, *true* or *false.*

The notation for the Boolean type is the keyword **BOOLEAN**. The notation for a Boolean value is either of two keywords, **TRUE** or **FALSE**:

**BooleanType** :: = **BOOLEAN**

**BooleanValue** :: = **TRUE** | **FALSE**

The representation for a Boolean is an element whose class is universal, whose form is primitive, whose ID Code is 1, and whose Contents is a single octet that encodes the value of the Boolean. **FALSE** is encoded as all bits zero, **TRUE** as any other combination of bits.

Example:

If of type **BOOLEAN**, the value **TRUE** can be encoded as:

| Boolean | *Length* | *Contents* |
|---|---|---|
| $01_{16}$ | $01_{16}$ | $FF_{16}$ |

## 5.2 *Integer*

An Integer represents an integer.

The notation for an Integer type is the keyword **INTEGER**, optionally followed by distinguished values and the distinct reference names assigned to them. By convention, the reference names begin with small letters. The notation for an Integer value is a signed number or one of the assigned reference names:

**IntegerType** :: = **INTEGER** | **INTEGER** {**NamedNumberList**}
**IntegerValue** :: = **number** | – **number** | **identifier**

**NamedNumberList** :: = **NamedNumber** | **NamedNumberList, NamedNumber**
**NamedNumber** :: = **identifier (number)**

The representation for an Integer is an element whose class is universal, whose form is primitive, whose ID Code is 2, and whose Contents is the value of the Integer, encoded as a twos complement binary number whose MSB and LSB are bit 8 of the first octet and bit 1 of the last octet, respectively. The value of the integer shall be encoded in the fewest possible octets, that is, with no more than the first (most significant) 9 bits being all 0 or all 1.

Example:

If of type **INTEGER**, the value **4294967296** is encoded as:

| Integer | *Length* | *Contents* |
|---|---|---|
| $02_{16}$ | $05_{16}$ | $0100000000_{16}$ |

Example:

If of type **INTEGER** {**low(0), medium(1), high(2)**}, the value **medium** is encoded as:

| Integer | *Length* | *Contents* |
|---|---|---|
| $02_{16}$ | $01_{16}$ | $01_{16}$ |

## 5.3 *Bit String*

A Bit String represents an ordered set of zero or more bits.

The notation for a Bit String type is the keywords **BIT STRING**, optionally followed by the numbers of distinguished bits and the distinct reference names assigned to them. By convention, the reference names begin with small letters. The first bit is numbered zero. The notation for a Bit String value is a series of binary or

hexadecimal digits enclosed in apostrophes and followed by the capital letter "B" or "H", respectively; or the assigned reference names of the one bits. The highest numbered bit that is explicitly specified is the value's last bit. Where such a convention is appropriate, missing bits are assumed to be zero.

**BitStringType** :: = **BIT STRING | BIT STRING {NamedNumberList}**
**BitStringValue** :: = **'string' B | 'string' H | {IdentifierList}**

**NamedNumberList** :: = **NamedNumber | NamedNumberList , NamedNumber**
**NamedNumber** :: = **identifier (number)**

**IdentifierList** :: = **identifier | IdentifierList, identifier**

The representation for a Bit String is an element whose class is universal, whose form is either primitive or constructor, whose ID Code is 3, and whose Contents depends upon the form. If the form is primitive, the Contents is the bits of the Bit String, packed eight to an octet and preceded by a single octet that encodes the number of unused bits in the final octet of the Contents – from zero to seven – as an unsigned binary number whose MSB and LSB are bit 8 and bit 1, respectively. If the form is constructor, the Contents is an ordered set of zero or more Bit Strings in their standard representations – as if Bit String were defined as **[UNIVERSAL 3] IMPLICIT SEQUENCE OF BIT STRING**. Each component Bit String represents a *segment* – zero or more bits – of the overall Bit String. The number of bits in all segments but the last is a multiple of eight. Segment boundaries are insignificant, that is, they carry no information. The constructor form is typically used in conjunction with the indefinite form of Length for very long Bit Strings whose total length may not be readily available.

*Note* – A Bit String is ordered in the octets that form its representation such that bit 0 of the Bit String corresponds to bit 8 of the first octet, bit 7 to bit 1 of the first octet, bit 8 to bit 8 of the second octet, etc.

Example:

If of type **BIT STRING**, the value **'0A3B5F291CD'H** can be encoded as shown below. In this example, the Bit String is represented as a primitive:

| Bit String | *Length* | *Contents* |
|---|---|---|
| $03_{16}$ | $07_{16}$ | $040A3B5F291CD0_{16}$ |

Example:

The value shown above can also be encoded as shown below. In this example, the Bit String is represented as a constructor:

| Bit String | *Length* | *Contents* | | |
|---|---|---|---|---|
| $23_{16}$ | $80_{16}$ | | | |
| | | Bit String<br>$03_{16}$ | *Length*<br>$03_{16}$ | *Contents*<br>$000A3B_{16}$ |
| | | Bit String<br>$03_{16}$ | *Length*<br>$05_{16}$ | *Contents*<br>$045F291CD0_{16}$ |
| | | EOC<br>$00_{16}$ | *Length*<br>$00_{16}$ | |

Example:

If of type **BIT STRING {married(0), employed(1)}**, the value **{employed}** can be encoded as:

| Bit String | *Length* | *Contents* |
|---|---|---|
| $03_{16}$ | $02_{16}$ | $0640_{16}$ |

## 5.4 *Octet String*

An Octet String represents an ordered set of zero or more octets.

The notation for an Octet String type is the keywords **OCTET STRING**. The notation for an Octet String value takes either of two forms. The first form is a series of binary or hexadecimal digits enclosed in apostrophes and followed by the capital letter "B" or "H", respectively. The highest numbered octet that is explicitly specified at least partially is the value's last octet. Unspecified bits, if any, in the last octet are assumed to be zero. The second form is a series of characters enclosed in quotation marks. A quotation mark may appear in the series but must be doubled to distinguish it from the quotation mark that terminates the series. The interpretation of this second form is context-specific. Every use of it must be accompanied by a detailed specification of the characters that are allowed, their graphical depictions, and their representations as sequences of octets (see § 6.1, for example):

**OctetStringType** :: = **OCTET STRING**

**OctetStringValue** :: = **'string' B | 'string' H | "string"**

The representation for an Octet String is an element whose class is universal, whose form is either primitive or constructor, whose ID Code is 4, and whose Contents depends upon the form. If the form is primitive, the Contents is the octets of the Octet String. If the form is constructor, the Contents is an ordered set of zero or more Octet Strings in their standard representations – as if Octet String were defined as **[UNIVERSAL 4] IMPLICIT SEQUENCE OF OCTET STRING**. Each component Octet String represents a *segment* – zero or more octets – of the overall Octet String. Segment boundaries are insignificant, that is, they carry no information. The constructor form is typically used in conjunction with the indefinite form of Length for very long Octet Strings whose total length may not be readily available.

*Note* – The octets in an Octet String are numbered from 0 to n. An Octet String is ordered in the octets that form its representation such that octet 0 of the Octet String corresponds to the first octet and octet n to the last octet of the representation.

Example:

If of type **OCTET STRING**, the value **'48692054686572652E'H** can be encoded as shown below. In this example, the Octet String is represented as a primitive:

| Octet String | *Length* | *Contents* |
|---|---|---|
| $04_{16}$ | $09_{16}$ | $4869205468657265 2E_{16}$ |

Example:

The value shown above can also be encoded as shown below. In this example, the Octet String is represented as a constructor:

| Octet String | *Length* | *Contents* | | |
|---|---|---|---|---|
| $24_{16}$ | $80_{16}$ | | | |
| | | Octet String | *Length* | *Contents* |
| | | $04_{16}$ | $04_{16}$ | $48692054_{16}$ |
| | | Octet String | *Length* | *Contents* |
| | | $04_{16}$ | $05_{16}$ | $686572652E_{16}$ |
| | | EOC | *Length* | |
| | | $00_{16}$ | $00_{16}$ | |

## 5.5 *Null*

A Null represents a valueless placeholder.

The notation for the Null type and the Null value is the keyword **NULL**:

**NullType** :: = **NULL**

**NullValue** :: = **NULL**

The representation for a **NULL** is an element whose class is universal, whose form is primitive, whose ID Code is 5, and whose Contents is unused and absent.

Example:

If of type **NULL**, the value **NULL** is encoded as:

| NULL | *Length* |
|---|---|
| $05_{16}$ | $00_{16}$ |

Example:

If of type **SEQUENCE {name IA5String, age CHOICE {INTEGER, NULL}}**, the value **{name "Smith", age NULL}** can be encoded as:

| Sequence | *Length* | *Contents* | | |
|---|---|---|---|---|
| $30_{16}$ | $09_{16}$ | | | |
| | | IA5 String | *Length* | *Contents* |
| | | $16_{16}$ | $05_{16}$ | $536DG97468_{16}$ |
| | | Null | *Length* | |
| | | $05_{16}$ | $00_{16}$ | |

In the above, **NULL** is provided as an optional, valueless placeholder to be used in case no value for age is given.

## 5.6 *Sequence*

A Sequence represents an ordered set of zero or more values called its *elements.*

The notation for a Sequence type is the keyword **SEQUENCE**, followed by any constraints imposed upon the elements. Three forms of Sequence are allowed with different constraints.

A Sequence may have elements that are variable in number but of one type, in which case that type is specified.

Alternatively a Sequence may have elements that are fixed in number, and possibly of several (although not necessarily distinct) types, in which case those types and the distinct reference names optionally assigned to the elements are specified.

A Sequence of multiple types may have elements that are optional, that is, may be omitted at the discretion of the entity constructing the Sequence. In this case all elements of the Sequence, including optional elements, must be of distinct types. Default values may (but need not) be associated with optional elements. Optional elements are identified by means of the keyword **OPTIONAL**, if no default value is specified, or **DEFAULT**, otherwise.

The notation for a Sequence value is the reference names (if any) and values of its elements:

**SequenceType** ::= **SEQUENCE | SEQUENCE OF Type | SEQUENCE {ElementTypes}**

**SequenceValue** ::= **{ElementValues}**

**ElementTypes** ::= **OptionalTypeList | empty**
**OptionalTypeList** ::= **OptionalType | OptionalTypeList, OptionalType**
**OptionalType** ::= **NamedType | NamedType OPTIONAL | NamedType DEFAULT Value | ComponentsOf**
**NamedType** ::= **identifier Type | Type**

**ElementValues** ::= **NamedValueList | empty**
**NamedValueList** ::= **NamedValue | NamedValueList, NamedValue**
**NamedValue** ::= **identifier Value | Value**

*Note 1* – In the case of a Sequence of multiple types with optional elements, failure to assign a reference name to each member may render the value notation ambiguous: it may not be clear which member is being specified.

The construct **COMPONENTS OF** may be used within the definition of a Sequence to include the elements of another Sequence among those of the Sequence being defined. The notation is the keywords **COMPONENTS OF** followed by a Sequence type:

**ComponentsOf :: = COMPONENTS OF SequenceType**

The effect of **COMPONENTS OF** is to insert the members of its argument as members of the Sequence within whose definition it appears.

*Note 2* – In the case of a Sequence with optional elements, care must be taken when using this construct to ensure that the types of the elements of the incorporated Sequence do not conflict with those of the Sequence being defined.

Example: If:

**A :: = SEQUENCE {X, Y, Z}**

**B :: = SEQUENCE {T, U, COMPONENTS OF A}**

Then an exactly equivalent definition of **B** is:

**B :: = SEQUENCE {T, U, X, Y, Z}**

The representation for a Sequence is an element whose class is universal, whose form is constructor, whose ID Code is 16, and whose Contents is the elements of the Sequence, ordered and in their standard representations.

Example:

If of type **SEQUENCE OF IA5String**, the value {**"Smith", "Jones"**} can be encoded as:

| Sequence | *Length* | *Contents* | | |
|---|---|---|---|---|
| $30_{16}$ | $0E_{16}$ | | | |
| | | IA5 String | *Length* | *Contents* |
| | | $16_{16}$ | $05_{16}$ | $536D697468_{16}$ |
| | | IA5 String | *Length* | *Contents* |
| | | $16_{16}$ | $05_{16}$ | $4A6F6E6573_{16}$ |

Example:

If of type **SEQUENCE {name IA5String, ok BOOLEAN}**, the value {**name "Smith", ok TRUE**} can be encoded as:

| Sequence | *Length* | *Contents* | | |
|---|---|---|---|---|
| $30_{16}$ | $0A_{16}$ | | | |
| | | IA5 String | *Length* | *Contents* |
| | | $16_{16}$ | $05_{16}$ | $536D697468_{16}$ |
| | | Boolean | *Length* | *Contents* |
| | | $01_{16}$ | $01_{16}$ | $FF_{16}$ |

## 5.7 *Set*

A Set represents an unordered set of zero or more values called its *members.*

The notation for a Set type is the keyword **SET**, followed by any constraints imposed upon the members. Three forms of Set are allowed, with different constraints.

A Set may have members that are variable in number but of one type, in which case that type is specified.

Alternatively a Set may have members that are fixed in number and of distinct types, in which case those types and the distinct reference names optionally assigned to the members are specified.

A Set of multiple types may have members that are optional, that is, may be omitted at the discretion of the entity constructing the Set. In this case as well, all members of the Set, including optional members, must be of distinct types. Default values may (but need not) be associated with optional members. Optional members are identified by means of the keyword **OPTIONAL**, if no default value is specified, or **DEFAULT**, otherwise.

The notation for a Set value is the reference names (if any) and values of its members.

**SetType** :: = **SET | SET OF Type | SET {MemberTypes}**
**SetValue** :: = **{MemberValues}**

**MemberTypes** :: = **OptionalTypeList | empty**
**OptionalTypeList** :: = **OptionalType | OptionalTypeList, OptionalType**
**OptionalType** :: = **NamedType | NamedType OPTIONAL | NamedType DEFAULT Value | ComponentsOf**
**NamedType** :: = **identifier Type | Type**

**MemberValues** :: = **NamedValueList | empty**
**NamedValueList** :: = **NamedValue | NamedValueList, NamedValue**
**NamedValue** :: = **identifier Value | Value**

*Note 1* – In the case of a Set of multiple types, failure to assign a reference name to each member may render the value notation ambiguous: it may not be clear which member is being specified.

The construct **COMPONENTS OF** may be used within the definition of a Set to include the members of another Set among those of the Set being defined. The notation is the keywords **COMPONENTS OF** followed by a Set type:

**ComponentsOf :: = COMPONENTS OF SetType**

The effect of **COMPONENTS OF** is to insert the members of its argument as members of the Set within whose definition it appears.

*Note 2* – Care must be taken when using this construct to ensure that the types of the members of the incorporated Set do not conflict with those of the Set being defined.

Example: If:

**A :: = SET {X, Y, Z}**

**B :: = SET {T, U, COMPONENTS OF A}**

Then an exactly equivalent definition of **B** is:

**B :: = SET {T, U, X, Y, Z}**

The representation for a Set is an element whose class is universal, whose form is constructor, whose ID Code is 17, and whose Contents is the specified members of the Set, in any order but in their standard representations. The elements that represent the members must have distinct Identifiers. This requirement is commonly met by context-specifically tagging each Set member using the Tagged type (see § 5.8).

Example:

If of type **SET {name [0] IA5String, age [1] INTEGER OPTIONAL}**, the value **{name "Smith"}** can be encoded as:

| Set | *Length* | *Contents* | | | | |
|---|---|---|---|---|---|---|
| $31_{16}$ | $09_{16}$ | | | | | |
| | | Name | *Length* | *Contents* | | |
| | | $A0_{16}$ | $07_{16}$ | | | |
| | | | | IA5 String | *Length* | *Contents* |
| | | | | $16_{16}$ | $05_{16}$ | $536D697468_{16}$ |

## 5.8 *Tagged*

A Tagged represents a value that has been tagged for identification. The Tagged type provides a means for creating new types from types that already exist, and in such a way that the new types are distinguishable from the old. A type thus defined can be universal, application-wide, context-specific, or private-use (see § 3.1). The type of the value being tagged can be either *explicit* in the representation or *implicit.*

*Note 1* – A value of type Choice or Any may not be implicitly tagged.

*Note 2* – Every value is tagged in the sense that its type can be determined from the Identifier component of the element that represents it. An Integer is inherently distinguishable from an IA5 String, for example. The Tagged type allows a value to be tagged at a higher level to reflect its semantics and/or its syntactical constraints (for example, the allowed range of an Integer or the characters allowed in an IA5 String). Thus, for example, a user name and country name can be distinguished even if both are (tagged) IA5 Strings.

The notation for a Tagged type is a numeric tag followed by the type of the value being identified. The tag must be distinct from the tags – and, if the new type is universal, the ID Codes – assigned to other types of the same class. If the new type is universal application-wide, or private-use, the keyword **UNIVERSAL**, **APPLICATION** or **PRIVATE**, respectively, precedes the tag. Otherwise, the type is context-specific. If the type of the value being tagged is implicit, the keyword **IMPLICIT** precedes the type. Otherwise, the type is explicit. The notation for a Tagged value is that of the value being identified:

**TaggedType** :: = **Tag IMPLICIT Type | Tag Type**
**TaggedValue** :: = **Value**

**Tag** :: = **[Class number]**
**Class** :: = **UNIVERSAL | APPLICATION | PRIVATE | empty**

*Note* – The normal context for a context-specifically tagged data element is a Sequence, Set, or Choice.

The representation for a Tagged is an element whose class is that specified and whose ID Code is the tag. In the explicit case, the form of the element is constructor, and its Contents is the value being tagged, in its standard representation. In the implicit case, the form of the element is that of the value being tagged, and its Contents is the Contents (only) of the value being tagged. Implicitly tagging a value thus effectively alters its class and ID Code.

Example:

If user names are of type **[APPLICATION 3] IA5String**, the value **"Jones"** can be encoded as shown below. In this example, the IA5 String is explicit:

| UserName | *Length* | *Contents* | | |
|---|---|---|---|---|
| $63_{16}$ | $07_{16}$ | | | |
| | | IA5 String | *Length* | *Contents* |
| | | $16_{16}$ | $05_{16}$ | $4A6F6E6573_{16}$ |

Example:

If user names are of type **[APPLICATION 3] IMPLICIT IA5String**, the value **"Jones"** can be encoded as shown below. In this example, the IA5 String is implicit:

| UserName | *Length* | *Contents* |
|---|---|---|
| $43_{16}$ | $05_{16}$ | $4A6F6E6573_{16}$ |

## 5.9 *Choice*

A choice represents a value whose type is chosen from a set of one or more distinct *alternatives.*

The notation for a Choice type is the keyword **CHOICE**, followed by the types of the alternatives and the distinct reference names optionally assigned to them. By convention, the reference names begin with small letters. The notation for a Choice value is the reference name (if any) and value of the chosen alternative:

**ChoiceType** :: = **CHOICE {AlternativeTypeList}**
**ChoiceValue** :: = **identifierValue | Value**

**AlternativeTypeList** :: = **NamedType | AlternativeTypeList, NamedType**
**NamedType** :: = **identifier Type | Type**

*Note* – Failure to assign a reference name to each alternative may render the value notation ambiguous: it may not be clear which alternative is being specified.

A *bound* Choice represents a particular alternative. The notation for a bound Choice type is the reference name of that alternative, followed by the notation for the Choice type itself. The notation for a bound Choice value is that of the chosen alternative. Bound Choice types are commonly defined by referring (see § 4.1) to previously defined unbound Choice types:

**BoundChoiceType** :: = **identifier** < **ChoiceType**

**BoundChoiceValue** :: = **Value**

The representation for a Choice or bound Choice is that of the chosen alternative. The elements that represent the alternatives must have distinct identifiers. This requirement is commonly met by context-specifically tagging each Choice alternative using the Tagged type (see § 5.8).

Example:

If of type **CHOICE** {**name [0] IA5String, age [1] INTEGER**}, the value **name "Smith"** can be encoded as shown below. In this example, the choice is *unbound:*

| Name | *Length* | *Contents* | | |
|---|---|---|---|---|
| $A0_{16}$ | $07_{16}$ | | | |
| | | IA5 String | *Length* | *Contents* |
| | | $16_{16}$ | $05_{16}$ | $536D697468_{16}$ |

Example:

If of type **name** < **Attribute**, where **Attribute** is defined as **CHOICE** {**name [0] IA5String, age [1] INTEGER**}, the value **"Smith"** can be encoded as above. In this example, the Choice is bound.

## 5.10 *Any*

An Any represents a value whose type is unrestricted, that is, chosen from the set of all built-in and defined types.

The notation for the Any type is the keyword ANY. The notation for an Any value is the chosen type followed by a value of that type:

**AnyType** :: = **ANY**

**AnyValue** :: = **Type Value**

The representation for an Any is that of the chosen type.

Example:

If of type **ANY**, the value **IA5String "Smith"** can be encoded as:

| IA5 String | *Length* | *Contents* |
|---|---|---|
| $16_{16}$ | $05_{16}$ | $536D697468_{16}$ |

# 6 Defined types

This Recommendation defines seven defined types. A *defined* type is one specified using the standard notation described in § 4.1.

The *IA5 String* type models textual data conforming to International Alphabet No. 5. The *Numeric String* type models textual data enterable from devices such as telephone handsets. The *Printable String* type models textual data enterable from devices such as Telex terminals. The *T.61 String* type models textual data suitable for processing by Teletex terminals. The *T.100 String* type models textual and graphical data suitable for processing by Videotex terminals. The *Generalized Time* type models the date and local or UTC time and allows their values to be specified using a variety of units and precisions. The *UTC Time* type is a constrained form of Generalized Time which models the date and UTC time.

6.1 *IA5 String*

An IA5 String represents an ordered set of zero or more characters chosen from the International Reference Version of International Alphabet No. 5.

The IA5 String type is formally defined as shown below. The characters allowed and their graphical depictions and seven-bit numeric codes are those specified for the International Reference Version of IA5 by Recommendation T.50. Each octet contains a single code. Bit 8 of each octet is zero, and Bits 7-1 correspond to $b_7$ - $b_1$ of the code (using the T.50 bit numbering convention):

**IA5String :: = [UNIVERSAL 22] IMPLICIT OCTET STRING**

Example:

If of type **IA5String**, the value **"Hi There."** can be encoded as:

| IA5 String | *Length* | *Contents* |
|---|---|---|
| $16_{16}$ | $09_{16}$ | $4869205468657265 2E_{16}$ |

6.2 *Numeric String*

A Numeric String represents an ordered set of zero or more characters that collectively encode numeric information in textual form. It models data entered from such devices as telephone handsets.

The Numeric String type is formally defined as shown below. The characters allowed are those listed in Table 3/X.409:

**NumericString :: = [UNIVERSAL 18] IMPLICIT IA5String**

TABLE 3/X.409

**Characters allowed in numeric strings**

| Name | Graphic |
|---|---|
| Digits | 0, 1, ... 9 |
| Space | (space) |

Example:

If of type **NumericString**, the value **"234"**, which is the X.121-specified country code for the United Kingdom in decimal notation, can be encoded as:

| Numeric String | *Length* | *Contents* |
|---|---|---|
| $12_{16}$ | $03_{16}$ | $323334_{16}$ |

6.3 *Printable String*

A Printable String represents an ordered set of zero or more characters chosen from a subset of the printable characters. It models data entered from devices with a limited character repertoire (for example, Telex terminals).

The Printable String type is formally defined as shown below. The characters allowed are those listed in Table 4/X.409:

**PrintableString :: = [UNIVERSAL 19] IMPLICIT IA5String**

Example:

If of type **PrintableString**, the value **"United Kingdom"** can be encoded as:

| Printable String | *Length* | *Contents* |
|---|---|---|
| $13_{16}$ | $0E_{16}$ | $556E69746564204B696E67646F6D_{16}$ |

TABLE 4/X.409

**Characters allowed in printable strings**

| Name | Graphic |
|---|---|
| Capital letters | A, B, ... Z |
| Small letters | a, b, ... z |
| Digits | 0, 1, ... 9 |
| Space | (space) |
| Apostrophe | ' |
| Left parenthesis | ( |
| Right parenthesis | ) |
| Plus sign | + |
| Comma | , |
| Hyphen | - |
| Full stop | . |
| Solidus | / |
| Colon | : |
| Equal sign | = |
| Question mark | ? |

6.4 *T.61 String*

A T.61 String represents an ordered set of zero or more characters and presentation commands chosen from the set defined by Recommendation T.61. It models textual data suitable for processing by Teletex terminals.

The T.61 String type is formally defined as shown below. The characters and commands allowed and their graphical depictions and eight-bit numeric codes are those specified by Recommendation T.61. Each octet contains a single code. Diacritically marked characters are represented by a pair of codes:

**T.61String :: = [UNIVERSAL 20] IMPLICIT OCTET STRING**

Example:

If of type **T.61String**, the value **"España"** can be encoded as:

| T.61 String | *Length* | *Contents* |
|---|---|---|
| $14_{16}$ | $07_{16}$ | $45737061C46E61_{16}$ |

6.5 *Videotex String*

A Videotex String represents an ordered set of zero or more alphabetic characters, pictorial characters, pictorial drawing commands, display attribute commands, etc., chosen from the set defined by the Data Syntaxes of Recommendation T.101 or the options from Recommendation T.100. It models textual and graphical data suitable for processing by Videotex terminals.

The Videotex String type is formally defined as shown below. The characters and commands allowed and their graphical depictions and seven-bit numeric codes are those specified by Recommendations T.100 and T.101. Each octet contains a single numeric code. Bit 8 of each octet is zero, and Bits 7-1 correspond to $b_7$ - $b_1$ of the code:

**VideotexString :: = [UNIVERSAL 21] IMPLICIT OCTET STRING**

Example:

If of type **VideotexString**, the value **"'double height' HIGH"** can be encoded as:

| Videotex String | *Length* | *Contents* |
|---|---|---|
| $15_{16}$ | $06_{16}$ | $1B4D48494748_{16}$ |

## 6.6 *Generalized Time*

A Generalized Time represents a calendar date and time of day to various precisions, as provided for by ISO 2014, ISO 3307, and ISO 4031. The time of day can be specified as local time only, UTC time only (see Recommendation B.11), or as both local and UTC time.

The Generalized Time type is formally defined as shown below. It is a string of characters, as follows:

a) Where the local time only is present, the Generalized Time is a string consisting of the date, as specified in ISO 2014, followed by the local time of day, using one of the forms specified in ISO 3307.

b) Where the UTC time only is present, the representation is as for Case a), followed by the letter "Z" to indicate that the time is based on UTC.

c) Where both local time and UTC are present, the representation is as for Case a), followed by a TDF (Time Differential Factor), as defined in ISO 4031, which represents the difference of local time for UTC.

The characters required to represent the Generalized Time (the digits "0" to "9", ".", ",", "+", "−", and "Z") are taken from International Alphabet Number 5:

**GeneralizedTime :: = [UNIVERSAL 24] IMPLICIT IA5String**

Example:

If of type **GeneralizedTime**, the value **"8201020700"**, which represents a local time of 7AM on 2 January 1982, can be encoded as:

| Generalized Time | *Length* | *Contents* |
|---|---|---|
| $18_{16}$ | $08_{16}$ | $3832303130323037303_{16}$ |

## 6.7 *UTC Time*

The UTC Time type is a particular form of Generalized Time which is defined for use in international applications where the local time only is not adequate, and where the flexibility to use all of the possible forms of ISO 3307 and ISO 2014 is not required. The UTC Time type permits the time of day to be specified to a precision of one minute or one second.

The UTC Time type is formally defined as shown below. It is a numeric string of either ten (*YYMMDDhhmm*) or twelve (*YYMMDDhhmmss*) digits, followed by either the letter "Z" or an offset of the form "+ *hhmm*" or "− *hhmm*". (*YY* is the two low-order digits of the Christian era year and *MM*, *DD*, *hh*, *mm*, and *ss* are the month, day, hour, minute, and second, respectively):

**UTCTime :: = [UNIVERSAL 23] IMPLICIT GeneralizedTime**

Example:

If of type **UTCTime**, the value **"8201021200Z"**, which represents a UTC time of noon on 2 January 1982, can be encoded as:

| UTC Time | *Length* | *Contents* |
|---|---|---|
| $17_{16}$ | $0B_{16}$ | $383230313032313230305A_{16}$ |

Example:

If of type **UTCTime**, the value **"8201020700-0500"**, which represents a local time of 7AM on 2 January 1982 in New York City, can be encoded as:

| UTC Time | *Length* | *Contents* |
|---|---|---|
| $17_{16}$ | $0E_{16}$ | $383230313032303730302D30353030_{16}$ |

ANNEX A

(to Recommendation X.409)

**Abbreviations**

The following abbreviations are used in this Recommendation.

| | |
|---|---|
| BNF | Backus-Naur Form |
| EOC | End-of-contents |
| LSB | Least significant bit |
| MSB | Most significant bit |
| OSI | Open Systems Interconnection |
| TDF | Time differential factor |
| UTC | Coordinated Universal Time |

ANNEX B

(to Recommendation X.409)

**Notation summary**

The notation defined in this Recommendation is summarized below in compressed and reorganized form.

**TypeDefinition** :: = **identifier "::=" Type**

**Type** :: = **BuiltinType | DefinedType**
**BuiltinType** :: = **BOOLEAN | INTEGER | INTEGER {NamedNumberList} | BIT STRING | BIT STRING {NamedNumberList} | OCTET STRING | NULL | SEQUENCE | SEQUENCE OF Type | SEQUENCE {ElementTypes} | SET | SET OF Type | SET {MemberTypes} | Tag IMPLICIT Type | Tag Type | CHOICE {AlternativeTypeList} | identifier < Type | ANY**
**DefinedType** :: = **identifier | identifier.identifier**

**NamedNumberList** :: = **NamedNumber | NamedNumberList, NamedNumber**
**NamedNumber** :: = **identifier (NumericValue)**
**NumericValue** :: = **number | DefinedValue**

**ElementTypes** :: = **OptionalTypeList | empty**
**MemberTypes** :: = **OptionalTypeList | empty**
**OptionalTypeList** :: = **OptionalType | OptionalTypeList, OptionalType**
**OptionalType** :: = **NamedType | NamedType OPTIONAL | NamedType DEFAULT Value | ComponentsOf**
**NamedType** :: = **identifier Type | Type**
**ComponentsOf** :: = **COMPONENTS OF Type** — *either Sequence or Set*
**AlternativeTypeList** :: = **NamedType | AlternativeTypeList, NamedType**

**Tag** :: = **[Class NumericValue]**
**Class** :: = **UNIVERSAL | APPLICATION | PRIVATE | empty**

**ValueDefinition** :: = **identifier Type "::=" Value**

**Value** :: = **BuiltinValue | DefinedValue**
**BuiltinValue** :: = **TRUE | FALSE | number | – number | identifier | 'string' B | 'string' H | {IdentifierList} | "string" | NULL | {Values} | identifier Value | Type Value**

**DefinedValue** :: = **identifier | identifier.identifier**

**IdentifierList** :: = **identifier | IdentifierList, identifier**

**Values** :: = **NamedValueList | empty**
**NamedValueList** :: = **NamedValue | NamedValueList, NamedValue**
**NamedValue** :: = **identifier Value | Value**

**MacroDefinition** :: = **identifier MACRO "::=" BEGIN MacroBody END**

**MacroBody** :: = **TypeProduction ValueProduction SupportingProductions**
**TypeProduction** :: = **TYPE NOTATION "::=" AlternativeList**
**ValueProduction** :: = **VALUE NOTATION "::=" AlternativeList**
**SupportingProductions** :: = **ProductionList | empty**

**ProductionList** :: = **Production | ProductionList Production**
**Production** :: = **identifier "::=" AlternativeList**

**AlternativeList** :: = **Alternative | AlternativeList "|" Alternative**
**Alternative** :: = **SymbolList**

**SymbolList** :: = **Symbol | SymbolList Symbol**
**Symbol** :: = **Terminal | Nonterminal | EmbeddedDefinitions**

**Terminal** :: = **"string"**
**Nonterminal** :: = **identifier | "string" | "identifier" | "number" | "empty" | type | type (identifier) | value (Type) | value (identifier Type)**
**EmbeddedDefinitions** :: = **<EmbeddedDefinitionList>**

**EmbeddedDefinitionList** :: = **EmbeddedDefinition | EmbeddedDefinitionList EmbeddedDefinition**
**EmbeddedDefinition** :: = **TypeDefinition | ValueDefinition**

**ModuleDefinition** :: = **identifier DEFINITIONS "::=" BEGIN ModuleBody END**

**ModuleBody** :: = **DefinitionList | empty**
**DefinitionList** :: = **Definition | DefinitionList Definition**

**Definition** :: = **TypeDefinition | ValueDefinition | MacroDefinition**

Comments may be embedded in the notation. They are preceded by two hyphens ("– –") and terminated by either two hyphens or the end of a line (see IA5).

ANNEX C

(to Recommendation X.409)

**ID Code summary**

The ID Codes assigned to the universal data types defined in this Recommendation are summarized in Table C-1/X.409.

TABLE C-1/X.409

**Assignment of universal ID Codes**

| ID Code | Data type |
|---|---|
| 0 | End-of-contents [a] |
| 1 | Boolean |
| 2 | Integer |
| 3 | Bit String |
| 4 | Octet String |
| 5 | Null |
| 6-15 | *Unassigned* |
| 16 | Sequence |
| 17 | Set |
| 18 | Numeric String |
| 19 | Printable String |
| 20 | T.61 String |
| 21 | Videotex String |
| 22 | IA5 String |
| 23 | UTC Time |
| 24 | Generalized Time |

[a] Although not a data type, the end-of-contents data element (see § 3.2) consumes an ID Code.

APPENDIX I

(to Recommendation X.409)

**List of Terms**

The following terms are contained in this Recommendation. References to definitions, implicit or explicit, are included where appropriate and available.

## A

**Alternative**
cont. in X.409; impl. def. in X.409

**Application-wide**
cont. in X.409; impl. def. in X.409

## B

**Bit String**
cont. in X.409; impl. def. in X.409

**Boolean**
cont. in X.409; impl. def. in X.409

**Bound**
cont. in X.409; impl. def. in X.409

## C

**Choice**
cont. in X.409; impl. def. in X.409

**Class**
cont. in X.409; impl. def. in X.409

**Constructed**
cont. in X.409; impl. def. in X.409

**Constructor**
cont. in X.409; impl. def. in X.409

**Contents**
cont. in X.409; impl. def. in X.409

**Context-specific**
cont. in X.409; impl. def. in X.409

**Coordinated Universal Time**
cont. in X.409

## D

**Data Element**
cont. in X.409; impl. def. in X.409

**Data Type**
cont. in X.409; impl. def. in X.409

**Data Value**
cont. in X.409; impl. def. in X.409

**Defined Types**
cont. in X.409; impl. def. in X.409

## E

**Element**
cont. in X.409; impl. def. in X.409

**End-of-contents**
cont. in X.409; impl. def. in X.409

**Explicit**
cont. in X.409; impl. def. in X.409

**Extension (octets)**
cont. in X.409; impl. def. in X.409

## F

**False**
cont. in X.409; impl. def. in X.409

**Form**
cont. in X.409; impl. def. in X.409

## G

**Generalized Time**
cont. in X.409; impl. def. in X.409

## I

**Id Code**
cont. in X.409; impl. def. in X.409

**IA5 String**
cont. in X.409; impl. def. in X.409

**Identifier**
cont. in X.409; impl. def. in X.409

**Implicit**
cont. in X.409; impl. def. in X.409

**Indefinite (form of Length)**
cont. in X.409; impl. def. in X.409

**Integer**
cont. in X.409; impl. def. in X.409

## L

**Least Significant Bit**
cont. in X.409

**Length**
cont. in X.409; impl. def. in X.409

**Long (form of Length)**
cont. in X.409; impl. def. in X.409

## M

**Macro**
cont. in X.409; impl. def. in X.409

**Member**
cont. in X.409; impl. def. in X.409

**Module**
cont. in X.409; impl. def. in X.409

**Most Significant Bit**
cont. in X.409

## N

**Non-standard Notation**
cont. in X.409; impl. def. in X.409

**Notation**
cont. in X.409; impl. def. in X.409

**Null**
cont. in X.409; impl. def. in X.409

**Numeric String**
cont. in X.409; impl. def. in X.409

## O

**Octet**
cont. in X.409; impl. def. in X.409

**Octet String**
cont. in X.409; impl. def. in X.409

## P

**Primitive**
cont. in X.409; impl. def. in X.409

**Printable String**
cont. in X.409; impl. def. in X.409

**Private-use**
cont. in X.409; impl. def. in X.409

## R

**Representation**
cont. in X.409; impl. def. in X.409

**Result**
cont. in X.409; impl. def. in X.409

## S

**Segment**
cont. in X.409; impl. def. in X.409

**Sequence**
cont. in X.409; impl. def. in X.409

**Set**
cont. in X.409; impl. def. in X.409

**Short (form of Length)**
cont. in X.409; impl. def. in X.409

**Standard Notation**
cont. in X.409; impl. def. in X.409

**Standard Representation**
cont. in X.409; impl. def. in X.409

**String**
cont. in X.409; impl. def. in X.409

## T

**Tagged**
cont. in X.409; impl. def. in X.409

**Time Differential Factor**
cont. in X.409

**True**
cont. in X.409; impl. def. in X.409

**Type**
cont. in X.409; impl. def. in X.409

**T.61 String**
cont. in X.409; impl. def. in X.409

## U

**UTC Time**
cont. in X.409; impl. def. in X.409

**Unbound**
cont. in X.409; impl. def. in X.409

**Universal**
cont. in X.409; impl. def. in X.409

## V

**Value**
cont. in X.409; impl. def. in X.409

**Videotex String**
cont. in X.409; impl. def. in X.409

APPENDIX II

(to Recommendation X.409)

**Example**

The use of the standard notation and representation defined in this Recommendation is illustrated below by means of a simple, hypothetical personnel record.

II.1 *Informal description of personnel record*

The structure of the personnel record and its value for a particular individual are shown below.

| | |
|---|---|
| Name: | **John P. Smith** |
| Title: | **Director** |
| Employee Number: | **51** |
| Date of Hire: | **17 September 1971** |
| Name of Spouse: | **Mary T. Smith** |
| Number of Children: | **2** |
| Child Information | |
| Name: | **Ralph T. Smith** |
| Date of Birth: | **11 November 1957** |
| Child Information | |
| Name: | **Susan B. Jones** |
| Date of Birth: | **17 July 1959** |

II.2 *Formal description of record structure*

The structure of every personnel record is formally described below using the standard notation for data types.

```
PersonnelRecord :: = [APPLICATION 0] IMPLICIT SET {
    Name,
    title [0] IA5String,
    EmployeeNumber,
    dateOfHire [1] Date,
    nameOfSpouse [2] Name,
    [3] IMPLICIT SEQUENCE OF ChildInformation DEFAULT {}}

ChildInformation :: = SET {
    Name,
    dateOfBirth [0] Date}

Name :: = [APPLICATION 1] IMPLICIT SEQUENCE {
    givenName IA5String,
    initial IA5String,
    familyName IA5String}

EmployeeNumber :: = [APPLICATION 2] IMPLICIT INTEGER

Date :: = [APPLICATION 3] IMPLICIT IA5String -- YYYYMMDD
```

II.3 *Formal description of record value*

The value of John Smith's personnel record is formally described below using the standard notation for data values.

```
{
  {givenName "John", initial "P", familyName "Smith"},
  title "Director",
  51,
  dateOfHire "19710917",
  nameOfSpouse {givenName "Mary", initial "T", familyName "Smith"},
  {
    {
      {givenName "Ralph", initial "T", familyName "Smith"},
      "19571111"},
    {
      {givenName "Susan", initial "B", familyName "Jones"},
      "19590717"}}}
```

## II.4 *Representation of record value*

The octet-level representation of the record value given above is shown below. The values of Identifiers, Lengths, and the Contents of Integers are shown in hexadecimal, two hexadecimal digits per octet. The values of the Contents of Octet Strings are shown as text, one character per octet.

| Personnel Record 60 | *Length* 8185 | *Contents* | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Name 61 | *Length* 10 | *Contents* | | | | | | |
| | | | | IA5 String 16 | *Length* 04 | *Contents* "John" | | | | |
| | | | | IA5 String 16 | *Length* 01 | *Contents* "P" | | | | |
| | | | | IA5 String 16 | *Length* 05 | *Contents* "Smith" | | | | |
| | | Title A0 | *Length* 0A | *Contents* | | | | | | |
| | | | | IA5 String 16 | *Length* 08 | *Contents* "Director" | | | | |
| | | Employee Number 42 | *Length* 01 | *Contents* 33 | | | | | | |
| | | Date of Hire A1 | *Length* 0A | *Contents* | | | | | | |
| | | | | Date 43 | *Length* 08 | *Contents* "19710917" | | | | |
| | | Name of spouse A2 | *Length* 12 | *Contents* | | | | | | |
| | | | | Name 61 | *Length* 10 | *Contents* | | | | |
| | | | | | | IA5 String 16 | *Length* 04 | *Contents* "Mary" | | |
| | | | | | | IA5 String 16 | *Length* 01 | *Contents* "T" | | |
| | | | | | | IA5 String 16 | *Length* 05 | *Contents* "Smith" | | |
| | | [3] A3 | *Length* 42 | *Contents* | | | | | | |
| | | | | Set 31 | *Length* IF | *Contents* | | | | |
| | | | | | | Name 61 | *Length* 11 | *Contents* | | |
| | | | | | | | | IA5 String 16 | *Length* 05 | *Contents* "Ralph" |
| | | | | | | | | IA5 String 16 | *Length* 01 | *Contents* "T" |
| | | | | | | | | IA5 String 16 | *Length* 05 | *Contents* "Smith" |
| | | | | | | Date of birth A0 | *Length* 0A | *Contents* | | |
| | | | | | | | | Date 43 | *Length* 08 | *Contents* "19571111" |
| | | | | Set 31 | *Length* IF | *Contents* | | | | |
| | | | | | | Name 61 | *Length* 11 | *Contents* | | |
| | | | | | | | | IA5 String 16 | *Length* 05 | *Contents* "Susan" |
| | | | | | | | | IA5 String 16 | *Length* 01 | *Contents* "B" |
| | | | | | | | | IA5 String 16 | *Length* 05 | *Contents* "Jones" |
| | | | | | | Date of Birth A0 | *Length* 0A | *Contents* | | |
| | | | | | | | | Date 43 | *Length* 08 | *Contents* "19590717" |

APPENDIX III

(to Recommendation X.409)

**Guidelines for use of the notation**

The data types and formal notation defined by this Recommendation are flexible, allowing a wide range of protocols to be designed using them. This flexibility, however, can sometimes lead to confusion, especially when the notation is approached for the first time. This document attempts to minimize confusion by giving guidelines for, and examples of, the use of the notation. For each of the ten *built-in* data types, one or more usage guidelines are offered. The *defined* data types (for example, IA5 String) specified by the Recommendation are not dealt with here.

III.1 *Boolean*

BO1 Use a Boolean to model the value of a logical (that is, two-state) variable, for example, the answer to a yes-or-no question.

Example: **Employed :: = BOOLEAN**

BO2 When assigning a reference name to a Boolean, choose one that describes the *true* state.

Example: **Married :: = BOOLEAN**
*not*
**MaritalStatus :: = BOOLEAN**

Also see Guideline IN4.

III.2 *Integer*

IN1 Use an Integer to model the value – for all practical purposes, unlimited in magnitude – of a cardinal or integer variable.

Example: **CheckingAccountBalance :: = INTEGER** *– – in cents; negative means overdrawn*

IN2 Define the minimum and maximum allowed values of an Integer as distinguished values.

Example: **DayOfTheMonth :: = INTEGER {first(1), last(31)}**

IN3 Use an Integer with distinguished values to model the value of a variable with three or more states. Assign values starting with zero if their only constraint is distinctness.

Example: **DayOfTheWeek :: = INTEGER {sunday(0), monday(1), tuesday(2), wednesday(3), thursday(4), friday(5), saturday(6)}**

IN4 Use an Integer with distinguished values to model the value of a variable that has just two states now but that may have additional states in a future version of the protocol.

Example: **MaritalStatus :: = INTEGER {single(0), married(1)}**
*in anticipation of*
**MaritalStatus :: = INTEGER {single(0), married(1), widowed(2)}**

III.3 *Bit String*

BI1 Use a Bit String to model binary data whose format and length are unspecified, or specified in another Recommendation, and whose length in bits is not necessarily a multiple of eight.

Example: **G3FacsimilePage :: = BIT STRING**
*– – a sequence of bits conforming to Recommendation T.4*

BI2 Define the first and last meaningful bits of a fixed-length Bit String as distinguished bits.

Example: **Nibble :: = BIT STRING {first(0), last(3)}**

BI3 Use a Bit String to model the value of a *bit map*, an ordered collection of logical variables indicating whether a particular condition holds for each of a correspondingly ordered collection of objects.

Example: **SunnyDaysOfTheMonth :: = BIT STRING {first(1), last(31)}**
*— — Day* i *was sunny if and only if Bit* i *is one*

BI4 Use a Bit String with distinguished values to model the values of a collection of *related* logical variables.

Example: **PersonalStatus :: = BIT STRING {married(0), employed(1), veteran(2), collegeGraduate(3)}**

III.4 *Octet String*

OC1 Use an Octet String to model binary data whose format and length are unspecified, or specified in another Recommendation, and whose length in bits is a multiple of eight.

Example: **G4FacsimileImage :: = OCTET STRING**
*— — a sequence of octets conforming to Recommendations T.5 and T.6*

OC2 Use a *defined* string type in preference to Octet String, where an appropriate one is available.

Example: **Surname :: = PrintableString**

OC3 Use an Octet String to model any string of information which cannot be modelled using one of the *defined* string types. Be sure to specify the repertoire of characters and their numeric codes.

Example: **PackedBCDString :: = OCTET STRING**
*— — the digits 0 through 9, two digits per octet, $1111_2$ used for padding*

III.5 *Null*

NU1 Use a Null to indicate the effective absence of an element of a Sequence.

Example:
```
PatientIdentifier :: = SEQUENCE {
  name IA5String,
  roomNumber CHOICE {INTEGER, NULL — — if an out-patient — —}}
```

III.6 *Sequence*

SQ1 Use a Sequence to model a collection of variables whose types are the same, whose number is large or unpredictable, and whose order is significant.

Example: **NamesOfMemberNations :: = SEQUENCE OF IA5String**
*— — in the order in which they joined*

SQ2 Use a Sequence to model a collection of variables whose types are the same, whose number is known and modest, and whose order is significant, provided the makeup of the collection is unlikely to change from one version of the protocol to the next.

Example:
```
NamesOfOfficers :: = SEQUENCE {
  president IA5String,
  vicePresident IA5String,
  secretary IA5String}
```

SQ3 Use a Sequence to model a collection of variables whose types differ, whose number is known and modest, and whose order is significant, provided the makeup of the collection is unlikely to change from one version of the protocol to the next.

Example:
```
Credentials :: = SEQUENCE {
  userName IA5String,
  password IA5String,
  accountNumber INTEGER}
```

SQ4 If the elements of a Sequence are fixed in number but of several types, a reference name should be assigned to every element whose purpose is not fully evident from its type.

Example:
```
File :: = SEQUENCE {
  ContentType,
  other FileAttributes,
  content ANY}
```

Also see Guidelines B13, B14, and NU1.

III.7 *Set*

ST1 Use a Set to model a collection of variables whose number is known and modest and whose order is insignificant. Identify each variable by context specifically tagging it.

Example:
```
UserName :: = SET {
    personalName [0] IMPLICIT IA5String,
    organizationName [1] IMPLICIT IA5String,
    countryName [2] IMPLICIT IA5String}
```

ST2 Use a Set to model a collection of variables that is a (proper or improper) subset of another collection of variables whose number is known and reasonably small and whose order is insignificant. Identify each variable by context-specifically tagging it.

Example:
```
UserName :: = SET {
    personalName [0] IMPLICIT IA5String,
    organizationName [1] IMPLICIT IA5String OPTIONAL,
      -- defaults to that of local organization
    countryName [2] IMPLICIT IA5String OPTIONAL
      -- defaults to that of local country --}
```

ST3 Use a Set to model a collection of variables whose makeup is likely to change from one version of the protocol to the next. Identify each variable by context-specifically tagging it.

Example:
```
UserName :: = SET {
    personalName [0] IMPLICIT IA5String,
    organizationName [1] IMPLICIT IA5String OPTIONAL,
      -- defaults to that of local organization
    countryName [2] IMPLICIT IA5String OPTIONAL
      -- defaults to that of local country
    -- other optional user attributes are for further study --}
```

ST4 If the members of a Set are fixed in number, a reference name should be assigned to every member whose purpose is not fully evident from its type.

Example:
```
FileAttributes :: = SET {
    owner [0] IMPLICIT UserName,
    sizeOfContentInOctets [1] IMPLICIT INTEGER,
    [2] IMPLICIT AccessControls, ...}
```

ST5 Use a Set to model a collection of variables whose types are the same and whose order is insignificant.

Example:
```
Keywords :: = SET OF IA5String  -- in arbitrary order
```

Also see Guidelines BI4 and TA3.

III.8 *Tagged*

TA1 Use a universal Tagged to define – *in this Recommendation only* – a generally useful, application-independent data type that must be distinguishable (by means of its representation) from all other data types.

Example:
```
EncryptionKey :: = [UNIVERSAL 24] IMPLICIT OCTET STRING  -- seven octets
```

TA2 Use an application-wide Tagged to define a data type that finds wide, scattered use within a particular application and that must be distinguishable (by means of its representation) from all other data types used in the application.

Example:
```
FileName :: = [APPLICATION 8] IMPLICIT SEQUENCE {
    directoryName IA5String, directoryRelativeFileName IA5String}
```

TA3 Use context-specific Taggeds to distinguish the members of a Set. Assign numeric tags starting with zero if their only constraint is distinctness.

Example:
```
CustomerRecord :: = SET {
    name [0] IMPLICIT IA5String,
    mailingAddress [1] IMPLICIT IA5String,
    accountNumber [2] IMPLICIT INTEGER,
    balanceDue [3] IMPLICIT INTEGER  -- in cents --}
```

TA4 Where a particular Set member has been application-wide tagged, a further context-specific Tagged need not be used, unless it is (or may be in the future) needed for distinctness. Where the Set member has been universally tagged, a further context-specific tag should be used.

Example:
```
ProductRecord :: = SET {
    UniformCode,
    description [0] IMPLICIT IA5String,
    inventoryNo [1] IMPLICIT INTEGER,
    inventoryLevel [2] IMPLICIT INTEGER}

UniformCode :: = [APPLICATION 13] IMPLICIT INTEGER
```

TA5 Use context-specific Taggeds to distinguish the alternatives of a Choice. Assign numeric tags starting with zero if their only constraint is distinctness.

Example:
```
CustomerAttribute :: = CHOICE {
    name [0] IMPLICIT IA5String,
    mailingAddress [1] IMPLICIT IA5String,
    accountNumber [2] IMPLICIT INTEGER,
    balanceDue [3] IMPLICIT INTEGER -- in cents --}
```

TA6 Where a particular Choice alternative has been defined using an application-wide Tagged, a further context-specific Tagged need not be used, unless it is (or maybe in the future) needed for distinctness.

Example:
```
ProductDesignator :: = CHOICE {
    UniformCode,
    description [0] IMPLICIT IA5String,
    inventoryNo [1] IMPLICIT INTEGER}

UniformCode :: = [APPLICATION 13] IMPLICIT INTEGER
```

TA7 Where a particular Choice alternative has been universally tagged, a further context-specific Tagged should be used, unless the provision of more than one universal type is the purpose of the choice.

Example:
```
CustomerIdentifier :: = CHOICE {
    name IA5String,
    number INTEGER}
```

TA8 Use a private-use Tagged to define a data type that finds use within a particular organization or country and that must be distinguishable (by means of its representation) from all other data types used by that organization or country.

Example:
```
AcmeBadgeNumber :: = [PRIVATE 2] IMPLICIT INTEGER
```

TA9 These guidelines use implicit tagging in the examples whenever it is legal to do so. This results in a compact representation which is highly desirable in some applications. In other applications, compactness may be less important than, for example, the ability to carry out strong type-checking. In the latter case, explicit tagging can be used.

Also see Guidelines ST1, ST2, CH1 and CH2.

III.9 *Choice*

CH1 Use a Choice to model a variable that is selected from a collection of variables whose number is known and modest. Identify each candidate variable by context-specifically tagging it.

Example:
```
FileIdentifier :: = CHOICE {
    relativeName [0] IMPLICIT IA5String,
        -- name of file (for example, "March Progress Report")
    absoluteName [1] IMPLICIT IA5String,
        -- names of file and containing directory
        -- (for example, "<Williams> March Progress Report")
    serialNumber [2] IMPLICIT INTEGER -- system-assigned identifier for file --}
```

CH2 Use a Choice to model a variable that is selected from a collection of variables whose makeup is likely to change from one version of the protocol to the next. Identify each candidate variable by context-specifically tagging it.

Example: **FileIdentifier :: = CHOICE** {
**relativeName [0] IMPLICIT IA5String,**
*– – name of file (for example, "March Progress Report")*
**absoluteName [1] IMPLICIT IA5String**
*– – names of file and containing directory*
*– – (for example, "<Williams> March Progress Report")*
*– – other forms of file identifier are for further study – –*}

CH3 A reference name should be assigned to each alternative whose purpose is not fully evident from its type.

Example: **FileIdentifier :: = CHOICE** {
**relativeName [0] IMPLICIT IA5String,**
*– – name of file (for example, "March Progress Report")*
**absoluteName [1] IMPLICIT IA5String,**
*– – names of file and containing directory*
*– – (for example, "<Williams> March Progress Report")*
**[2] IMPLICIT SerialNumber** *– – system-assigned identifier for file – –*}

CH4 Where implicit tagging is the norm in a particular application of this Recommendation, use a Choice of only one type where the possibility is envisaged of more than one type being permitted in the future. This precludes the possibility of implicit tagging taking place, and thus aids transition.

Example: **Greeting :: = [APPLICATION 12] CHOICE {IA5String}**
*in anticipation of*
**Greeting :: = [APPLICATION 12] CHOICE {IA5String, Voice}**

CH5 Use a bound Choice to model a variable whose type is that of a particular alternative of a previously defined, unbound Choice.

Example: **AbsoluteFileName :: = absoluteName < FileIdentifier** *– – see CH3*

Also see Guideline TA5.

III.10 *Any*

AN1 Use an Any to model a variable whose type is unspecified, or specified in another Recommendation.

Example: **ContentsOfFile :: = ANY**
*– – a data element whose type is specified in Recommendation X.Fictitious*

**Recommendation X.410**

## MESSAGE HANDLING SYSTEMS: REMOTE OPERATIONS AND RELIABLE TRANSFER SERVER

*(Malaga-Torremolinos, 1984)*

The establishment in various countries of telematic services and computer-based store-and-forward message services in association with public data networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT,

*considering*

(a) the need for interpersonal messaging and message transfer services;

(b) the need to transfer messages of different types having a large variety of formats;

(c) that Recommendation X.1 includes specific user classes of service and Recommendation X.10 categories of access; Recommendation X.2 defines optional user facilities for public data networks;

(d) that Recommendations X.20, X.20 *bis*, X.21, X.21 *bis*, X.22, X.25, X.28 and X.29 define DTE/DCE interfaces to public data networks;

(e) that Recommendations X.300 and X.121 permit international connections between public data networks;

(f) that Recommendation T.62 defines terminal equipment control procedures for Teletex and facsimile services;

(g) that the V-series of Recommendations provides the means for data communication over the telephone network;

(h) that Recommendation X.200 defines the reference model of open systems interconnection for CCITT applications;

(i) that a set of Recommendations describes various aspects of message handling: X.400, X.401, X.408, X.409, X.411, X.420 and X.430,

*unanimously declares*

(1) that general aspects of remote operations, the reliable transfer server, and use of the session service are described in § 1;

(2) that remote operations are described in § 2;

(3) that the reliable transfer server is described in § 3;

(4) that use of the presentation and session services is described in § 4;

(5) that lower layer protocols are described in § 5.

CONTENTS

## 1 Purpose and scope

This Recommendation is one of a series of Recommendations and describes the Remote Operations and Reliable Transfer Server for the Message Handling System (MHS). MHS service elements are provided by means of the Interpersonal Messaging (IPM) and Message Transfer (MT) Services.

Other Recommendations in the series contain additional information essential to a thorough understanding of the IPM and MT Services. Recommendation X.400 describes the MHS model and defines the IPM and MT service elements. Recommendation X.401 outlines its basic service elements and optional user facilities. Recommendation X.408 specifies the algorithms that MHS uses when converting between different types of encoded information. Recommendation X.409 defines the notation and the representational technique used to specify and encode MHS protocols. Recommendation X.411 specifies the protocol aspects of the MT Service. Recommendation X.420 specifies the protocol aspects of the IPM Service. Recommendation X.430 describes how Teletex terminals access the MHS.

This Recommendation defines Remote Operations, which are used to structure interactive Application Layer protocols such as the Submission and Delivery Protocol (P3, defined in Recommendation X.411), and describes the Reliable Transfer mechanism used between peer entities supporting the message handling protocols such as Message Transfer Protocol (P1, defined in Recommendation X.411) and P3.

§ 2 of this Recommendation describes the Remote Operations concept and specifies the notation and protocol data units required by interactive protocols, such as P3, which use it. § 3 defines the service primitives used to describe the reliable transfer, for example, of P1 and P3 protocol data units. § 4 describes how such message handling protocols as P1 and P3 make use of the OSI Presentation Service (Recommendation X.200), and Session Service (Recommendation X.215) which underlies it. § 5 specifies the lower layer protocols that are applicable to Message Handling Systems.

## 2 Remote operations

### 2.1 *Introduction*

Some Application Layer protocols are inherently interactive. The Submission and Delivery Protocol is one example. In P3, one entity – either the Submission and Delivery Entity or the Message Transfer Agent Entity – requests that a particular operation be performed. The other entity attempts to perform the operation and then reports the outcome of the attempt.

The specification and implementation of interactive protocols is facilitated by the concept of *remote operations* (or *operations* for short) and *remote errors* (or *errors* for short). This Recommendation uses the macro concept defined in Recommendation X.409 to define Operation and Error data types as a vehicle for concisely specifying interactive protocols. This section specifies the protocol data units that are exchanged to invoke an operation and report its outcome. These protocol data units thus provide a vehicle for uniformly implementing interactive protocols.

The protocol data units described in this Section are transferred by means of the Reliable Transfer Server (RTS) described in § 3. However, the details of the use of the RTS for this purpose are application protocol-specific. The entities governed by that application protocol are referred to here generically as *Application Entities (AEs).*

### 2.2 *Interpretation of Operation and Error data types*

An interactive protocol is specified using the Operation and Error data types. This section defines those types and previews the protocol data units specified in § 2.3. It also explains how the notational definitions of a particular Operation, and of the particular Errors it can report, determine the contents of the protocol data units used to invoke that operation and report its outcome.

#### 2.2.1 *Definitions of Operation and Error data types*

##### 2.2.1.1 *Operation data type*

A data value of type Operation represents the identifier for an operation that an AE in one open system may request be performed by a peer AE in another open system. A single data value, the argument of the

operation, accompanies the request. Some operations report their outcome, whether success or failure. Other operations report their outcome only if they succeed, others only if they fail, and still others never at all. A single data value, the result of the operation, accompanies a report of success; a report of failure identifies the exceptional condition that was encountered.

The notation for an Operation type is the keyword **OPERATION**, optionally followed by the type of the operation's argument, the reference name optionally assigned to it and the nature of the operation's outcome reporting (if any). If the operation reports success, the type of its result and the reference name optionally assigned to it are specified. If the operation reports failure, the reference names of the errors it reports are specified (see § 2.2.1.2). The notation for an Operation value is the operation's numeric identifier.

The macro which defines the Operation type is as follows:

```
OPERATION MACRO ::=
    BEGIN
    TYPE NOTATION       ::= "ARGUMENT" NamedType Result Errors | empty
    VALUE NOTATION      ::= value(VALUE INTEGER)
    Result              ::= empty | "RESULT" NamedType
    Errors              ::= empty | "ERRORS" {ErrorNames}

    NamedType           ::= identifier type | type
    ErrorNames          ::= empty | IdentifierList
    IdentifierList      ::= identifier | IdentifierList "," identifier

    END
```

From this definition, it can be observed that the representation for an Operation is that of an Integer.

*Example* – If the Operation that enumerates the files in a remote directory is of the following type:

```
OPERATION
    ARGUMENT SET {
        directoryName IA5String OPTIONAL,
            -- defaults to that specified via setDefaultDirectory
        sortKey INTEGER {fileName(0), modifyDate(1)} DEFAULT fileName}
    RESULT fileNames SEQUENCE OF IA5String
    ERRORS {noSuchDirectory, accessDenied}
```

and has the value **7**, it can be encoded as shown below. In this example, the operation always reports its outcome:

| *Integer* | *Length* | *Contents* |
|---|---|---|
| $02_{16}$ | $01_{16}$ | $07_{16}$ |

*Example* – If the Operation that sets the default directory for subsequent remote file operations is of the following type:

```
OPERATION
    ARGUMENT directoryName IA5String
```

and has the value **6**, it can be encoded as shown below. In this example, the operation never reports its outcome:

| *Integer* | *Length* | *Contents* |
|---|---|---|
| $02_{16}$ | $01_{16}$ | $06_{16}$ |

2.2.1.2 *Error data type*

A data value of type Error represents the identifier for an exceptional condition that an AE in one open system may report to a peer AE in another open system as the outcome of a previously requested operation. A single data value, the parameter of the error, accompanies the report.

The notation for an Error type is the keyword **ERROR**, optionally followed by the type of the error's parameter and the reference name optionally assigned to it. The notation for an Error value is the error's numeric identifier.

The macro which defines the Error type is as follows:

```
ERROR MACRO ::=
    BEGIN
    TYPE NOTATION       ::= "PARAMETER" NamedType | empty
    VALUE NOTATION      ::= value (VALUE INTEGER)
    NamedType           ::= identifier type | type
    END
```

From this description, it can be observed that the representation for an Error is that of an Integer.

*Example* – If the Error that signifies denial of access to a remote directory is of the following type:

**ERROR**
**PARAMETER accessGranted BIT STRING {none(0), read(1), add(2), modify(3)}**

and has the value **3**, it can be encoded as:

| *Integer* | *Length* | *Contents* |
|---|---|---|
| $02_{16}$ | $01_{16}$ | $03_{16}$ |

### 2.2.2 *Preview of protocol data units*

One AE invokes operations provided by another by entering into an exchange of Operation Protocol Data Units (OPDUs). The three principal OPDUs are briefly described below.

The Invoke OPDU requests that an operation be performed. It is sent whenever one AE desires assistance from another, and has three components. The invoke identifier component is used to correlate the request with its subsequent response. The operation component identifies the operation to be performed. The argument component is the operation's argument.

The ReturnResult OPDU reports the successful completion of an operation. It is sent in eventual response to an Invoke OPDU if the operation succeeds, and has two components. The invoke identifier component identifies the original request. The result component is the operation's result.

The ReturnError OPDU reports the unsuccessful completion of an operation. It is sent in eventual response to an Invoke OPDU if the operation fails and has three components. The invoke identifier component identifies the original request. The error component identifies the error being reported. The parameter component is the error's parameter.

### 2.2.3 *Relationship of data types to protocol data units*

The notational definition of a particular Operation determines the value of the operation component of the Invoke OPDU (when used to invoke that particular operation). It also determines the types of the argument component of the Invoke OPDU and the result component of the ReturnResult OPDU.

The notational definition of a particular Error determines the value of the error component of the ReturnError OPDU (when used to report that particular error). It also determines the type of the parameter component.

The above relationships are illustrated in Figures 1/X.410 and 2/X.410, which are based upon the examples of Operations and Errors given above. The first figure is indicative of all operations that report their outcome, whether success or failure. The second figure is indicative of all operations that never report their outcome. The situation that applies for operations that report either success or failure but not both is shown in neither of the figures, but is easily grasped once the other cases have been understood.

At the center of Figure 1/X.410 are shown the OPDUs previewed in § 2.2.2. (Figure 2/X.410 is effectively a portion of Figure 1/X.410 and is not discussed further.) At the extreme top and bottom of the figure are the formal descriptions of a hypothetical enumerate-files operation and one of the two errors it can report, access-denied. The formal descriptions can be viewed as shorthands for the informal descriptions shown nearer to the center of the figure. The informal description of enumerate-files has five parts: (1) the type of its argument, a Set; (2) the type of its result, a Sequence; (3) its op code, 7; (4) a statement of its outcome reporting behavior, the outcome is always reported; and (5) the reference names of the errors it reports, no-such-directory and access-denied. The informal description of access-denied has two parts: (1) the type of its parameter, a Bit String; and (2) its error code, 3. The circled numbers in the figure show which elements of the informal descriptions determine various components of the OPDUs. The invoke identifier component of all three OPDUs distinguishes one operation invocation from another and is independent of the particular operation or error involved.

*Formal description of operation:*

```
enumerateFiles OPERATION
   ARGUMENT SET {
      directoryName IA5String OPTIONAL,
      sortKey INTEGER {fileName(0), modifyDate(1)}
         DEFAULT fileName}
   RESULT fileNames SEQUENCE OF IA5String
   ERRORS {noSuchDirectory*, accessDenied}
      :: = 7
```

* This error is not depicted below, but is handled analogously to **accessDenied.**

*Equivalent informal description of operation:*

① ● **EnumerateFilesArgument :: = SET {**
**directoryName IA5String OPTIONAL,**
**sortKey INTEGER {fileName(0), modifyDate(1)}**
**DEFAULT fileName}**

② ● **EnumerateFilesResult :: = SEQUENCE OF IA5String**
*– – file names*

③ ● **enumerateFiles** op code: 7

● Both success and failure are reported.

● Errors reported: **noSuchDirectory** and **accessDenied.**

*Protocol data units for:*

- *invoking the operation*
- *reporting success*
- *reporting failure*

Invoke OPDU: ③ ①
**[1] SEQUENCE {invokeID INTEGER, OPERATION, argument ANY}**

ReturnResult OPDU: ②
**[2] SEQUENCE {invokeID INTEGER, result ANY}**

ReturnError OPDU: ⑤ ④
**[3] SEQUENCE {invokeID INTEGER, ERROR, parameter ANY}**

*Equivalent informal description of error:*

④ ● **AccessDeniedParameter :: =**
**BIT STRING {none(0), read(1), add(2), modify(3)}** *– – access granted*

⑤ ● **accessDenied** error code: 3

*Formal description of error:*

```
accessDenied ERROR
   PARAMETER accessGranted BIT STRING {
         none(0), read(1), add(2), modify(3)}
      :: = 3
```

FIGURE 1/X.410

**Relationship of Operation and Error data types to OPDUs (Outcome Reported)**

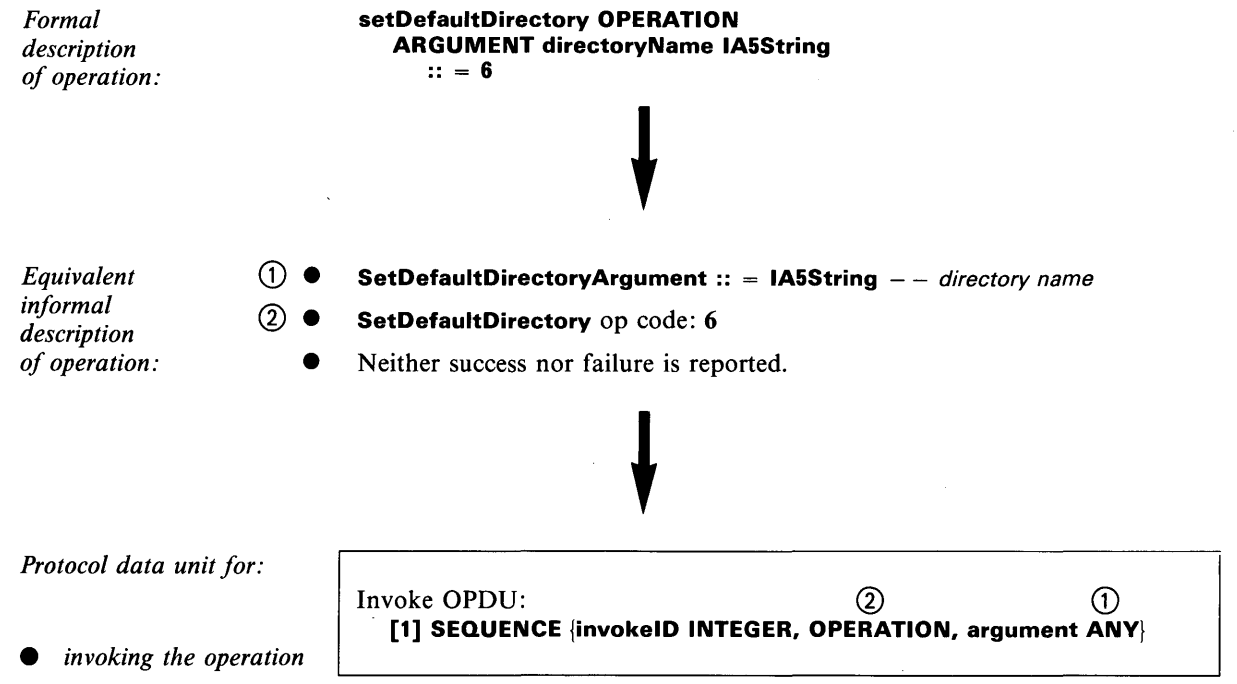



FIGURE 2/X.410

**Relationship of Operation data type to OPDUs (Outcome Not Reported)**

## 2.3 *Protocol data units*

One AE invokes operations provided by another by entering into an exchange of *Operation Protocol Data Units (OPDUs).* The OPDUs are transferred by means of the Reliable Transfer Server (see § 3), the details of whose use for this purpose are application protocol-specific. This section specifies the structure of each type of OPDU using the standard notation defined in Recommendation X.409. The binary representation of each OPDU type can be inferred from its notational description with the aid of that Recommendation.

As shown in Figure 3/X.410, four types of OPDU are defined:

**OPDU :: = CHOICE {[1] Invoke, [2] ReturnResult, [3] ReturnError, [4] Reject}**

The Invoke *OPDU* requests that an operation be performed. The ReturnResult OPDU reports the successful completion of an operation. The ReturnError OPDU reports the unsuccessful completion of an operation. The Reject OPDU reports the receipt and rejection of a malformed OPDU.

### 2.3.1 *Invoke OPDU*

The *Invoke OPDU* requests that an operation be performed. It is sent whenever one AE desires assistance from another. An AE need not wait for one operation to complete before invoking another. At any point in time, an AE may have any number of operations in progress at a particular remote AE (although the latter may reject an Invoke OPDU for lack of resources – see § 2.3.4):

**Invoke :: = SEQUENCE {invokeID INTEGER, OPERATION, argument ANY}**

The **invokeID** element specifies the *invoke identifier* assigned to the operation (invocation). It distinguishes the present operation from others the invoking AE may have in progress at the invoked AE. The invoking AE may begin to reuse invoke identifier values whenever it chooses (for example, after a fixed time interval), subject to the constraint that it may not reuse an identifier that was previously assigned to an invocation for which it expects but has not yet received a response. The invoked AE assumes that an Invoke OPDU bearing an identifier that violates this rule is a duplicate; therefore, it does not perform the requested operation. Instead, it rejects the duplicate Invoke OPDU as specified in § 2.3.4.

```
RemoteOperations DEFINITIONS ::=
BEGIN

-- macro definition for Operations

OPERATION MACRO ::=
BEGIN

TYPE NOTATION     ::= "ARGUMENT" NamedType Result Errors | empty
VALUE NOTATION    ::= value (VALUE INTEGER)

Result            ::= empty | "RESULT" NamedType
Errors            ::= empty | "ERRORS" "{"ErrorNames"}"

NamedType         ::= identifier type | type

ErrorNames        ::= empty | IdentifierList
IdentifierList    ::= identifier | IdentifierList "," identifier

END

-- macro definition for Errors

ERROR MACRO       ::=
BEGIN

TYPE NOTATION     ::= "PARAMETER" NamedType | empty
VALUE NOTATION    ::= value (VALUE INTEGER)

NamedType         ::= identifier type | type

END
```

FIGURE 3/X.410 (part 1 of 2)

**Formal definition of Remote Operations data types**

```
OPDU                    :: = CHOICE {[1] Invoke, [2] ReturnResult, [3] ReturnError,
                             [4] Reject}

-- OPDU types

Invoke                  :: = SEQUENCE {invokeID INTEGER, OPERATION, argument ANY}

ReturnResult            :: = SEQUENCE {invokeID INTEGER, result ANY}

ReturnError             :: = SEQUENCE {invokeID INTEGER, ERROR, parameter ANY}

Reject                  :: = SEQUENCE {
                                 invokeID CHOICE {INTEGER, NULL},
                                 problem CHOICE {
                                     [0] IMPLICIT GeneralProblem,
                                     [1] IMPLICIT InvokeProblem,
                                     [2] IMPLICIT ReturnResultProblem,
                                     [3] IMPLICIT ReturnErrorProblem}}

-- OPDU rejection reasons

GeneralProblem          :: = INTEGER {
                                 unrecognizedOPDU(0),
                                 mistypedOPDU(1),
                                 badlyStructuredOPDU(2)}

InvokeProblem           :: = INTEGER {
                                 duplicateInvocation(0),
                                 unrecognizedOperation(1),
                                 mistypedArgument(2)}

ReturnResultProblem     :: = INTEGER {
                                 unrecognizedInvocation(0),
                                 resultResponseUnexpected(1),
                                 mistypedResult(2)}

ReturnErrorProblem      :: = INTEGER {
                                 unrecognizedInvocation(0),
                                 errorResponseUnexpected(1),
                                 unrecognizedError(2),
                                 unexpectedError(3),
                                 mistypedParameter(4)}

END
```

FIGURE 3/X.410 (part 2 of 2)

**Formal definition of Remote Operations data types**

The **OPERATION** element specifies the operation to be performed, distinguishing it from other operations of which the invoked AE is capable. This value must agree with that specified in the AE's defining Recommendation.

The **argument** element specifies the operation's argument. Its type must be that specified in the AE's defining Recommendation.

### 2.3.2 *ReturnResult OPDU*

The *ReturnResult OPDU* reports the successful completion of an operation. It is sent in eventual response to an Invoke OPDU if the latter is well-formed, the operation is one that reports success only or both success and failure, and the operation succeeds:

```
ReturnResult :: = SEQUENCE {invokeID INTEGER, result ANY}
```

The **invokeID** element specifies the operation (invocation) whose success is being reported.

The **result** element specifies the operation's result. Its type must be that specified in the AE's defining Recommendation.

### 2.3.3 *ReturnError OPDU*

The *ReturnError OPDU* reports the unsuccessful completion of an operation. It is sent in eventual response to an Invoke OPDU if the latter is well-formed, the operation is one that reports failure only or both success and failure, and the operation fails:

```
ReturnError :: = SEQUENCE {invokeID INTEGER, ERROR, parameter ANY}
```

The **invokeID** element identifies the operation (invocation) whose failure is being reported.

The **ERROR** element specifies the error being reported, distinguishing it from other errors the invoked AE reports. Its value must agree with that specified in the AE's defining Recommendation, and must be one of those specified as reportable by the operation that was invoked.

The **parameter** element specifies the error's parameter. Its type must be that specified in the AE's defining Recommendation.

### 2.3.4 *Reject OPDU*

The *Reject OPDU* reports the receipt and rejection of a malformed OPDU. It is sent in eventual response to a malformed OPDU whose type is other than Reject:

```
Reject :: = SEQUENCE {
    invokeID CHOICE {INTEGER, NULL},
    problem CHOICE {
        [0] IMPLICIT GeneralProblem,
        [1] IMPLICIT InvokeProblem,
        [2] IMPLICIT ReturnResultProblem,
        [3] IMPLICIT ReturnErrorProblem}}
```

The **invokeID** element specifies the invoke identifier carried by the rejected OPDU. If none was present, the element is of type Null and has the value **NULL**.

The **problem** element specifies why the OPDU was rejected. The problems are categorized by OPDU type. The following general problems are reported:

```
GeneralProblem :: = INTEGER {
    unrecognizedOPDU(0),
    mistypedOPDU(1),
    badlyStructuredOPDU(2)}
```

The problem **unrecognizedOPDU** signifies that the type of the OPDU, as evidenced by its Identifier, is not one of the four defined by this Recommendation. The problem **mistypedOPDU** signifies that the structure of the OPDU does not conform to this Recommendation. The problem **badlyStructuredOPDU** signifies that the structure of the OPDU does not conform to Recommendation X.409.

The following Invoke OPDU-specific problems are reported:

```
InvokeProblem :: = INTEGER {
    duplicateInvocation(0),
    unrecognizedOperation(1),
    mistypedArgument(2)}
```

The problem **duplicateInvocation** signifies that the invoke identifier violates the assignment rule of § 2.3.1. The problem **unrecognizedOperation** signifies that the operation is not one of those specified in the AE's defining Recommendation. The problem **mistypedArgument** signifies that the type of the operation argument supplied is not that specified in the AE's defining Recommendation.

The following ReturnResult OPDU-specific problems are reported:

```
ReturnResultProblem :: = INTEGER {
    unrecognizedInvocation(0),
    resultResponseUnexpected(1),
    mistypedResult(2)}
```

The problem **unrecognizedInvocation** signifies that no operation with the specified invoke identifier is in progress. The problem **resultResponseUnexpected** signifies that the invoked operation does not report success. The problem **mistypedResult** signifies that the type of the operation result supplied is not that specified in the AE's defining Recommendation.

The following ReturnError OPDU-specific problems are reported:

```
ReturnErrorProblem :: = INTEGER {
    unrecognizedInvocation(0),
    errorResponseUnexpected(1),
    unrecognizedError(2),
    unexpectedError(3),
    mistypedParameter(4)}
```

The problem **unrecognizedInvocation** signifies that no operation with the specified invoke identifier is in progress. The problem **errorResponseUnexpected** signifies that the invoked operation does not report failure. The problem **unrecognizedError** signifies that the reported error is not one of those specified in the AE's defining Recommendation. The problem **unexpectedError** signifies that the reported error is not one that the invoked operation may report. The problem **mistypedParameter** signifies that the type of the error parameter supplied is not that specified in the AE's defining Recommendation.

### 2.4 *State diagrams*

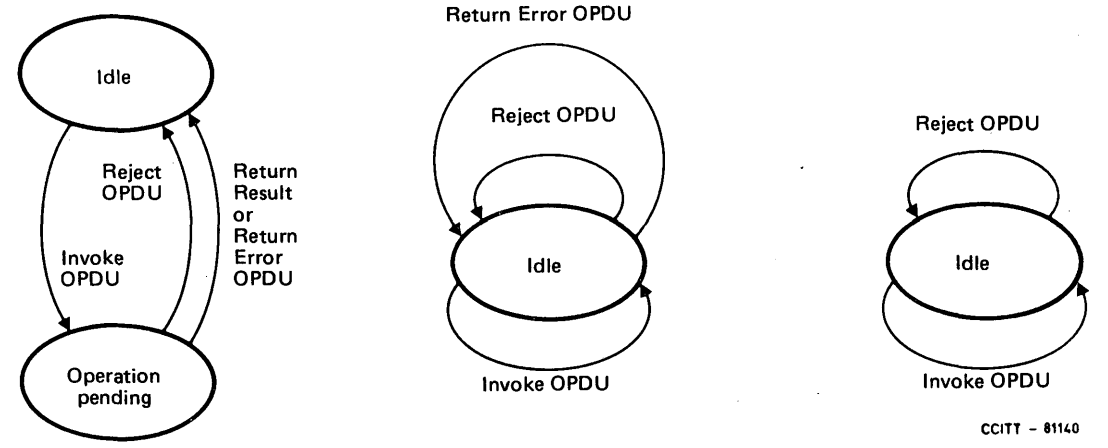
The behavior of an AE when invoking an operation or when performing an invoked operation is depicted in Figure 4/X.410. (A single state diagram suffices for both invoking and invoked AEs.) There are three cases to consider. Figure 4a/X.410 applies to an operation that reports its outcome, whether success or failure. Figure 4b/X.410 applies to an operation that reports failure but not success. Figure 4c/X.410 applies to an operation that reports neither success nor failure. Operations that report success but not failure are not considered here.

## 3 Reliable Transfer Server

### 3.1 *Introduction*

The *Reliable Transfer Server (RTS)* is that part of an Application Entity (AE) that is responsible for creating and maintaining associations between the AE and its peers, and for reliably transferring *Application Protocol Data Units (APDUs)* by means of them. The remainder of the AE, which drives the application protocol proper, is hereafter referred to as the *RTS-user.*

An association created by the RTS may be either monologue (APDUs transferred in one direction only), or two-way alternate. In the case of a two-way alternate association, the right to transfer APDUs is governed by the turn, which can be transferred from one AE to the other.




*a) Operations that report both success and failure*

*b) Operations that report failure but not success*

*c) Operations that report neither success nor failure*

FIGURE 4/X.410

**State diagrams for remote operations**

## 3.2 *Definition of primitives*

The interactions between the RTS and RTS-user are described as a set of *service primitives.* Service primitives are abstractions. They attempt to capture only those details of the interaction between entities that are aspects of the layer service itself. A service primitive neither specifies nor constrains the implementation of entities or the interfaces between them.

The service primitive descriptions that follow adhere to the conventions outlined in Annex C.

### 3.2.1 *OPEN: establishment of an association*

An RTS-user issues the *OPEN.Request* primitive to establish or open a new association with another RTS-user. The association may comprise several session connections in sequence; whenever a session connection fails, the RTS opens a new one.

Four events occur when an association is opened, as indicated in Figure 5/X.410.

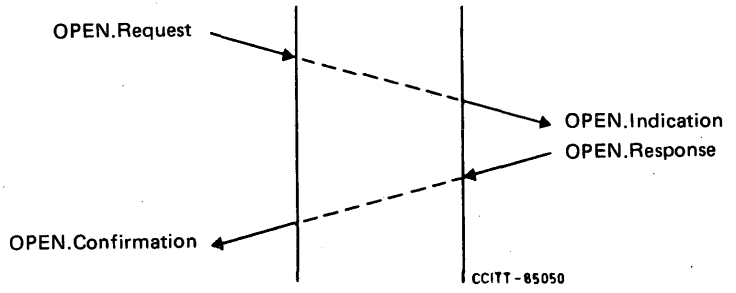



FIGURE 5/X.410

**OPEN primitive events**

The parameters of the OPEN.Request primitive are:

| Parameter name | Type | Parameter description | Notes |
|---|---|---|---|
| Responder-address | M | The session address of the RTS associated with the RTS-user to which the new association is to be opened | |
| Dialogue-mode | M | The type of association to be opened: *monologue* or *two-way alternate* | |
| Initial-turn | M | The RTS-user that is to have the turn initially: *initiator* or *responder* | |
| Application-protocol | M | Designates the application protocol that will govern communication over the association | 1 |
| User-data | C | User data associated with opening the association | |

*Note 1* – The values defined for this parameter include P1 and P3.

The parameters of the OPEN.Indication primitive are:

| Parameter name | Type | Parameter description | Notes |
|---|---|---|---|
| Initiator-address | M | The session address of the RTS associated with the RTS-user that issued the OPEN.Request primitive | |
| Dialogue-mode | M | The type of association being opened: *monologue* or *two-way alternate* | 2 |
| Initial-turn | M | The RTS-user that is to have the turn initially: *initiator* or *responder* | 2 |
| Application-protocol | M | Designates the application protocol that will govern communication over the association | 1, 2 |
| User-data | C | User data associated with opening the association | 2 |

*Note 1* – The values defined for this parameter include *P1* and *P3.*

*Note 2* – This parameter conveys the same value as in the OPEN.Request primitive.

The parameters of the OPEN.Response primitive are:

| Parameter name | Type | Parameter description | Notes |
|---|---|---|---|
| Disposition | M | The disposition of the request for a new association: *accepted* or *refused* | |
| User-data | C | User data associated with accepting the association | 1 |
| Refusal-reason | C | The reason for refusing the association | 2 |

*Note 1* – This parameter is present only if the association is accepted.

*Note 2* – This parameter is present only if the association is refused. Values defined are: unacceptable dialogue mode, authentication failure, busy.

The parameters of the OPEN.Confirmation primitive are identical to, and have the same values as, those of the OPEN.Response primitive.

### 3.2.2 *CLOSE: release of an association*

The initiating RTS-user issues the *CLOSE.Request* primitive to release or close the association. It may do so only if it possesses the turn.

Four events occur when an association is closed, as indicated in Figure 6/X.410.

None of the CLOSE primitives has any parameters.

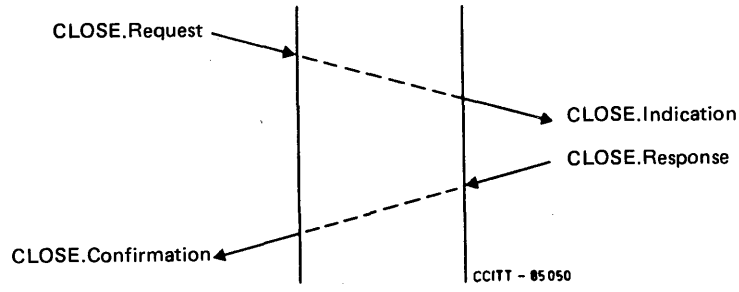



FIGURE 6/X.410

**CLOSE primitive events**

### 3.2.3 *TURN-PLEASE: request for exchange of the turn*

An RTS-user issues the *TURN-PLEASE.Request* primitive to request the turn. It may do so only if it does not already possess the turn. The turn is requested either to transfer data or release the association. The request conveys the priority of the action to be taken so that the other RTS-user can decide when to actually relinquish the turn.

Two events occur when exchange of the tokens is requested, as indicated in Figure 7/X.410.

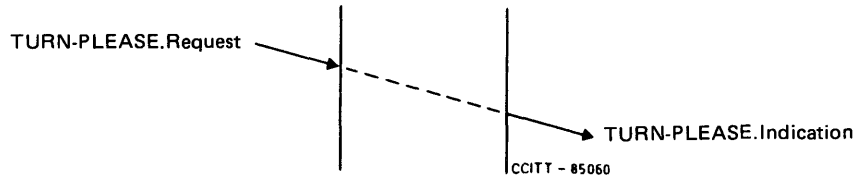



FIGURE 7/X.410

**TURN-PLEASE primitive events**

The parameters of the TURN-PLEASE.Request primitive are:

| Parameter name | Type | Parameter description | Notes |
|---|---|---|---|
| Priority | C | The priority of the action, governed by the turn, that the requesting RTS-user wishes to carry out | 1 |

*Note 1* – A priority is assigned to each RTS-user action. The actions of releasing an association and of transferring various APDUs will be assigned priorities. The range of valid priorities is a property of the application protocol in use.

The parameters of the TURN-PLEASE.Indication primitive are identical to, and have the same values as, those of the TURN-PLEASE.Request primitive.

### 3.2.4 *TURN-GIVE: exchange of the turn*

An RTS-user issues the *TURN-GIVE.Request* primitive to relinquish the turn to its peer. It may do so only if it possesses the turn.

Two events occur when the tokens are exchanged, as indicated in Figure 8/X.410.

Neither the TURN-GIVE.Request nor TURN-GIVE.Indication primitive has parameters.

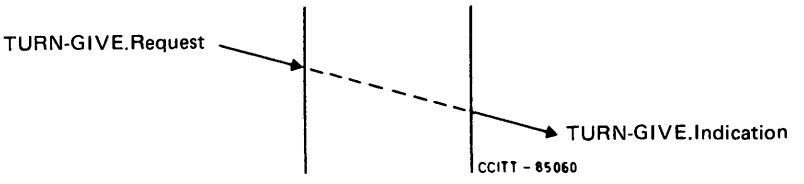



FIGURE 8/X.410

**TURN-GIVE primitive events**

### 3.2.5 *TRANSFER: reliable transfer of an application protocol data unit*

An RTS-user issues the *TRANSFER.Request* primitive to request the reliable transfer of an APDU over an association. It may do so only when the RTS is able to start sending data on an actual session connection.

Two events occur when an APDU is reliably transferred, as indicated in Figure 9/X.410.

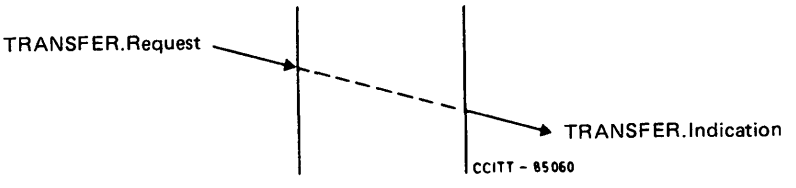



FIGURE 9/X.410

**TRANSFER primitive events**

The parameters of the TRANSFER.Request primitive are:

| Parameter name | Type | Parameter description |
|---|---|---|
| APDU | M | The APDU to be transferred |
| Transfer-time | M | The time period within which the RTS must successfully transfer the APDU to the other RTS-user |

The parameters of the TRANSFER.Indication primitive are:

| Parameter name | Type | Parameter description | Notes |
|---|---|---|---|
| APDU | M | The APDU being transferred | 1 |

*Note 1* – This parameter conveys the same value as in the TRANSFER Request primitive.

### 3.2.6 *EXCEPTION: indication of transfer failure*

The RTS issues the *EXCEPTION.Indication* primitive if it cannot complete the transfer of an APDU as requested.

One event occurs when a transfer failure is indicated, as shown in Figure 10/X.410.

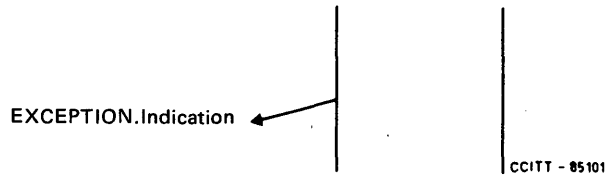



FIGURE 10/X.410

**EXCEPTION primitive event**

The parameters of the EXCEPTION.Indication primitive are:

| Parameter name | Type | Parameter description |
|---|---|---|
| APDU | M | The APDU that could not be transferred within the allotted time |

## 4 Use of the Presentation and Session Services

### 4.1 *Introduction*

This Section describes how the Reliable Transfer Server (RTS) is supported by the OSI Presentation and Session Services. The RTS has minimal requirements on the Presentation Layer itself. However, it makes extensive use of the Session Layer Services (Recommendation X.215) which are made directly available to Application Layer entities by the Presentation Layer. A minimal Presentation Layer protocol, which meets the needs of the RTS, is implicitly defined in § 4.2.1 below, and the remainder of this Section defines the use of the Session Layer Services by the RTS. The encoding of various parameters is defined using the notation of Recommendation X.409.

The subset of the Session Service used is composed of the following functional units. The corresponding session services are also listed:

| | |
|---|---|
| Kernel | Session Connection<br>Normal Data Transfer<br>Orderly Release<br>U-Abort<br>P-Abort |
| Exceptions | User Exception Reporting<br>Provider Exception Reporting |
| Activity Management | Activity Start<br>Activity Resume<br>Activity End<br>Activity Interrupt<br>Activity Discard<br>Please Tokens<br>Give Tokens<br>Give Control |
| Half-duplex | Give Tokens<br>Please Tokens |
| Minor Synchronize | Minor Synchronization Point<br>Give Tokens<br>Please Tokens |

Whenever the use of one of the corresponding Session service primitives is described, the layout shown in Figure 11/X.410 is employed.

*Note* – Bit alignment. Where referenced in these Recommendations, bit positions in octets are as defined in Recommendation X.409, § 3. Bit 8 as defined there corresponds to the high order bit of the Session Service, and bit 1 corresponds to the low order bit.

S-SERVICE_PRIMITIVE

| **Parameter** | **Req/Ind** | **Resp/Conf** |
|---|---|---|
| First parameter | Note 1 | |
| Second parameter | | Note 2 |
| Inapplicable parameter | – | – |

*Note 1* – Describes the setting of the first parameter in both the Request and Indication.

*Note 2* – Describes the setting of the second parameter in both the Response and Confirmation.

FIGURE 11/X.410

**Layout of Session Service primitive usage descriptions**

## 4.2 *Session Connection Establishment phase*

The RTS attempts Session Connection Establishment in response to either of the following:

1) OPEN: An OPEN.request issued by the RTS-user.

2) RECOVER: An internal rule when attempting to recover a session connection that was aborted.

The two cases are distinguished by means of data elements in the User-Data parameter.

An RTS may reject an incoming connect indication, for example, because of RTS congestion. Specific reasons are outlined in the connect response.

| Parameter | Req/Ind | Resp/Conf |
|---|---|---|
| Session Connection Identifier | Note 1 | Note 1 |
| Calling SSAP Address | Note 2 | |
| Called SSAP Address | Note 2 | Note 2 |
| Result | | Note 3 |
| Quality of Service (QOS) | Note 4 | Note 4 |
| Session Requirements | Note 5 | Note 5 |
| Synchronisation Point Serial Number | Note 6 | Note 6 |
| Initial Data Token Assignment | Note 7 | Note 7 |
| Initial Minor Synch Token Assignment | Note 7 | Note 7 |
| Initial Major/Act Token Assignment | Note 7 | Note 7 |
| SS-User Data | Note 8 | Note 8 |

*Note 1* – The initiating RTS will supply a Session Connection Identifier, which will be used to uniquely identify the connection. This identifier is formed of the following components: Calling SS-user Reference, Common Reference, and, optionally, Additional Reference Information. The identifier is returned unchanged by the responding RTS, except that the Calling SS-user Reference supplied by the initiator is conveyed as the Called SS-user Reference.

Each component, when present, will contain a data element of the appropriately named type from the following definitions:

**CallingSSUserReference** :: = **SSAPAddress** – – *of the initiator*
**CommonReference** :: = **UTCTime**
**AdditionalReferenceInformation** :: = **T61String**

The SSAP address is equivalent to the transport address (see Note 2) and is represented as a string of T.61 characters:

**SSAPAddress :: = T61String**

*Note 2* – Message handling will not use Session Layer addressing, that is, a session address will not be passed in the Connect SPDU of the Session Layer.

The SSAP Addresses are passed down to (and then passed up from) the Transport Layer. They have a one-to-one relationship with transport addresses.

*Note 3* – The receiving RTS may choose to ACCEPT or REJECT the S-CONNECT.Indication. The Result parameter indicates which has occurred. Some qualification of the reason for the rejection may be present in the User Data parameter. The Session provider may also reject the connection request under some circumstances.

*Note 4* – The parameters "Extended Control" and "Optimized Dialogue Transfer" are set to "not required". The remaining parameters are set such that default values are used.

*Note 5* – This parameter specifies the functional units to be used as listed in § 4.1 above.

*Note 6* – Set to zero.

*Note 7* – The connecting RTS will always request that the data token be available to obtain either a one-way monologue or two-way alternate dialogue session.

In the OPEN case, the connecting RTS will specify which RTS will initially hold the data token (minor synch token and major/activity token) upon successful completion of the connection phase, according to the initial-turn parameter supplied by means of OPEN.request.

The connecting RTS must assign all of the tokens to the same RTS. The connection may be rejected if this rule is violated. At any particular point in time, the holder of the tokens is referred to as the *sending* RTS, the other as the *receiving* RTS.

In the RECOVER case, the following rules apply:

a) If the RTS has the tokens, it specifies that it retains them.

b) If the RTS does not have the tokens, but has issued an S-CONTROL-GIVE.request with no confirmation that the tokens were received, it gives the tokens to the other RTS. (Receipt of data serves as confirmation that the tokens were received.)

c) If the RTS does not have the tokens and does not have an S-CONTROL-GIVE.request outstanding, it specifies that the assignment of the token is "acceptor chooses". In this case, the responding RTS will either keep or return the tokens depending upon whether it had them before the session was aborted.

*Note 8* – In the S-CONNECT.request, this parameter contains a (Presentation Layer) **PConnect** data element. In the S-CONNECT.response, it contains a **PAccept** or **PRefuse** data element depending upon the Result parameter. These data elements are defined as follows:

```
PConnect :: = SET {
    [0] IMPLICIT DataTransferSyntax,
    [1] IMPLICIT pUserData SET {
        checkpointSize [0] IMPLICIT INTEGER DEFAULT 0,
        windowSize [1] IMPLICIT INTEGER DEFAULT 3,
        dialogueMode [2] IMPLICIT INTEGER {monologue(0), twa(1)} DEFAULT monologue,
        [3] ConnectionData,
        applicationProtocol [4] IMPLICIT INTEGER {p1(1), p3(3)} DEFAULT p1}}

PAccept :: = SET {
    [0] IMPLICIT DataTransferSyntax,
    [1] IMPLICIT pUserData SET {
        checkpointSize [0] IMPLICIT INTEGER DEFAULT 0,
        windowSize [1] IMPLICIT INTEGER DEFAULT 3,
        [2] ConnectionData}}

PRefuse ::= SET {[0] IMPLICIT RefuseReason}

DataTransferSyntax :: = SET {[0] IMPLICIT INTEGER {X409(0)}}
```

The only value defined for the **DataTransferSyntax** parameter refers to the Presentation Transfer Syntax specified in Recommendation X.409.

```
ConnectionData :: = CHOICE {
    open [0] ANY, -- RTS user data
    recover [1] IMPLICIT SessionConnectionIdentifier}

RefuseReason :: = INTEGER {
    rtsBusy(0), cannotRecover(1), validationFailure(2), unacceptableDialogueMode(3)}
```

The **SessionConnectionIdentifier** parameter is used to specify the previous session connection which was abnormally terminated. This is used in order to relate the new session to the existing RTS association.

The **windowSize** parameter allows negotiation of the number of outstanding minor synchronization points before data transfer must be suspended. The receiving RTS must supply a value in the Response or Confirmation that is less than or equal to the value requested. This becomes the agreed size and governs both directions of transfer.

The **checkpointSize** parameter allows negotiation of the maximum amount of data (in units of 1024 octets) that may be sent between two minor synchronization points. A value of zero indicates that no checkpointing will be done. The receiving RTS must supply a value in the Response or Confirmation that is less than or equal to the value requested. Exceptionally, it may supply any value in the case that the value of zero has been requested by the transmitting RTS. The value supplied by the receiving RTS becomes the agreed maximum value and governs both directions of transfer.

### 4.3 *Data Transfer phase: actions of sending RTS*

Each APDU, conveyed in an RTS TRANSFER.request, constitutes a Session Activity. For each session connection, only one activity may exist at a time, and only one interrupted activity awaiting resumption may exist at a time, including any that may have been aborted by S-ABORT.

*Note* – A block of small APDUs could be conveyed in a single TRANSFER.request to improve transmission efficiency. This is for further study.

An activity is transferred as a single SSDU if checkpointing is not used or the APDU is smaller than the **checkpointSize**. Otherwise, the APDU is transferred as a series of SSDUs, the maximum size of each being the negotiated **checkpointSize**.

*Note* – The possibility of allowing more than one SSDU between minor synchronization points if for further study.

#### 4.3.1 *Starting a new activity*

To start a new activity, the sending RTS (that which holds the tokens) issues an S-ACTIVITY-START.request. The sending RTS may start transmitting the APDU in an S-DATA.request immediately after the S-ACTIVITY-START.request is issued, since the latter service is not confirmed.

##### 4.3.1.1 *S-ACTIVITY-START (New APDU)*

| Parameter | Req/Ind |
|---|---|
| Activity Identifier | See Note |
| SS-User Data | – |

*Note* – The Activity Identifier identifies the APDU by means of a serial number. The first APDU started on the session connection is assigned the number 1. Each successive APDU for that direction of transfer is assigned the next number. Thus numbering is separate for each direction of transfer.

#### 4.3.2 *Transmitting an APDU*

The maximum SSDU size, as stated previously, will have been negotiated during the connection phase. The sending RTS must submit in S-DATA.requests, SSDUs that conform to that agreement.

After each SSDU (S-DATA.request), the sending RTS normally inserts a checkpoint (S-SYNCH-MINOR.request) or indicates the end of the activity (S-ACTIVITY-END.request). In abnormal circumstances, such as upon receipt of an error report from the receiving RTS, the sending RTS may interrupt the activity (S-ACTIVITY-INTERRUPT.request), discard the activity (S-ACTIVITY-DISCARD.request), or even abort the connection (S-U-ABORT.request), as described in § 4.3.5.

Consecutive S-DATA.requests must not be given, and all data transfer must take place within an activity.

4.3.2.1 *S-DATA*

| Parameter | Req/Ind |
|---|---|
| SS-User Data | See Note |

*Note* – The maximum size is as described above.

4.3.3 *Insertion of checkpoints*

Checkpoints may only be inserted if a **checkpointSize** greater than zero was negotiated during the connection phase.

To facilitate checkpointing, the receiving RTS must be able to secure each SSDU. The sending RTS will interpret a confirmed checkpoint as signifying that the data has been secured and therefore does not require retransmission.

To insert a checkpoint into the data stream, the sending RTS uses the minor synchronization service which has the following primitives and parameters.

4.3.3.1 *S-SYNCH-MINOR*

| Parameter | Req/Ind | Resp/Conf |
|---|---|---|
| Type | Note 1 | |
| Synchronisation Point Serial Number | Note 2 | Note 2 |
| SS-User Data | – | – |

*Note 1* – The RTS uses only the "explicit confirmation expected" type of minor synchronization. Each minor synchronization must be confirmed in the order received by the receiving RTS. When the sending RTS receives the confirmation to a minor synch, it assumes that the receiving RTS has secured the data up to that point. If the receiving RTS has detected a problem, it should not confirm the minor synchronization, but rather should issue an S-U-EXCEPTION-REPORT.request or S-U-ABORT.request, depending upon the severity of the problem.

The sending RTS may issue further S-DATA.requests and S-SYNCH-MINOR.requests unless the agreed window size has been reached. The window is advanced when a Confirmation is received. The window is not policed by the Session Layer.

The minor synchronization window mechanism should not be used for flow control. The Transport Service provides a back-pressure flow control mechanism to control the rate of acceptance of data, in units of less than an SSDU.

*Note 2* – The Session-provider allocates checkpoint serial numbers and passes them to the sending and receiving RTSs to associate with the transmitted data.

4.3.4 *Normal termination of an activity*

When an entire APDU has been transmitted, the sending RTS normally issues an S-ACTIVITY-END.request. An activity end is an implicit major synchronization point and once successfully confirmed indicates to both RTSs that the APDU has been secured. The sending RTS may then delete the transmitted APDU.

If the receiving RTS cannot secure the APDU, it must not confirm the activity end. It may either issue an S-U-EXCEPTION-REPORT.request or S-U-ABORT.request, depending upon the severity of its local problem (see § 4.4.1).

4.3.4.1 *S-ACTIVITY-END*

| Parameter | Req/Ind | Resp/Conf |
|---|---|---|
| Synchronisation point Serial Number | See Note | |
| SS-User Data | – | – |

*Note* – The serial number of the implied major synchronization point is allocated by the Session-provider and passed up to both RTSs.

4.3.5 *Other actions of the sending RTS during APDU transfer*

If it detects a local system problem or receives an S-U-EXCEPTION-REPORT.indication (see § 4.4.1), the sending RTS may take one of the following actions:

1) Interrupt the current activity by issuing an S-ACTIVITY-INTERRUPT.request.
2) Discard the current activity by issuing an S-ACTIVITY-DISCARD.request.
3) A discarded activity cannot be resumed, and an RTS that receives such an indication should delete all knowledge and contents of the associated APDU so far received. If the sending RTS wishes to retransmit a previously discarded activity (APDU), it must reintroduce it as a new activity (using S-ACTIVITY-START.request).
4) Abort the session connection by issuing an S-U-ABORT.Request, which also has the effect of interrupting the current activity.

*Note* – The following mechanism has been proposed and is for further study. The sending RTS may also choose to interrupt the current activity (APDU in transfer), for example, so it can send a higher priority APDU or a shorter APDU of the same priority. The sending RTS may only interrupt one activity and only if an activity is not already suspended by either RTS. The connection is then outside an activity and can be said to be in a between-activities state.

4.4 *Data Transfer phase: actions of receiving RTS*

4.4.1 *Responding to problems*

If the receiving RTS detects a problem during receipt of an APDU (for example, within an activity), it may issue either an S-U-ABORT.request or S-U-EXCEPTION-REPORT.request. Aborting the session connection is the most severe action; this is described in § 4.6.2. By issuing an S-U-EXCEPTION-REPORT.request, the

receiving RTS indicates a less severe problem which it anticipates the sending RTS can overcome, and allows the sending RTS to take the most appropriate course of action. The actions the sending RTS may take are described in § 4.3.5.

In exceptional circumstances, the receiving RTS may have to delete a partially received activity, even though some minor synchronization points have been confirmed. In this case, the RTS responds to an S-ACTIVITY-START.indication for that activity by issuing an S-U-EXCEPTION-REPORT.request with the "unrecoverable procedure error" reason code. Alternatively, the receiving RTS may issue an S-U-ABORT.request.

#### 4.4.1.1 *S-U-EXCEPTION-REPORT*

| Parameter | Req/Ind |
|---|---|
| Reason | See Note |
| SS-User Data | – |

*Note* – This parameter may specify one of the following reasons:

a) receiving ability jeopardized
b) local ss-user error
c) sequence error
d) unrecoverable procedure error
e) non specific error

### 4.4.2 *Requesting control of the session connection*

When the RTS-user issues a TURN-PLEASE.request, the RTS issues an S-TOKEN-PLEASE.request. This may be done either inside or outside an activity. Upon receipt of the S-TOKEN-PLEASE.Indication, the sending RTS issues a TURN-PLEASE.indication to the RTS-user.

#### 4.4.2.1 *S-TOKEN-PLEASE*

| Parameter | Req/Ind |
|---|---|
| Tokens | Note 1 |
| SS-User Data | Note 2 |

*Note 1* – The receiving RTS will only request the data token. Since, for message handling, the tokens cannot be separated, the sending RTS always surrenders all of the other available tokens when issuing the S-CONTROL-GIVE request.

*Note 2* – This is the priority parameter of the TURN-PLEASE.request primitive, defined as follows:

**Priority ::= INTEGER**

To ensure that an RTS user's request for the turn is not lost during a Session abort/reconnection, the following rule applies. An RTS that has issued an S-TOKEN-PLEASE.request but not received an S-CONTROL-GIVE.indication should re-issue the S-TOKEN-PLEASE.request after session recovery.

### 4.5 *Possible actions of sending RTS outside an activity*

The sending RTS may choose to abort or release the session connection when there is no current activity, as described in § 4.6. It may also choose to send a new APDU (or a previously discarded APDU) by starting another activity and transferring the APDU as described in § 4.3.1. Other actions are described in the following subsections.

#### 4.5.1 *Resuming an interrupted activity*

The sending RTS may continue an APDU that was previously interrupted by an S-ACTIVITY-INTERRUPT.request or user- or provider-initiated session abort. To do so, it uses the S-ACTIVITY-RESUME.request primitive. Parameters are supplied to link the continued activity with the previously interrupted part.

It is possible for the sending RTS to try to resume an activity that the receiving RTS believes to be complete. This happens when the session connection is aborted after the receiving RTS has confirmed the activity end, and the confirmation is lost. To overcome this, the following rules should be followed:

1) As the first activity started on a new connection, the sending RTS must attempt to resume the last interrupted activity (for example, the one that was current when the connection was aborted).
2) The receiving RTS should always record the Connection Id and Activity Id of the last APDU which is completely secured.
3) The receiving RTS should check the details of each resumed activity to see if it coincides with the last APDU the RTS secured. If it does, the receiving RTS should respond correctly to the sending RTS but discard the data it receives.

If the time to transfer an APDU has expired, the sending RTS should generate an EXCEPTION.Indication unless an S-ACTIVITY END.confirm is outstanding. Since the APDU might be completely received by the other RTS, the sending RTS should retain the APDU until the activity is successfully resumed or discarded. This avoids the possibility of generating duplicate APDUs.

##### 4.5.1.1 *S-ACTIVITY-RESUME*

| Parameter | Req/Ind |
|---|---|
| Activity Identifier | Note 1 |
| Old Activity Identifier | Note 2 |
| Synchronisation Point Serial Number | Note 3 |
| Old Session Connection Identifier | Note 4 |
| SS-User Data | – |

*Note 1* – The issuing RTS must allocate and supply the next Activity Identifier number for the current session.

*Note 2* – The RTS must supply the Activity Identifier assigned to the previously interrupted part of the APDU.

*Note 3* – The RTS will specify the Serial Number of the last confirmed checkpoint in the interrupted APDU. The Session-provider will also set the current session serial number to this value. If there was no previously confirmed checkpoint, the activity (APDU) cannot be resumed and thus must be discarded.

*Note 4* – If the interrupted APDU was suspended during a previous session connection (or on another current session connection), the Session Connection Identifier of that session must be supplied for recovery.

The Session Connection Identifier of the previous (or current other) session connection is conveyed in the Calling SS-user Reference, Common Reference, and, optionally, Additional Reference Information components of this parameter. The Called SS-user Reference component is not used.

4.5.2 *Giving control of the session to the receiving RTS*

When an RTS-user issues a TURN-GIVE.request, the sending RTS will give control of the session to the other RTS using the S-CONTROL-GIVE.request primitive.

4.5.2.1 *S-CONTROL-GIVE*

The data, minor synch, and major/activity tokens are automatically passed to the other RTS.

*Note* – This enables S.62 CSCC/RSCCP to be exchanged without parameters, that is, compatibly with Recommendation S.62.

4.6 *Session connection termination phase*

4.6.1 *Orderly release*

The sending RTS issues an S-RELEASE request in response to a CLOSE.request. Upon receiving an S-RELEASE.indication, the RTS issues a CLOSE.indication and an S-RELEASE.response.

*Note* – The transport connection need not be released with the session connection and may be reused.

4.6.1.1 *S-RELEASE*

| **Parameter** | **Req/Ind** | **Resp/Conf** |
|---|---|---|
| Result | | see Note |
| SS-User Data | – | – |

*Note* – The responding RTS will always accept the release.

4.6.2 *User initiated abort*

If an RTS detects a serious problem, either inside or outside an activity, it may issue an S-U-ABORT.request. It should provide a reason parameter for diagnostic purposes, and if the abort was provoked by receipt of an invalid message handling parameter, the invalid parameter should also be reflected.

4.6.2.1 *S-U-ABORT*

| Parameter | Req/Ind |
|---|---|
| User Data | see Note |

*Note* – The User Data parameter contains an **AbortInformation** data element, defined as follows:

```
AbortInformation :: = SET {
    [0] IMPLICIT AbortReason OPTIONAL,
    reflectedParameter [1] IMPLICIT BIT STRING OPTIONAL}

AbortReason :: = INTEGER {
    localSystemProblem(0),
    invalidParameter(1), -- reflectedParameter supplied
    unrecognizedActivity(2),
    temporaryProblem(3), -- the RTS cannot accept a session for a period of time --
    protocolError(4) -- RTS level protocol error --}
```

It is assumed that the session is aborted by the session provider upon detection of any unrecoverable error, that is, that S-P-EXCEPTION-REPORT.indication is not issued.

In the case of a **temporaryProblem**, no attempt should be made to re-establish the session for a period of time to be determined by bilateral agreement.

4.6.3 *Provider initiated abort*

The Session-provider may abort a session connection for any of a variety of reasons (for example, transport connection failure or local or remote provider problem), indicated by the reason parameter.

4.6.3.1 *S-P-ABORT*

| Parameter | Ind |
|---|---|
| Reason | see Note |

*Note* – The following reason codes may be supplied:
a) transport disconnect
b) protocol error
c) undefined

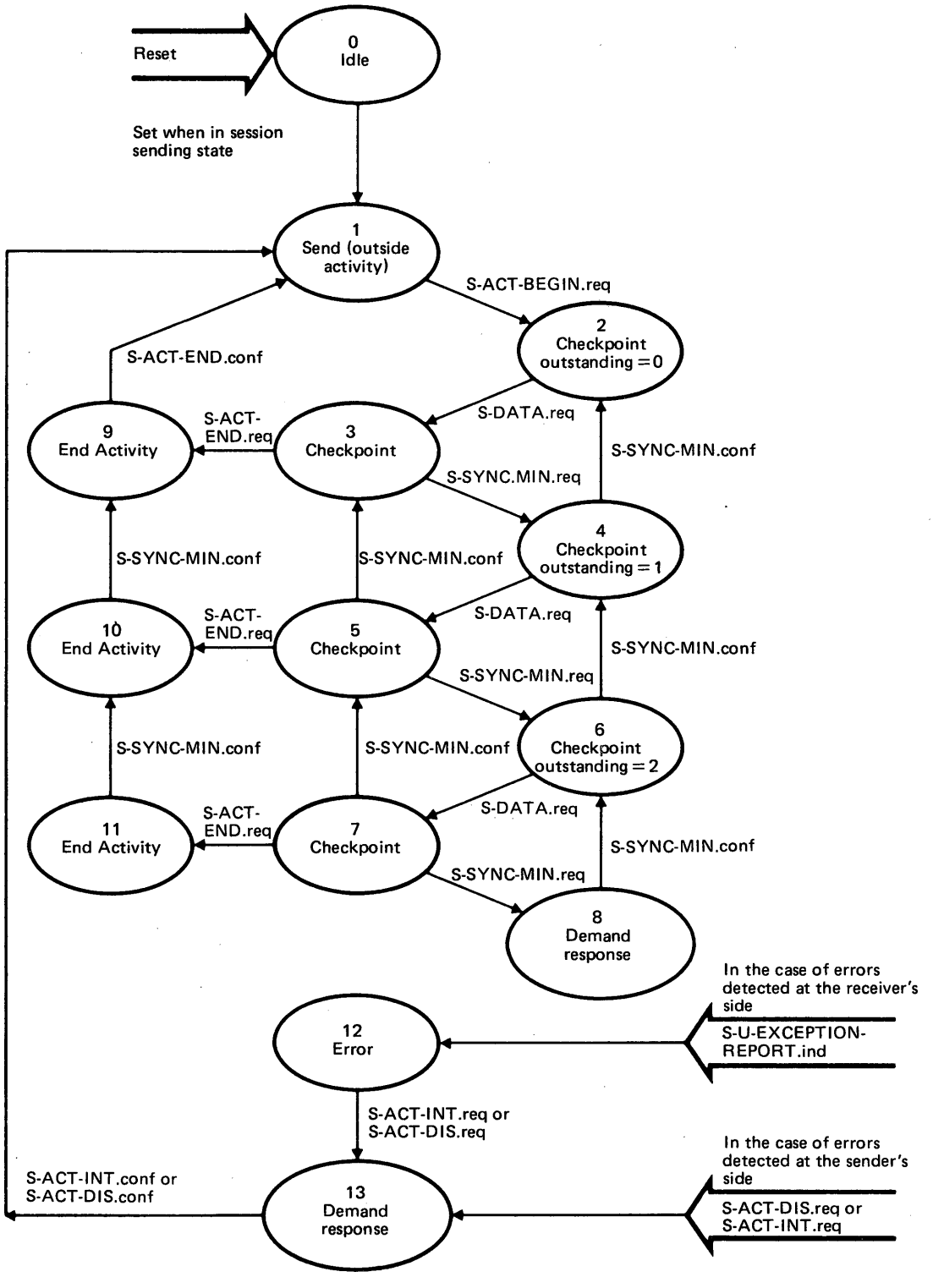
4.7 *State diagrams*

4.7.1 *Sending RTS*

Figure 12/X.410 specifies the legal interactions and relationships between the Session Service primitives when manipulated by the sending RTS.

4.7.2 *Receiving RTS*

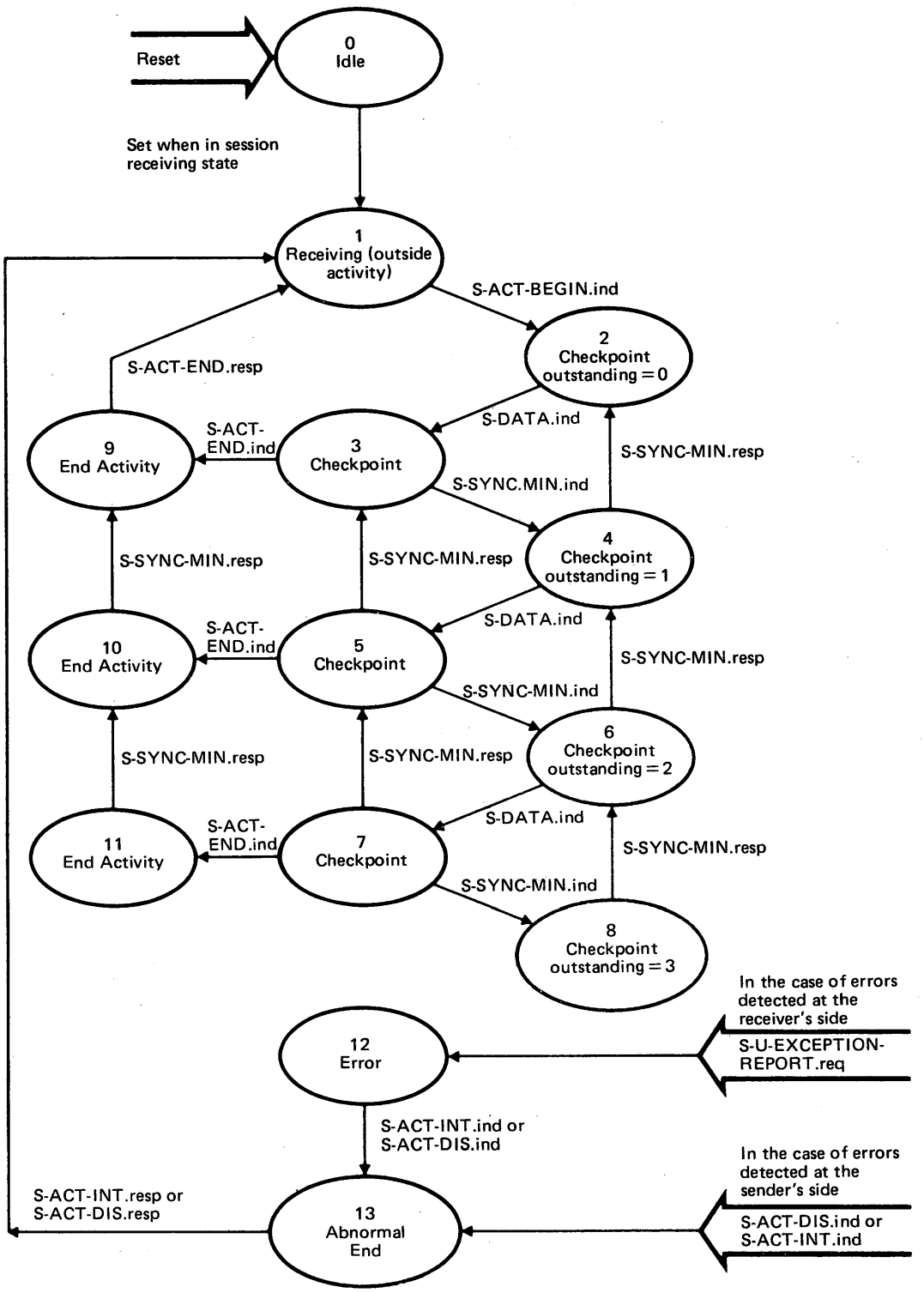
Figure 13/X.410 specifies the legal interactions and relationships between the Session service primitives when manipulated by the receiving RTS.




*Note* – This diagram assumes a window size of 3.

FIGURE 12/X.410

**State diagram for allowed sequences of session service primitives for sending RTS**




*Note* – This diagram assumes a window size of 3.

CCITT-38392

FIGURE 13/X.410

**State diagram for allowed sequences of sessions service primitives for receiving RTS**

## 5 Lower layer protocols

The following lower layer protocols are applicable to Message Handling Systems:

**Transport:** Recommendations on Transport Service (Recommendation X.214) and Transport Protocol (Recommendation X.224) with the following restrictions:

1) Only Class 0 is mandatory.
2) Class 1 is optional.
3) Transport Expedited Service is not used.

*Note* – The possible use of other Transport protocol classes is for further study.

Within the Transport Layer, the Transport Address is used to derive a TSAP Identifier and a Network Address. A TSAP Identifier is carried within the Transport Protocol and consists of a sequence of up to 16 International Alphabet No. 5 digits. A Network Address is as defined in Recommendation X.121.

**Network, Data Link**, and **Physical**: Administrations may, as a national matter, support MHS on networks of any type. Therefore a variety of protocols are applicable. Interworking between networks of different types, where required, will be provided in accordance with Recommendation X.300.

*Note* – The order of bits is preserved by the protocols used for MHS.

ANNEX A

(to Recommendation X.410)

### Abbreviations

The following abbreviations are used in this Recommendation.

| | |
|---|---|
| AE | Application Entity |
| APDU | Application protocol data unit |
| IA5 | International Alphabet No. 5 |
| IPM | Interpersonal messaging |
| MHS | Message handling system |
| MT | Message transfer |
| MTA | Message transfer agent |
| MTAE | Message transfer agent entity |
| NSDU | Network service data unit |
| OPDU | Operation protocol data unit |
| P1 | Message transfer protocol |
| P3 | Submission and delivery protocol |
| RTS | Reliable transfer server |
| SPDU | Session protocol data unit |
| SSAP | Session service access point |
| SSDU | Session service data unit |
| TPDU | Transport protocol data unit |
| TSAP | Transport service access point |
| TSDU | Transport service data unit |
| UTC | Coordinated universal time |

ANNEX B

(to Recommendation X.410)

**Relationships between layers**

For message handling, the relationships between the Application, Session, Transport, and Network Layers are as indicated below. The symbol "=" means "corresponds to".

B.1 At any point in time:

1 Session connection = 1 transport connection
1 transport connection = 1 network connection

B.2 For PDUs that contain user data:

| | |
|---|---|
| 1 OPDU (P3) or 1 MPDU (P1) | = 1 APDU |
| 1 APDU | = $m$ SSDU, where $m \geqslant 1$ |
| 1 SSDU | = 1 SPDU, or k SPDU where k > 1 if the session segmentation capability is negotiated at session establishment |
| 1 SPDU | = 1 TSDU |
| 1 TSDU | = $n$ TPDU, where $n \geqslant 1$ |
| 1 TPDU | = 1 NSDU |

B.3 A connection may be authenticated in the Transport Layer by using network qualified addresses (for example, X.121 addresses).

B.4 A transport connection may be reused (for example, in case of recovery).

ANNEX C

(to Recommendation X.410)

**Layer service description conventions**

In this Recommendation, and other Recommendations related to Message Handling Systems, the abstract model for "layer services" and the descriptive techniques used to define them are those which are outlined in Recommendation X.210, OSI Layer Service Conventions.

The conventions used for naming service primitives in the MHS Recommendations are in general conformance with the guidelines for primitive naming set forth in Recommendation X.210, but differ slightly in specific syntax. The MHS service primitive naming conventions are as follows:

1) The name of each primitive consists of two elements, a generic name which indicates the primitive group and a specific name which indicates the type of primitive.
2) Generic names and Specific names are chosen as described in Recommendation X.210, except that the name given to the fourth type of Specific primitive is "Confirmation" rather than "confirm".
3) A primitive name is represented as the Generic name followed by the Specific name and separated by a single character which is a period ("."); the Generic name is written entirely in capital letters while only the first letter of the specific name is a capital and all other characters of the Specific name appear in lower case.

The parameters associated with each service primitive are presented in a table formatted as shown in Figure C-1/X.410. The table displays the name, type, and a short description of each parameter, along with possible references to notes – located immediately below the table – which contain additional information.

| Parameter name | Type | Parameter description | Notes |
|---|---|---|---|
| ... | ... | ... | ... |
| ... | ... | ... | ... |

FIGURE C-1/X.410

**Parameter table format**

Each parameter is classified as one of the following two types:

1) Mandatory (M) – A parameter that must *always* be supplied (although the value may be null or defaulted where appropriate).

2) Conditional (C) – A parameter that is present under certain conditions. A note references the section of the Recommendation where the conditions or choice is explained.

APPENDIX I

(to Recommendation X.410)

**List of terms**

The following terms are contained in this Recommendation. References to definitions, implicit or explicit, are included where appropriate and where available.

## A

**Activity Discard**
cont. in X.410; impl. def. in X.410

**Activity End**
cont. in X.410; impl. def. in X.410

**Activity Interrupt**
cont. in X.410; impl. def. in X.410

**Activity Management**
cont. in X.410; impl. def. in X.410

**Application Entity**
cont. in X.410; impl. def. in X.200

**Application Layer**
cont. in X.410; expl. def. in X.200

**Application Protocol**
cont. in X.410; impl. def. in X.410

**Application Protocol Data Unit**
cont. in X.410; impl. def. in X.410

**Activity Resume**
cont. in X.410; impl. def. in X.410

**Activity Start**
cont. in X.410; impl. def. in X.410

**Association**
cont. in X.410; impl. def. in X.410

## B

**Bit Alignment**
cont. in X.410; impl. def. in X.410

## C

**Calling SSAP Address**
cont. in X.410; impl. def. in X.410

**Calling SSAP Identifier**
cont. in X.410; impl. def. in X.410

**Checkpoint**
cont. in X.410; impl. def. in X.410

**Confirmation**
cont. in X.410; impl. def. in X.410

## D

**Data Types**
cont. in X.410; impl. def. in X.409

**Dialogue-mode**
cont. in X.410; impl. def. in X.410

**Disposition**
cont. in X.410; impl. def. in X.410

## E

**Error Data Types**
cont. in X.410; impl. def. in X.410

**Establishment of an Association**
cont. in X.410; impl. def. in X.410

**Exceptions**
cont. in X.410; impl. def. in X.410

## G

**Give Control**
cont. in X.410; impl. def. in X.410

**Give Tokens**
cont. in X.410; impl. def. in X.410

## I

**Indication**
cont. in X.410; impl. def. in X.410

**Indication of Transfer Failure**
cont. in X.410; impl. def. in X.410

**Initial Data Token Assignment**
cont. in X.410; impl. def. in X.410

**Initial Major/Act Token Assignment**
cont. in X.410; impl. def. in X.410

**Initial Minor Synch Token Assignment**
cont. in X.410; impl. def. in X.410

**Initial-turn**
cont. in X.410; impl. def. in X.410

**Initiator-address**
cont. in X.410; impl. def. in X.410

**Initiator-RTS-address**
cont. in X.410; impl. def. in X.410

**Interactive Protocol**
cont. in X.410; impl. def. in X.410

**Interconnection**
cont. in X.410; impl. def. in X.410

**Interpersonal Messaging**
cont. in X.410; impl. def. in X.410

**Invoke**
cont. in X.410

# T



# U



*Note* – Terms with several words combined with a hyphen are generally names of service primitive parameters.

# MESSAGE HANDLING SYSTEMS: MESSAGE TRANSFER LAYER

*(Malaga-Torremolinos, 1984)*

The establishment in various countries of telematic services and computer-based store-and-forward message services in association with public data networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT,

*considering*

(a) the need for interpersonal messaging and message transfer services;

(b) the need to transfer messages of different types having a large variety of formats;

(c) that Recommendation X.1 includes specific user classes of service and X.10 categories of access; Recommendation X.2 defines optional user facilities for public data networks;

(d) that Recommendations X.20, X.20 *bis*, X.21, X.21 *bis*, X.22, X.25, X.28 and X.29 define DTE/DCE interfaces to public data networks;

(e) that Recommendations X.300 and X.121 permit international connections between public data networks;

(f) that the F-series of Recommendations defines telematic services and that the T-series of Recommendations defines terminal equipment and control procedures for telematic services;

(g) that the V-series of Recommendations provides the means for data communication over the telephone network;

(h) that Recommendation X.200 defines the reference model of open systems interconnection for CCITT applications;

(i) that a set of Recommendations describes various aspects of message handling: X.400, X.401, X.408, X.409, X.410, X.411, X.420 and X.430,

*unanimously declares:*

(1) that the general aspects of the message transfer layer are described in § 1;

(2) that the services of the message transfer layer are described in § 2;

(3) that the message transfer protocol is described in § 3;

(4) that the submission and delivery protocol is described in § 4.

TABLE OF CONTENTS

## 1 Purpose and scope

This Recommendation is one of a series of Recommendations and describes the Message Transfer Layer for Message Handling System (MHS). MHS service elements are provided by means of the Interpersonal Messaging (IPM) and Message Transfer (MT) Services. This Recommendation defines the conceptual "layer service" provided by the Message Transfer Layer (MTL), and the peer protocols of that layer.

Other Recommendations in the series contain additional information essential to a thorough understanding of the IPM and MT Services. Recommendation X.400 describes the MHS model and defines the IPM and MT service elements. Recommendation X.401 lists the basic service elements and optional user facilities. Recommendation X.408 specifies the algorithms that MHS uses when converting between different types of encoded information. Recommendation X.409 defines the notation and the representational technique used to specify and encode MHS protocols. Recommendation X.410 describes general techniques used for MHS protocols and the way in which Standard Open Systems Interconnection (OSI) protocols are used to support MHS applications. Recommendation X.420 specifies the protocol aspects of the IPM Service. Recommendation X.430 describes how Teletex terminals access the MHS.

Paragraph 2 of this Recommendation defines the MTL layer service. § 3 defines the Message Transfer Protocol (P1). § 4 defines the Submission and Delivery Protocol (P3).

## 2 Message transfer layer service

### 2.1 *Overview*

The layer service of the MTL provides User Agent Entities (UAEs) with the means for transferring messages. It delivers messages within a defined time period and, where required, performs encoded information type conversion on the contents of messages.

The MTL service is described by means of a number of service primitives, which are listed in § 2.2. Associated with each primitive are zero or more parameters. The MT service element (defined in Recommendation X.400) supported by each primitive or parameter is indicated in parentheses in the descriptions that follow. The primitives provide for the following:

1) the establishment and release of a dialogue between the UAE and the MTL;

2) the modification by the UAE of the values of its MTL-maintained registration parameters;

3) the temporary control by the UAE of the message types and lengths that the MTL may send to it;

*Note* – Capabilities 1, 2 and 3 may be used to support the Hold for Delivery, Access Management and Registered Encoded Information Types service elements.

4) the submission of a message by a UAE for transfer to one or more recipient UAEs (Message Transfer and Multi-destination Delivery). When submitting the message, the originating UAE can specify:

   a) the encoded information types of the message (Original Encoded Information Types Indication)

   b) the type of transfer service (Grade of Delivery Selection)

   c) that non-delivery notification is not to be provided (Prevention of Non-delivery Notification)

   d) that explicit delivery notification is to be provided (Delivery Notification)

   e) that conversion is not to be performed (Conversion Prohibition)

   f) whether the O/R names of all other originally specified recipient UAEs should be disclosed to the recipient UAEs when the message is delivered (Disclosure of Other Recipients)

   g) a data and time before which the message is not to be delivered (Deferred Delivery)

   h) that the message may be delivered to an alternate recipient (Alternate Recipient Allowed)

   i) that the message content is to be returned with any non-delivery notification (Return of Contents)

   j) the means to request that an explicit conversion be performed (Explicit Conversion)

   k) the type of the message, which identifies the User Agent protocol (Content Type Indication).

   The MTL then provides the originating UAE with:

   i) a means to identify the message (Message Identification)

   ii) the time of submission (Submission Time Stamp Indication);

5) the determination of whether or not a message could be delivered to one or more recipient UAEs if it were submitted (Probe). When requesting this determination, the originating UAE can specify:

   a) the encoded information types of the message (Original Encoded Information Types Indication)

   b) whether or not conversion is permitted (Conversion Prohibition)

   c) the length of the message

   d) the type of message, which identifies the User Agent protocol (Content Type Indication)

   e) that the message may be delivered to an alternate recipient (Alternate Recipient Allowed)

   f) that explicit conversion is to be performed (Explicit Conversion).

   A delivery and/or non-delivery notification conveys the result of the Probe;

6) the delivery of a message to a specified UAE. Upon delivery, the MTL indicates:

   a) the original encoded information types of the message (Original Encoded Information Types Indication)

   b) whether or not conversion has been performed and, if so, the resulting encoded information types (Implicit Conversion and Converted Indication)

   c) the time at which the message was submitted (Submission Time Stamp Indication)

   d) the time at which the message was delivered (Delivery Time Stamp Indication)

   e) the O/R names of the other UAEs specified as recipients of the message, provided this service element was selected by the originating UAE (Disclosure of Other Recipients)

   f) the type of transfer service (Grade of Delivery Selection)

   g) whether or not conversion was permitted (Conversion Prohibition)

   h) the type of the message, which identifies the User Agent protocol (Content Type Indication)

   i) the intended recipient of the message if the message is delivered to an alternate recipient (Alternate Recipient Assignment);

7) non-delivery notification in the event that a message is not successfully delivered to an intended recipient UAE. The MTL indicates to the originating UAE the reason for the delivery failure (Non-delivery Notification);

8) delivery notification in the event that the originating UAE requests explicit confirmation of message delivery (Delivery Notification). The MTL indicates to the originating UAE:
   a) the time at which the message was delivered (Delivery Time Stamp)
   b) if conversion was performed, the resulting encoded information types (Converted Indication);
9) the ability to request cancellation of a message previously submitted for deferred delivery (Deferred Delivery Cancellation).

## 2.2 *Specification of primitives*

The MT layer service is described as a set of *service primitives.* Service primitives are abstractions. They attempt to capture only those details of the interaction between entities that are aspects of the layer service itself. A service primitive neither specifies nor constrains the implementation of entities or the interfaces between them.

The service primitive descriptions that follow adhere to the conventions outlined in Annex C of Recommendation X.410.

### 2.2.1 *(UAL) LOGON: User-initiated Access Establishment Service*

A UAE issues the *LOGON.Request* primitive to initiate interaction with an MTAE.

Two events are associated with (UAL) LOGON, as illustrated in Figure 1/X.411.

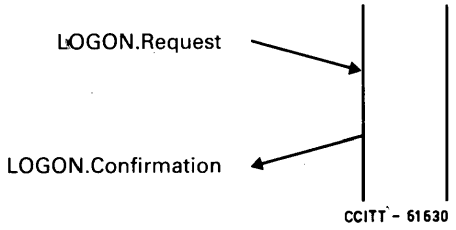



FIGURE 1/X.411

**(UAL) LOGON primitive events**

The parameters of the LOGON.Request primitive are:

| Parameter name | Type | Parameter description |
|---|---|---|
| O/R-name | M | The O/R name of the user requesting access to the MTAE |
| Password | M | The character string that validates the UAE |

The parameters of the LOGON.Confirmation primitive are:

| Parameter name | Type | Parameter description | Notes |
|---|---|---|---|
| Success-indication | M | Indicates the success of failure of the LOGON.Request | |
| Failure-reason | C | The reason for rejection of the LOGON.Request | 1 |
| Messages-waiting | C | Indicates the number of messages and total number of octets waiting to be delivered, for each priority (if LOGON succeeds) | 2 |

*Note 1* – This parameter is present only if the LOGON.Request fails. The possible reasons for which the request might be refused include that the user-O/R-name or password is invalid (Validation Failure) and that the system is busy.

*Note 2* – This parameter is present only if the user has subscribed to the Hold for Delivery service element.

2.2.2 *(MTL) LOGON: MTL-initiated Access Establishment Service*

The MTL issues the *LOGON.Indication* primitive to inform the UAE that access to it has been established (for example, for delivery of messages).

Two events are associated with (MTL) LOGON, as illustrated in Figure 2/X.411.

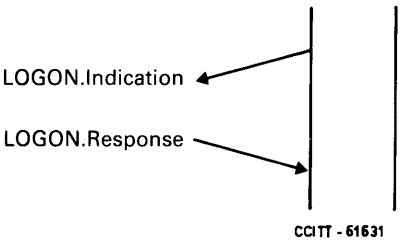



FIGURE 2/X.411

**(MTL) LOGON primitive event**

The parameters of the LOGON.Indication primitive are:

| Parameter name | Type | Parameter description | Notes |
|---|---|---|---|
| MTAE | M | The name of the MTAE establishing access to the UAE. | |
| Messages-waiting | C | Indicates the number of messages and total number of octets waiting to be delivered, for each priority | 1 |
| Password | M | The character string which validates the MTAE | |

*Note 1* – This parameter is present only if the user has subscribed to the Hold for Delivery service element.

The parameters of the LOGON.Response primitive are:

| Parameter name | Type | Parameter description | Notes |
|---|---|---|---|
| Success-indication | M | Indicates the success or failure of the LOGON.Request | |
| Failure-reason | C | The reason for rejection of the LOGON.Indication | 1 |

*Note 1* – This parameter is present only if the LOGON.Indication is refused. Reasons defined are: ValidationFailure, Busy.

### 2.2.3 *LOGOFF: Access Termination Service*

A UAE issues the *LOGOFF.Request* primitive to terminate interaction with an MTAE.

Two events are associated with LOGOFF, as illustrated in Figure 3/X.411.

The LOGOFF.Request and LOGOFF.Confirmation primitives have no parameters.

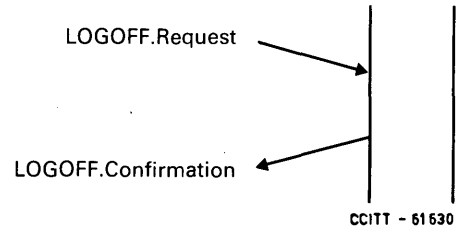



FIGURE 3/X.411

**LOGOFF primitive events**

### 2.2.4 *REGISTER: Registration Service*

A UAE issues the *REGISTER.Request* primitive to change the values of its MTAE-maintained registration parameters. The changes remain in effect indefinitely, unless changed again by means of a subsequent REGISTER.Request primitive.

Two events are associated with REGISTER, as illustrated in Figure 4/X.411.

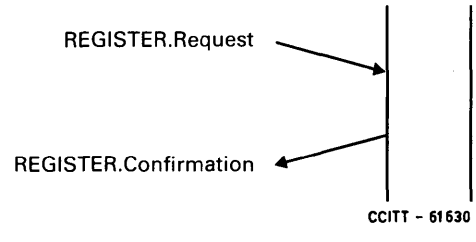



FIGURE 4/X.411

**REGISTER primitive events**

The parameters of the REGISTER.Request primitive are:

| Parameter name | Type | Parameter description |
|---|---|---|
| Deliverable-encoded-information-types | C | The encoded information types of the messages of which the UAE can take delivery. It is possible to include the value "undefined". |
| Maximum-content-length | C | The length of the longest message content the UAE can accept |
| Default-control-settings | C | The default values for the parameters settable by means of the CONTROL.Request primitive. They take effect immediately upon every Logon and remain in effect until over-ridden by a CONTROL.Request or CONTROL.Indication |
| Address | C | The network or transport address of the UAE if it has changed and is needed by the MTA |
| O/R-name | C | The O/R name of the UAE if it has changed |

The parameters of the REGISTER.Confirmation primitive are:

| Parameter name | Type | Parameter description |
|---|---|---|
| Success-indication | M | Indicates the success or failure of the REGISTER.Request |
| Failure-reason | C | The reason for rejection of the REGISTER.Request |

2.2.5 *(UAL) CONTROL: Hold for Delivery Service*

A UAE issues the *CONTROL.Request* primitive to change the restrictions that control what messages the MTL can send to it. The new restrictions override those currently in force and remain in effect until Logoff or a subsequent CONTROL.Request or a more restrictive REGISTER.Request is issued.

The control restrictions imply that those messages will be held by the MTAE for delivery. The control restrictions only apply to messages that can be delivered in accordance with the register parameters.

Two events are associated with CONTROL, as illustrated in Figure 5/X.411.

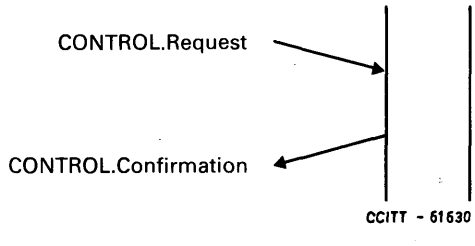



FIGURE 5/X.411

**(UAL) CONTROL primitive events**

The parameters of the CONTROL.Request primitive are:

| Parameter name | Type | Parameter description |
|---|---|---|
| Permissible-encoded-information-types | C | The encoded information types of the messages the MTAE may send to the UAE. This set must be a subset of the currently registered set (see § 2.2.4) |
| Maximum-content-length | C | The length of the longest message content the MTAE may send to the UAE. This value must not exceed the currently registered value (see § 2.2.4) |
| Lowest-priority | C | The priority of the least urgent message the MTL may send to the UAE |
| Msgs | C | Indicates whether or not the MTL may deliver messages to the UAE |
| Notifications | C | Indicates whether or not the MTL may send notifications to the UAE |

The parameters of the CONTROL.Confirmation primitive are:

| Parameter name | Type | Parameter description |
|---|---|---|
| Success-indication | M | Indicates the success or failure of CONTROL.Request |
| Failure-reason | C | The reason for rejection of the CONTROL.Request |
| Messages-held | C | Indicates whether messages and notifications are waiting at the MTAE due to restrictions imposed by means of CONTROL.Request |

### 2.2.6 *(MTL) CONTROL: Restriction Indication Service*

The MTL issues a *CONTROL.Indication* primitive to inform the UAE what messages will be accepted by the MTL from the UAE. The indications supersede those currently in force and remain in effect until Logoff or a subsequent CONTROL.Indication.

Two events are associated with (MTL) CONTROL, as illustrated in Figure 6/X.411.

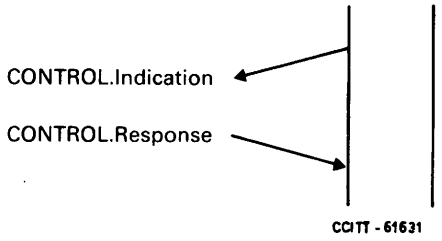



FIGURE 6/X.411

**(MTL) CONTROL primitive event**

The parameters of the CONTROL.Indication primitive are:

| Parameter name | Type | Parameter description |
|---|---|---|
| Maximum-content-length | C | The length of the longest message content the UAE may send to the MTAE |
| Lowest-priority | C | The priority of the least urgent message the UAE may send to the MTAE |
| Probes | C | Indicates whether or not the UAE may send probes to the MTAE |
| Msgs | C | Indicates whether or not the UAE may submit messages to the MTAE |

The parameter of the CONTROL.Response primitive is:

| Parameter name | Type | Parameter description |
|---|---|---|
| Messages-held | C | Indicates whether messages and Probes are waiting at the UAE due to restrictions imposed by means of CONTROL.Indication |

### 2.2.7 *SUBMIT: Message Submission Service*

A UAE issues the *SUBMIT.Request* primitive to initiate the transfer of a message to one or more recipients.

Two events are associated with SUBMIT, as illustrated in Figure 7/X.411.

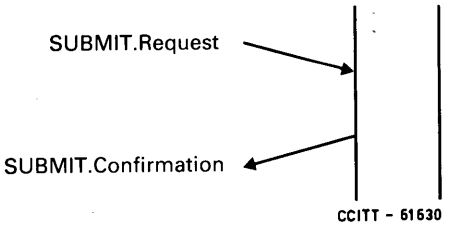



FIGURE 7/X.411

**SUBMIT primitive events**

The parameters of the SUBMIT.Request primitive are:

| Parameter name | Type | Parameter description | Notes |
|---|---|---|---|
| Recipient-O/R-names | M | The O/R name(s) of the recipient(s) | 1 |
| Originator-O/R-name | M | The O/R name of the originator, which is given to the recipient when the message is delivered | 1, 2 |
| Content | M | The information to be transferred | |
| Content-type | M | The content type of the message | 5 |
| Encoded-information-types | C | The type(s) of the information being transferred | 3 |
| NDN-suppress | C | Suppresses non-delivery notification | 6, 7 |
| Priority | M | Selects the urgent, non-urgent, or normal delivery option | |
| Deferred-delivery-time | C | Specifies the date and time before which the message is not to be delivered | |
| Delivery-notice | C | Requests delivery notification | 4, 7 |
| Conversion-prohibited | C | Requests that no conversion be performed | |
| Disclose-recipients | M | Indicates whether or not all recipient O/R names are to be revealed when the message is delivered | |
| Alternate-recipient-allowed | C | Indicates that the message may be delivered to an alternate recipient | |
| Content-return | C | Indicates that the content is to be returned as part of any non-delivery notification | 6 |
| UA-content-id | C | An identifier for the content of the message generated by the UAE. It may be used by the UAE for correlation of notifications with the content submitted | |
| Explicit-conversion | C | Indicates that a specified conversion be performed | 7 |

*Note 1* – See Recommendation X.400 for the structure of O/R names.

*Note 2* – The local MTA must verify that the originator-O/R-name is legitimate, that is, that it denotes the originating UA and can be used as a recipient-O/R-name to send a message to the originator. As a local implementation detail, the originating MTAE may supply this parameter's value.

*Note 3* – This parameter is provided for the purposes of detecting encoded information type incompatibilities and determining the required implicit conversion.

*Note 4* – Non-delivery notification is automatically provided in the event that a message cannot be delivered to a recipient UA.

*Note 5* – In case the IPMS is the user of the MTL, the content type will indicate P2.

*Note 6* – The originating MTAE must not allow the combined use of the content-return and NDN-Suppress services in the same SUBMIT.Request.

*Note 7* – If the message is to be transferred to more than one recipient UA, a different value for this parameter may be specified for each recipient.

The parameters of the SUBMIT.Confirmation primitive are:

| Parameter name | Type | Parameter description | Notes |
|---|---|---|---|
| Success-indication | M | Indicates the success or failure of the SUBMIT.Request | 1 |
| Submission-time | C | The time at which the request was accepted | 2 |
| Submit-event-id | C | The SUBMIT.Request event identifier | 2, 3 |
| Failure-reason | C | The reason for rejection of the SUBMIT.Request | 4 |
| UA-content-id | C | An identifier for the content of the message generated by the UAE | |

*Note 1* – Success indicates only that the MTL has accepted the request to transfer the message. It does not imply that the transfer and delivery of the message have been completed. Failure indicates that the MTL is unable to take responsibility for transferring the message.

*Note 2* – This parameter is present only if the SUBMIT.Request succeeds.

*Note 3* – This identifier is unique and meaningful only at the service-access-point where the SUBMIT.Request was issued. It is the means by which the MTL and originating UAE refer at their interface to a previously submitted message (for purposes, for example, of indicating a delivery or non-delivery notification).

*Note 4* – This parameter is present only if the SUBMIT.Request fails. The possible reasons for which the request might be refused include:

a) The recipient-O/R-names or another parameter is improperly specified.
b) The MTL's resources are inadequate to handle the message.
c) The originator-O/R-name is invalid.
d) A service element is not subscribed.
e) Violation of the Control restrictions imposed by the MTL.

### 2.2.8 *PROBE: Probe Service*

A UAE issues the *PROBE.Request* primitive to supply submission parameters to the MTL and inquire whether a SUBMIT.Request with the same parameters would be expected to result in successful delivery. For each recipient specified in the Probe, a delivery or non-delivery notification is returned.

A Probe may inquire about a recipient UAE whose messages are being held for delivery (see § 2.2.5). The results of the Probe in this case are for further study.

Two events are associated with PROBE, as illustrated in Figure 8/X.411.

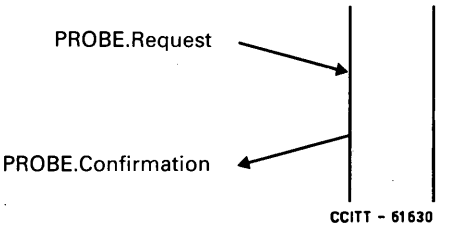



FIGURE 8/X.411

**PROBE primitive events**

The parameters of the PROBE.Request primitive are:

| Parameter name | Type | Parameter description | Notes |
|---|---|---|---|
| Recipient-O/R-names | M | One or more recipient O/R names | |
| Originator-O/R-name | M | The O/R name of the originator | |
| Content-type | M | The content type of the message | |
| Encoded-information-types | C | The encoded information type(s) of the message being probed | |
| Conversion-prohibited | C | Requests that no conversion (of the message being probed) be performed | |
| Alternate-recipient-allowed | C | Indicates that the Probe may test the possibility of delivery to an alternate recipient | |
| Content-length | C | The estimated length in octets of the content of the message being probed | |
| UA-content-id | C | An identifier for the content of the message generated by the UAE | |
| Explicit-conversion | C | Indicates that a specified conversion be performed | 1 |

*Note 1* – If the message is to be transferred to more than one recipient UA, a different value for this parameter may be specified for each recipient.

The parameters of the PROBE.Confirmation primitive are:

| Parameter name | Type | Parameter description | Notes |
|---|---|---|---|
| Success-indication | M | Indicates the success or failure of the PROBE.Request | |
| Probe-time | C | The time at which the request was accepted | 1 |
| Probe-event-id | C | The PROBE.Request event identifier | 1 |
| Failure-reason | C | The reason for rejection of the PROBE.Request | 2 |
| UA-content-id | C | An identifier for the content of the message generated by the UAE | 1 |

*Note 1* – This parameter is present only if the PROBE.Request succeeds.

*Note 2* – This parameter is present only if the PROBE.Request fails. The possible reasons for failure include the following:

a) The recipient-O/R-names or another parameter is improperly specified.
b) A control has been violated.
c) The originator-O/R-name is invalid.
d) A service element is not subscribed.
e) Violation of the Control Restriction imposed by the MTL.

### 2.2.9 *DELIVER: Message Delivery Service*

The MTL issues the *DELIVER.Indication* primitive to deliver a message to a recipient UAE. The recipient UA cannot refuse delivery of the message.

One event is associated with DELIVER, as illustrated in Figure 9/X.411.

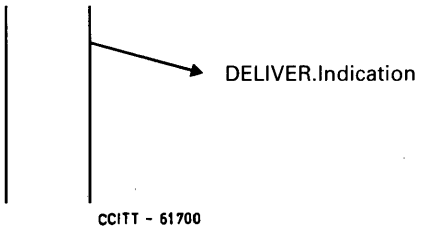



FIGURE 9/X.411

**DELIVER primitive event**

The parameters of the DELIVER.Indication primitive are:

| Parameter name | Type | Parameter description | Notes |
|---|---|---|---|
| Originator-O/R-name | M | The O/R name of the originator | 1 |
| This-recipient-O/R-name | M | The O/R name of this recipient | |
| Other-recipient-O/R-names | C | The O/R name(s) of all other specified recipients of this message | 2 |
| Content | M | The information being delivered | |
| Content-type | M | The content type of the message | |
| Converted-encoded-information-types | C | The encoded information type(s) of the information being delivered, if different from Original-encoded-information types | |
| Original-encoded-information-types | C | Whether or not conversion was performed, the message's original encoded information type(s) | |
| Delivery-time | M | The time at which the message is being delivered | |
| Submission-time | M | The time at which the message was submitted | |
| Priority | M | Indication of urgent, non-urgent, or normal delivery | |
| Intended-recipient. O/R-name | C | The O/R name of the intended recipient, as specified by the originator | 3 |
| Conversion-prohibited | C | Indication that conversion prohibition was requested by the originator | |
| Deliver-event-id | M | The DELIVER.Indication event identifier | |

*Note 1* – This parameter is the O/R name that was supplied by the originating UAE when the message was submitted and verified to be correct by the originating MTAE.

*Note 2* – When Disclosure of Other Recipients is selected by the originating UAE, these are the other recipient O/R names specified.

*Note 3* – This parameter is present only if the message is being delivered to an alternate recipient.

2.2.10 *NOTIFY: Message Notification Service*

The MTL issues the *NOTIFY.Indication* primitive to inform the UA that a previously submitted message was delivered or could not be delivered, or to convey the result of a Probe. The submission of a message or Probe addressed to two or more recipients may provoke several occurrences of this primitive.

One event is associated with NOTIFY, as illustrated in Figure 10/X.411.

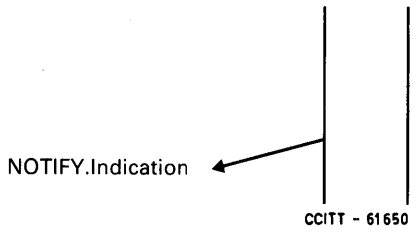



FIGURE 10/X.411

**NOTIFY primitive event**

The parameters of the NOTIFY.Indication primitive are:

| Parameter name | Type | Parameter description | Notes |
|---|---|---|---|
| Notification-type | M | Indicates whether the notification is of delivery or non-delivery | |
| Submit-or-Probe-event-id | M | Identifies the SUBMIT.Request or PROBE.Request event associated with the message or probe to which the notification refers | 1 |
| Recipient-O/R-names | M | The O/R name(s) of the original recipient(s) to whom the notification applies | |
| Non-delivery-reason | C | The reason for non-delivery | 2, 3, 7 |
| Delivery-time | C | The time at which the message was actually delivered. | 4, 7 |
| Converted-encoded-info.types | C | If conversion occurred, the resulting encoded information type(s) | 5, 6, 7 |
| Intended-recipient.O/R-name | C | The O/R name of the original recipient as specified by the originator, in case the failure occurred while attempting to deliver an alternate recipient or the message was delivered to an alternate recipient | 7 |
| Supplementary-information | C | Additional information for possible use by telematic services | 7 |
| Returned-Content | C | For non-delivery, the content of the message, if content return was requested under these circumstances | |
| Type-of-UA | C | For delivery, this parameter indicates whether the recipient UAE was privately owned, or owned by an Administration | 7 |
| UA-content-id | C | An identifier for the content of the message generated by the UAE | |

*Note 1* – The event-id corresponds to that supplied by the MTL in the SUBMIT.Confirmation or the PROBE.Confirmation primitive when the message was originally submitted and accepted.

*Note 2* – The possible reasons for non-delivery include the following:

a) The O/R name is unrecognized.
b) The O/R name is ambiguous.
c) The message could not be transferred between MDs, within a particular MD, or from the MTL to the recipient UAE.
d) The message's encoded information types are unsupported by the recipient UAE.
e) The maximum time for delivering the message expired. The UAE is not accepting messages (in case of Hold for Delivery).

*Note 3* – This parameter is present only if a non-delivery or probe failure is being reported.

*Note 4* – This parameter is present only if a successful delivery is being reported.

*Note 5* – This parameter is absent from the response to a probe.

*Note 6* – This parameter and the conversion indicated apply to all of the recipient-O/R-names.

*Note 7* – If the notification signals non-delivery to or probe failure for more than one recipient UAE, a different value for this parameter may be specified for each recipient.

### 2.2.11 *CANCEL: Cancel Deferred Delivery Service*

An originating UAE issues the *CANCEL.Request* primitive to request the originating MTAE to stop delivery (to a destination UAE) or to stop transfer (to another MTAE) of a previously submitted Deferred Delivery message. The originating UAE identifies the message whose delivery is to be cancelled by means of the submit-event-id, which corresponds to that supplied by the MTL in the SUBMIT.Confirmation primitive when the Deferred Delivery message was originally submitted and accepted. The originating UAE is notified of the result of the cancellation attempt.

Two events are associated with CANCEL, as illustrated in Figure 11/X.411.

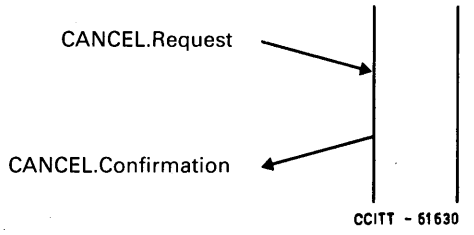



FIGURE 11/X.411

**CANCEL primitive events**

The parameter of the CANCEL.Request primitive is:

| Parameter name | Type | Parameter description | Notes |
|---|---|---|---|
| Submit-event-id | M | Identifies the SUBMIT.Request event associated with the message to which the CANCEL.Request refers | 1 |

*Note 1* – The submit-event-id corresponds to that supplied by the MTL in the SUBMIT.Confirmation primitive when the Deferred Delivery message was originally submitted and accepted.

The parameters of the CANCEL.Confirmation primitive are:

| Parameter name | Type | Parameter description | Notes |
|---|---|---|---|
| Success-indication | M | Indicates the success or failure of the CANCEL.Request | |
| Failure-reason | C | The reason for failure of the CANCEL.Request | 1 |

*Note 1* – Three possible reasons for failure are:

a) Invalid event id.
b) Message was delivered.
c) Message was transferred.

2.2.12 *(UAL) CHANGE-PASSWORD: Password Change Service*

The UAE issues a *CHANGE-PASSWORD.Request* primitive to indicate that the UAE's Logon password is to be changed to a new value.

Two events are associated with (UAL) CHANGE-PASSWORD, as illustrated in Figure 12/X.411.

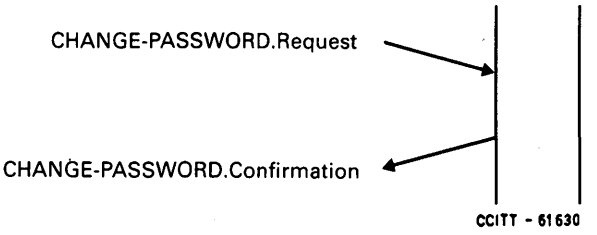



FIGURE 12/X.411

**(UAL) CHANGE-PASSWORD primitive events**

The parameters of the CHANGE-PASSWORD.Request are:

| Parameter name | Type | Parameter description |
|---|---|---|
| Old-password | M | The current password value |
| New-password | M | The new password value |

The parameters of the CHANGE-PASSWORD.Confirmation are:

| Parameter name | Type | Parameter description |
|---|---|---|
| Success-indication | M | Indicates the success or failure of the CHANGE-PASSWORD.Request |
| Failure-reason | C | The reason for failure of the CHANGE-PASSWORD.Request |

2.2.13 *(MTL) CHANGE-PASSWORD: Password Change Service*

The MTL issues a CHANGE-PASSWORD.Indication to indicate that the MTAE's password for that UAE is to be changed to a new value.

One event is associated with (MTL) CHANGE-PASSWORD, as illustrated in Figure 13/X.411.

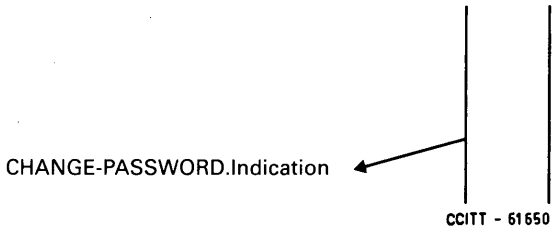



FIGURE 13/X.411

**(MTL) CHANGE-PASSWORD primitive event**

The parameters of the CHANGE-PASSWORD.Indication are:

| Parameter name | Type | Parameter description |
|---|---|---|
| Old-password | M | The current password value |
| New-password | M | The new password value |

## 3 Message Transfer Protocol (P1)

### 3.1. *Introduction*

The MTL provides various services to the User Agent Layer (UAL), as specified in § 2. Some of those services (for example, Hold for Delivery) are provided by means of functions located in a single MTAE and thus do not depend upon communication between MTAEs. The provision of other services, however, requires that two or more MTAEs cooperate with one another. Such cooperation is achieved by means of the *Message Transfer Protocol (P1)*.

The P1 protocol is used for communication between different Administration Management Domains (ADMDs) and to interconnect an ADMD and a Private Management Domain (PRMD). The operation of the P1 protocol between an ADMD and a PRMD is essentially the same as P1 operation between two ADMDs, except for some minor points which are defined in § 3.6. However, when an ADMD is attached to a PRMD, the ADMD is completely responsible for all messages which it accepts from or transfers to a PRMD. Furthermore, the ADMD assumes responsibility for any actions of the PRMD which affect such messages. As a result, the MTAE authentication and/or protocol validation tests which an MTAE in an ADMD applies when communicating with an MTAE in a PRMD may be more stringent than those which are used when communicating with another ADMD. The mechanisms of the P1 protocol permit both ADMDs and PRMDs to perform the necessary and desired level of verification.

The protocol elements of P1 are called *Message Protocol Data Units (MPDUs)*. MPDUs are classified as either *User MPDUs (UMPDUs)* or *Service MPDUs (SMPDUs)*, allowing easy identification of the functions for which cooperation is required. UMPDUs carry messages submitted by a UAE for transfer and delivery to another UAE. SMPDUs (for example, the Delivery Report MPDU) are used to convey information *about* messages between MTAEs.

The operation of P1 is such that MPDU parameters relating to Additional optional user facilities (as specified in Recommendation X.401) not supported by a particular relaying MD do not cause non-delivery of the message, and are passed transparently through the MD.

## 3.2 *Model of cooperating Message Transfer Agent Entities (MTAEs)*

The Message Transfer Protocol governs communication between MTAEs of different MDs. In this Recommendation, an MD is modelled as a single MTAE. The model describes the functions that an MD must perform, but dictates no aspects of the MD's internal structure or operation. In particular, it does not describe the functions performed when the MTAE (MD) serves both originator and recipient.

As illustrated in Figure 14/X.411, an MTAE executing the Message Transfer Protocol can be modelled as having three parts, the *Message Dispatcher*, the *Association Manager*, and the *Reliable Transfer Server (RTS)*. Figure 14/X.411 depicts an MTAE that serves two UAEs and is connected (adjacent) to two MTAEs. The role of the Message Dispatcher is to perform the P1 protocol actions indicated by the MPDUs that it receives from other MTAEs or that result from messages submitted by its UAEs. The role of the Association Manager is to establish, control and release the associations provided by the RTS. The RTS is responsible for providing associations and for completely and reliably transferring MPDUs to adjacent MTAEs by means of them. The functions of the MTAE that are strictly local and thus do not impact P1 (for example, the manner in which an MTAE delivers a message to a UAE that is directly attached to it) are not described by this model.

*Note* – The purpose of this MTAE model is to help explain the operation of the Message Transfer Protocol. It is not intended to constrain the design of an actual MTAE implementation.

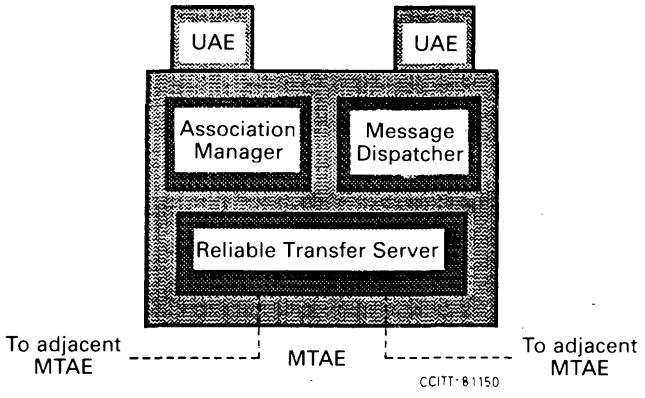



FIGURE 14/X.411

**Model of an MTAE**

From the Message Dispatcher's point of view, an MPDU is transferred to an adjacent Message Dispatcher in a single transaction. Both when the source Dispatcher initiates transfer of the MPDU and when the destination Dispatcher receives it, all of the information about the MPDU's transfer is passed between the Message Dispatcher and RTS in a single, logical service access event. The RTS performs all of the functions required to control and complete the MPDU's transfer. If the RTS is unable to transfer the MPDU, it so informs the source Message Dispatcher with an EXCEPTION.indication. The RTS and the way in which standard Open Systems Interconnection protocols are used to support the MTAE are described in detail in Recommendation X.410.

## 3.3 *Message Dispatcher*

### 3.3.1 *Function*

The Message Dispatcher performs the principal active functions of the MTAE, for example, relaying, generation of delivery reports, and content conversion.

3.3.1.1 *Relaying*

More than one MTAE may participate in the relaying and delivery of a message addressed to multiple recipients. If the various recipients are served by several different MTAEs, the message must be transferred through the MTS along several different paths, as illustrated in Figure 15/X.411. From the perspective of an MTAE relaying a message (for example, MTAE ≠ 1 in Figure 15/X.411), some recipients may be reached by one path while other recipients are reached by another. At such an MTA, two copies of the MPDU are created, and each is relayed to the next MTA along its respective path. The copying and branching of MPDUs is repeated until each copy has reached a final destination MTAE, where the message can be delivered to one or more recipient UAEs.

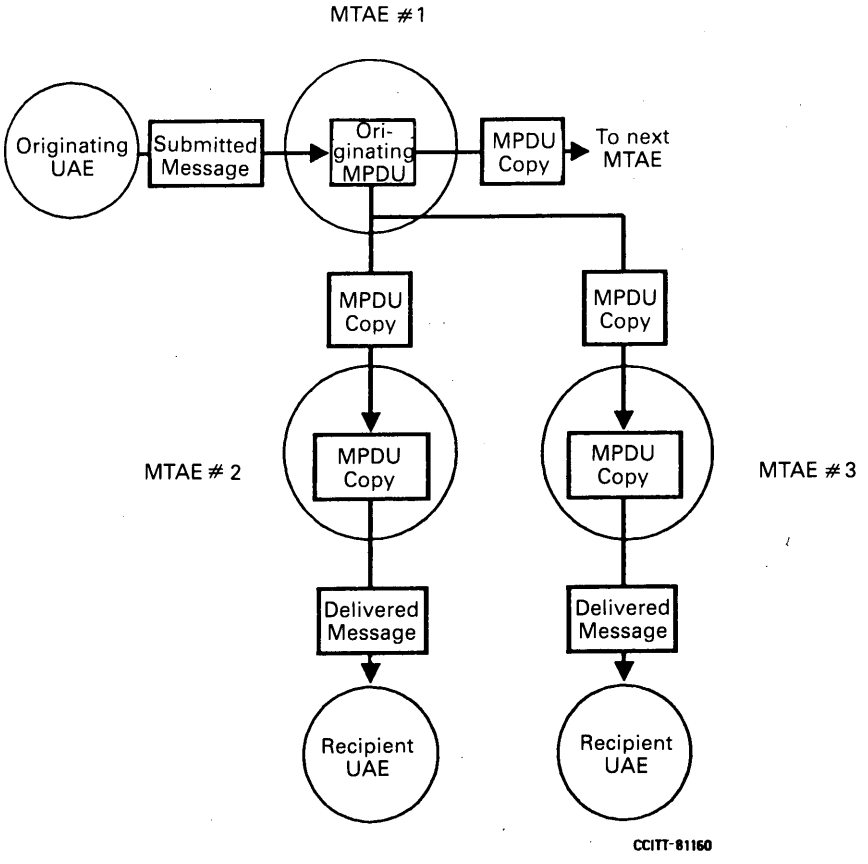



FIGURE 15/X.411

**MTS routing to multiple recipient UAEs**

Every MTAE along the path taken by an MPDU is responsible for delivering or relaying the message to a particular subset of the originally specified recipients. Other MTAEs take care of the delivery or relaying to remaining recipients, possibly using copies of the MPDU created along the way.

A *recipient information* parameter containing a *responsibility flag* appears in the MPDU envelope for each originally specified recipient. Each relaying MTAE indicates to the next MTAE the recipients for which is *not* to take responsibility in one of two ways.

1) By setting their responsibility flags to *false*. If Disclosure of Other Recipients has been requested, this approach must be taken.

2) By removing their recipient information parameters (**RecipientInfo** data element, see § 3.4.2) from the MPDU envelope.

The existence of the first alternative requires that MTAEs be able to process UMPDUs containing a mix of *true* and *false* responsibility flags, as well as UMPDUs containing only responsibility flags set to *true*.

3.3.1.2 *Delivery reports*

The MTL notifies UAEs of the delivery or non-delivery of messages. The Message Transfer Protocol facilitates this by relaying SMPDUs called *delivery report* MPDUs between Message Dispatchers. Delivery report MPDUs are generated in accordance with the value of the *report request* parameter, which is an additional piece of recipient information. Reports provoked by the report request parameter may be for use within the MTL or may be used to notify a UAE of the delivery or non-delivery of a message. A single report may serve both purposes.

A Message Dispatcher generates a positive delivery report upon successfully delivering a copy of a message to the recipient UA. It generates a negative delivery report upon determining that a copy of a message is undeliverable to a particular recipient, that is, that it can take neither of the following actions:

1) Deliver the message to the recipient UAE.
2) Relay the message to an adjacent MTAE that would take responsibility for relaying it further.

For efficiency, the Message Dispatcher may generate a single, combined delivery report that applies to several copies of a single multi-recipient message for which it is responsible. The Message Transfer Protocol makes allowance for this in the structure of the Delivery report MPDU. However, in order for delivery reports to be combined in this manner, the same content conversion, if any, must have been performed on the message for all recipients to whom the delivery report refers.

Delivery reports that pertain to copies of the same multi-recipient message but that were generated by different MTAEs are not combined in the Message Transfer Protocol. Instead, they remain distinct and are transferred independently, even when routed through the same intermediate MTAE.

3.3.1.3 *Content conversion*

When required, the MTAE performs content conversion. If the request for such conversion is implicit then neither the originating nor the recipient UAE specifically requests it; rather, the originating UAE simply indicates the (original) encoded information types of the message at submission time. The Message Transfer Protocol carries encoded information type parameters so that the necessary conversions can be performed unless explicitly prohibited by UAEs.

To determine that conversion is required, the MTAE must know which encoded information types a recipient UAE can receive. The MTAE may acquire this knowledge in three different ways:

1) When attaching to an MD (for example, at subscription time), the UAE indicates the encoded information types it supports.
2) The MTAE may deduce the type of conversion to be performed from the name of the MD that serves the intended recipient.
3) The recipient O/R name supplied by the originator may contain attribute(s) from which the encoded information types supported by the recipient's UAE can be deduced.

The possible conversions that may be performed are defined in Recommendation X.408.

*Note* – Conversion (for example, from Teletex to facsimile) may so increase the size of a message that performing the conversion in the final MD may be preferable.

3.3.2 *Operation of the message dispatcher*

In this section, an MD is modelled as a single MTAE. The model describes the functions that an MD must perform, but does not dictate the MD's internal structure. Similarly, the functions performed when the originator and recipient are served by the same MTAE are not described. This description assumes that a TRANSFER.Request may be issued to the RTS at any time. The queuing that would have to exist in any real implementation is not described. The description further assumes that any associations required have been opened by the Association Manager and that the turn is always available.

The Message Dispatcher is a logical machine that processes the Message Transfer Protocol. It is activated by either of the following two events at its interface with the UAL: a SUBMIT.Request or a PROBE.Request. It is also activated by either of two events at its interface with the RTS: a TRANSFER.Indication or an EXCEPTION.Indication.

*Note* – These primitives are assumed to be issued by a UAE co-located with the MTAE. The fact that they may be issued by a remote UAE, and carried to the MTAE by means of P3, does not affect P1's operation.

3.3.2.1 *Action on SUBMIT.Request Event*

1) The originator is validated (for example, by querying a local directory).

2) The parameters of the SUBMIT.Request are validated and a new UMPDU created. This validation is essentially syntactic; the semantics of the parameters are validated during the relaying of the MPDU.

3) The submit-event-id is generated.

4) If the preceding actions are correctly carried out, a SUBMIT.Confirmation event indicating success is generated. Otherwise, the SUBMIT.Confirmation indicates failure.

5) If Deferred Delivery is requested either the message is held until the Deferred Delivery time has arrived, and then handled in the normal way; or, if a bilateral agreement exists, it may be relayed to another MTA with the Deferred Delivery time parameter included in the envelope.

   *Note* – No procedure is defined to support the optional user facility of Deferred Delivery Cancellation in the case where deferred delivery is supported by an MTAE other than the originating MTAE. Such a procedure is for further study.

6) Each successfully generated UMPDU is passed to the RTS by means of a TRANSFER request.

3.3.2.2 *Action on PROBE.Request Event*

The actions taken are the same as those for a SUBMIT.Request, except that a Probe MPDU is created (unless there is a failure) and a PROBE.Confirmation event generated.

3.3.2.3 *Action on TRANSFER.Indication Event*

1) The MPDU envelope is validated. If the envelope is found invalid, a negative Delivery Report MPDU is returned, in the case of Probe and User MPDUs, provided that the originator O/R name is valid. The report carries a reason parameter indicating "unable to transfer". An invalid MPDU is not processed further, but the event should be recorded so that diagnostic and corrective action can be taken.

2) The global domain identifier of the MD and the arrival time are added to the trace information.

3) The Message Dispatcher selects the recipients (i.e. O/R names to which the MPDU is addressed) for which the routing algorithm is to be executed. For Delivery Report MPDUs there is only one. For Probe and User MPDUs the recipients are those for which the responsibility flag is *true*. For each recipient, three results are possible as follows:

   a) The MTA serves this recipient. The actions depend upon the MPDU type as follows:

      **User MPDU**: If the message can be delivered, the delivery-stamp is generated and a DELIVER.Indication primitive is issued. If the message cannot be delivered (for example, because of encoded information type incompatibility), a "last trace information" item is generated, incorporating the reason for non-delivery, If delivery is successful and if a confirmed or audit-and-confirmed report is requested, a "last trace information" item indicating that delivery has been successfully completed is created. The Message Dispatcher may create a Delivery Report MPDU and generate a TRANSFER.Request. Alternatively, it may combine this with the "last trace information" items for other recipients of the same message. At a later time, these will be incorporated in a Delivery Report MPDU, and a TRANSFER.Request will be generated.

      **Delivery Report MPDU**: The message is correlated with the related submission event for local purposes (for example, charging, statistics, or trace). If the returned trace information contains a non-delivery indication, a (negative) NOTIFY.Indication event with the associated parameters is issued (unless prevention of Non-Delivery Notification was requested at submission). If a delivery notification was requested and the returned trace information contains a delivery indication, a (positive) NOTIFY.Indication event is issued. If a copy of the MPDU was stored and all reports have been received, the MPDU is deleted. The time limit for storage when reports are not requested or do not come back is for further study.

**Probe MPDU**: If a DELIVER.Indication for a UMPDU with the same envelope parameters could be issued, the actions appropriate to completion of the DELIVER.Indication are taken. Otherwise, the actions for an undeliverable UMPDU are taken.

For User and Delivery Report MPDUs, this description assumes that the UA is logged on, and that no control restrictions are in force that prevent the MPDU from being passed to the UA. If such restrictions exist, the MPDU must wait, and this may cause the delivery to be abandoned after a period of time. Messages that cannot be delivered because of registration restrictions can be declared undeliverable immediately. The result of a probe takes into account registration but not control restrictions.

b) The MPDU must be relayed to another MTA. The action depends on MPDU type as follows:

**Probe or User MPDU**: The trace information is examined for loops. If a loop is detected, a Delivery Report MPDU with the appropriate trace information is generated. Otherwise, the Message Dispatcher proceeds as follows. If this is the first recipient for which the MPDU must be relayed to the particular other MTA, then a copy of the MPDU is created, with the responsibility flags set to *false* for all of the other recipients. Otherwise, the responsibility flag for this recipient is set to *true* in the existing copy.

**Delivery Report MPDU**: The trace information is examined for loops. If a loop is detected, the MPDU is discarded. Otherwise the MPDU is relayed.

c) The routing algorithm indicates an error in the O/R name.

A "last trace information" item indicating the error is generated. A Delivery Report MPDU may be generated or the trace information combined with other trace information.

4) After executing the routing algorithm for each recipient, a check is made to ensure that all necessary Delivery Report MPDUs have been generated. A TRANSFER.Request is then issued for each MPDU to be relayed. The transfer-time parameter is set, taking into account the time the message has already spent in the MTL and (possibly) the availability of an alternate route. Before a User or Probe MPDU is relayed, recipient information parameters whose responsibility flags are *false* may be removed, unless Disclosure of Other Recipients was requested.

3.3.2.4 *Action on EXCEPTION.Indication Event*

1) If the Message Dispatcher can perform alternate routing, the routing algorithm is re-executed for each recipient:

a) For each recipient for whom an alternate route is available, a determination is made whether a copy of the MPDU intended for other rerouted recipient(s) of that excepted MPDU has been prepared for the selected MTA. If so, the copy's responsibility flag for the current recipient is set to *true*. Otherwise a copy is created, and its responsibility flag for the recipient is set to *true*; all other responsibility flags are set to *false*.

b) The necessary trace information (indicating "rerouted" and the name of the MD to which the MPDU could not be transferred) is added to all MPDU copies being rerouted.

c) A TRANSFER.Request is issued for each UMPDU. Before transferring the MPDU, recipient information whose responsibility flags are *false* may be removed, unless Disclosure of Other Recipients was requested.

2) If the MPDU is a User or Probe MPDU then, for each recipient for whom no alternate route exists, or for the entire MPDU if the MTA does not perform alternate routing, a Delivery Report MPDU with the appropriate trace information is generated, and a TRANSFER request is issued for this Delivery report MPDU. Any copy which cannot be routed is then discarded.

3) If the MPDU is a Delivery Report MPDU, and no alternate route exists, the MPDU is discarded. The transfer may also be logged for accounting and/or diagnostic purposes.

## 3.4 *Specification of Message Protocol Data Units (MPDUs)*

This section specifies the structure of each MPDU type using the notation defined in Recommendation X.409. The binary representation of each MPDU can be inferred from its notational description with the aid of that Recommendation. Data elements identified as OPTIONAL either are "conditional" parameters of the layer service primitives or represent points where flexibility in the internal operation of the MTL is allowed.

*Note* – A limit on the maximum size of data element which can be represented with the definite form of length is for further study. However, this limit will not be less than 64 koctets.

As previously stated, two classes of MPDU are defined:

```
MPDU :: = CHOICE {[0] IMPLICIT UserMPDU, ServiceMPDU}
```

*User MPDUs* carry messages. *Service MPDUs* carry information *about* messages.

One type of User MPDU is currently defined. Formally defined in Figure 16/X.411, it carries a message toward one or more of its recipient UAs.

Two types of Service MPDU are currently defined:

```
ServiceMPDU :: = CHOICE {[1] IMPLICIT DeliveryReportMPDU, [2] IMPLICIT ProbeMPDU}
```

The *Delivery Report MPDU*, defined in Figure 17/X.411, carries a delivery or non-delivery report toward an originating UA. A *Probe MPDU*, defined in Figure 18/X.411, carries a request for information about the deliverability of a message toward one or more potential recipient UAs.

Every MPDU has two parts, envelope and content. The *envelope* contains the information the MTL requires to route the MPDU to its intended destination. Information in the envelope may be changed, added to, or deleted during the MPDU's passage through the MTL. The *content* is the primary information the MPDU is intended to convey. In general, it is carried unmodified to its destination.

### 3.4.1 *Common data types*

A number of data types figure in the definition of several MPDUs. These data types are defined in the following subsections.

#### 3.4.1.1 *O/R name*

An O/R name specifies the originator or recipient of a message according to the principles of naming and addressing laid out in Recommendation X.400. An O/R name consists of a number of standard attributes, followed optionally by a number of attributes defined by the MD to which the originator or recipient subscribes:

```
ORName :: = [APPLICATION 0] IMPLICIT SEQUENCE {
    StandardAttributeList,
    DomainDefinedAttributeList OPTIONAL}
```

The standard Attributes used in any O/R name are selected from those defined in Recommendation X.400.

```
StandardAttributeList :: = SEQUENCE {
    CountryName OPTIONAL,
    AdministrationDomainName OPTIONAL,
    [0] IMPLICIT X121Address OPTIONAL,
    [1] IMPLICIT TerminalID OPTIONAL,
    [2] PrivateDomainName OPTIONAL,
    [3] IMPLICIT OrganizationName OPTIONAL,
    [4] IMPLICIT UniqueUAIdentifier OPTIONAL,
    [5] IMPLICIT PersonalName OPTIONAL,
    [6] IMPLICIT SEQUENCE OF OrganizationalUnit OPTIONAL}
```

```
P1 DEFINITIONS ::=
BEGIN
-- P1 makes use of types defined in the following module
-- T73: Recommendation T.73
MPDU                    ::= CHOICE {[0] IMPLICIT User MPDU, Service MPDU}
ServiceMPDU             ::= CHOICE {[1] IMPLICIT DeliveryReportMPDU,
                                [2] IMPLICIT ProbeMPDU}
UserMPDU                ::= SEQUENCE {UMPDUEnvelope, UMPDUContent}
UMPDUEnvelope           ::= SET {
                                MPDUIdentifier,
                                originator ORName,
                                original EncodedInformationTypes OPTIONAL,
                                ContentType,
                                UAContentID OPTIONAL,
                                Priority DEFAULT normal,
                                PerMessageFlag DEFAULT {},
                                deferredDelivery [0] IMPLICIT Time OPTIONAL,
                                [1] IMPLICIT SEQUENCE OF PerDomainBilateralInfo OPTIONAL,
                                [2] IMPLICIT SEQUENCE OF RecipientInfo,
                                TraceInformation}
UMPDUContent            ::= OCTET STRING
-- time
Time                    ::= UTCTime
-- various envelope information
MPDUIdentifier          ::= [APPLICATION 4] IMPLICIT SEQUENCE {
                                GlobalDomainIdentifier, IA5String}
ContentType             ::= [APPLICATION 6] IMPLICIT INTEGER {p2(2)}
UAContentID             ::= [APPLICATION 10] IMPLICIT PrintableString
Priority                ::= [APPLICATION 7] IMPLICIT INTEGER {
                                normal(0), nonUrgent(1), urgent(2)}
PerMessageFlag          ::= [APPLICATION 8] IMPLICIT BIT STRING {
                                discloseRecipients(0),
                                conversionProhibited(1),
                                alternateRecipientAllowed(2),
                                contentReturnRequest(3)}
-- per-domain bilateral information
PerDomainBilateralInfo  ::= SEQUENCE {
                                CountryName,
                                AdministrationDomainName,
                                BilateralInfo}
```

FIGURE 16/X.411 (Part 1 of 4)

**Formal definition of User MPDU**

```
-- P1 DEFINITIONS continued --
BilateralInfo                ::= ANY
-- recipient information
RecipientInfo                ::= SET {
                                     recipient ORName,
                                     [0] IMPLICIT ExtensionIdentifier,
                                     [1] IMPLICIT PerRecipientFlag,
                                     [2] IMPLICIT ExplicitConversion OPTIONAL}
ExtensionIdentifier          ::= INTEGER
PerRecipientFlag             ::= BIT STRING -- see Figure 19/X.411
ExplicitConversion           ::= INTEGER {iA5TextTeletex(0), teletexTelex(1)}
-- trace information
TraceInformation             ::= [APPLICATION 9] IMPLICIT SEQUENCE OF
                                     SEQUENCE {GlobalDomainIdentifier, DomainSuppliedInfo}
DomainSuppliedInfo           ::= SET {
                                     arrival [0] IMPLICIT Time,
                                     deferred [1] IMPLICIT Time OPTIONAL,
                                     action [2] IMPLICIT INTEGER {relayed(0), rerouted(1)},
                                     converted EncodedInformationTypes OPTIONAL,
                                     previous GlobalDomainIdentifier OPTIONAL}
-- global domain identifier
GlobalDomainIdentifier       ::= [APPLICATION 3] IMPLICIT SEQUENCE {
                                     CountryName,
                                     AdministrationDomainName,
                                     PrivateDomainIdentifier OPTIONAL}
CountryName                  ::= [APPLICATION 1] CHOICE {
                                     NumericString,
                                     PrintableString}
AdministrationDomainName     ::= [APPLICATION 2] CHOICE {
                                     NumericString,
                                     PrintableString}
PrivateDomainIdentifier      ::= CHOICE {
                                     NumericString,
                                     PrintableString}
-- O/R name
ORName                       ::= [APPLICATION 0] IMPLICIT SEQUENCE {
                                     StandardAttributeList,
                                     DomainDefinedAttributeList OPTIONAL}
```

FIGURE 16/X.411 (Part 2 of 4)

**Formal definition of User MPDU**

```
-- P1 DEFINITIONS continued --
StandardAttributeList ::= SEQUENCE {
    CountryName OPTIONAL,
    AdministrationDomainName OPTIONAL,
    [0] IMPLICIT X121Address OPTIONAL,
    [1] IMPLICIT TerminalID OPTIONAL,
    [2] PrivateDomainName OPTIONAL,
    [3] IMPLICIT OrganizationName OPTIONAL,
    [4] IMPLICIT UniqueUAIdentifier OPTIONAL,
    [5] IMPLICIT PersonalName OPTIONAL,
    [6] IMPLICIT SEQUENCE OF OrganizationalUnit OPTIONAL}
DomainDefinedAttributeList ::= SEQUENCE OF DomainDefinedAttribute
DomainDefinedAttribute ::= SEQUENCE {
    type PrintableString,
    value PrintableString}
X121Address ::= NumericString
TerminalID ::= PrintableString
OrganizationName ::= PrintableString
UniqueUAIdentifier ::= NumericString
PersonalName ::= SET {
    surName [0] IMPLICIT PrintableString,
    givenName [1] IMPLICIT PrintableString OPTIONAL,
    initials [2] IMPLICIT PrintableString OPTIONAL,
    generationQualifier [3] IMPLICIT PrintableString OPTIONAL}
OrganizationalUnit ::= PrintableString
PrivateDomainName ::= CHOICE {
    NumericString,
    PrintableString}
-- encoded information types
EncodedInformationTypes ::= [APPLICATION 5] IMPLICIT SET {
    [0] IMPLICIT BIT STRING {
        undefined(0), tLX(1), iA5Text(2), g3Fax(3),
        tIF0(4), tTX(5), videotex(6), voice(7), sFD(8),
        tIF1(9)},
    [1] IMPLICIT G3NonBasicParams OPTIONAL,
    [2] IMPLICIT TeletexNonBasicParams OPTIONAL,
    [3] IMPLICIT PresentationCapabilities OPTIONAL
    -- other non-basic parameters are for further study --}
```

FIGURE 16/X.411 (Part 3 of 4)

**Formal definition of User MPDU**

```
-- P1 DEFINITIONS continued --
G3NonBasicParams            ::= BIT STRING {
                                    twoDimensional(8),
                                    fineResolution(9),
                                    unlimitedLength(20),
                                    b4Length(21),
                                    a3Width(22),
                                    b4Width(23),
                                    uncompressed(30)}
TeletexNonBasicParams       ::= SET {
                                    graphicCharacterSets [0] IMPLICIT T61String OPTIONAL,
                                    controlCharacterSets [1] IMPLICIT T61String OPTIONAL,
                                    pageFormats [2] IMPLICIT OCTET STRING OPTIONAL,
                                    miscTerminalCapabilities [3] IMPLICIT T61String OPTIONAL,
                                    privateUse [4] IMPLICIT OCTET STRING OPTIONAL}
PresentationCapabilities    ::= T73.PresentationCapabilities
```

FIGURE 16/X.411 (Part 4 of 4)

**Formal definition of User MPDU**

```
-- P1 DEFINITIONS continued --
DeliveryReportMPDU          ::= SEQUENCE {
                                    DeliveryReportEnvelope, DeliveryReportContent}
DeliveryReportEnvelope      ::= SET {
                                    report MPDUIdentifier,
                                    originator ORName,
                                    TraceInformation}
DeliveryReportContent       ::= SET {
                                    original MPDUIdentifier,
                                    intermediate TraceInformation OPTIONAL,
                                    UAContentId OPTIONAL,
                                    [0] IMPLICIT SEQUENCE OF ReportedRecipientInfo,
                                    returned [1] IMPLICIT UMPDUContent OPTIONAL,
                                    billingInformation [2] ANY OPTIONAL}
ReportedRecipientInfo       ::= SET {
                                    recipient [0] IMPLICIT ORName,
                                    [1] IMPLICIT ExtensionIdentifier,
                                    [2] IMPLICIT PerRecipientFlag,
                                    [3] IMPLICIT LastTraceInformation, intendedRecipient
                                    [4] IMPLICIT ORName OPTIONAL,
                                    [5] IMPLICIT SupplementaryInformation OPTIONAL}
-- last trace information
LastTraceInformation        ::= SET {
                                    arrival [0] IMPLICIT Time,
                                    converted EncodedInformationTypes OPTIONAL,
                                    [1] Report}
Report                      ::= CHOICE {
                                    [0] IMPLICIT DeliveredInfo,
                                    [1] IMPLICIT NonDeliveredInfo}
DeliveredInfo               ::= SET {
                                    delivery [0] IMPLICIT Time,
                                    typeOfUA [1] IMPLICIT INTEGER {
                                        public(0), private(1)}DEFAULT public}
NonDeliveredInfo            ::= SET {
                                    [0] IMPLICIT ReasonCode,
                                    [1] IMPLICIT DiagnosticCode OPTIONAL}
ReasonCode                  ::= INTEGER {
                                    transferFailure(0),
                                    unableToTransfer(1),
                                    conversionNotPerformed(2)}
```

FIGURE 17/X.411 (Part 1 of 2)

**Formal definition of Delivery Report MPDU**

```
-- P1 DEFINITIONS continued --
DiagnosticCode                  ::=  INTEGER {
                                         unrecognizedORName(0),
                                         ambiguousORName(1),
                                         mtaCongestion(2),
                                         loopDetected(3),
                                         uaUnavailable(4),
                                         maximumTimeExpired(5),
                                         encodedInformationTypesUnsupported(6),
                                         contentTooLong(7),
                                         conversionImpractical(8),
                                         conversionProhibited(9),
                                         implicitConversionNotRegistered(10),
                                         invalidParameters(11)}
-- supplementary information
SupplementaryInformation        ::=  PrintableString -- length limited and for further study --
```

FIGURE 17/X.411 (Part 2 of 2)

**Formal definition of Delivery Report MPDU**

```
-- P1 DEFINITIONS continued --
ProbeMPDU                       ::=  ProbeEnvelope
ProbeEnvelope                   ::=  SET {
                                         probe MPDUIdentifier,
                                         originator ORName,
                                         ContentType,
                                         UAContentID OPTIONAL,
                                         original EncodedInformationTypes OPTIONAL,
                                         TraceInformation,
                                         PerMessageFlag DEFAULT {},
                                         contentLength [0] IMPLICIT INTEGER OPTIONAL,
                                         [1] IMPLICIT SEQUENCE OF PerDomainBilateralInfo OPTIONAL,
                                         [2] IMPLICIT SEQUENCE OF RecipientInfo}
END -- of P1 DEFINITIONS --
```

FIGURE 18/X.411

**Formal definition of Probe MPDU**

Although the formal notation allows any combination of Standard Attributes and Domain Defined Attributes to be composed, only those combinations explicitly identified in Recommendation X.400 can be used to form a valid O/R name.

The allowed combinations are:

1) Telematic address:

   **X121Address, TerminalID** (optional)

2) Message handling address:

   a) **CountryName, AdministrationDomainName, UniqueUAIdentifier,** and, optionally, **DomainDefinedAttributes**

   b) **CountryName, AdministrationDomainName, X121Address,** and, optionally, **DomainDefinedAttributes**

   c) **CountryName, AdministrationDomainName** one or more selected from:

      **PersonalName, PrivateDomainName,**
      **OrganizationName, OrganizationalUnit**(s) and,
      optionally **DomainDefinedAttributes**

The country name is identified by means of a Numeric String whose value is equal to the X.121 country code, or a Printable String whose value is equal to the country code as specified in ISO 3166, ISO ALPHA-2 country code:

```
CountryName :: = [APPLICATION 1] CHOICE {
    NumericString,
    PrintableString}
```

An ADMD within a country is identified by means of a Printable String taken from a defined list administered by that country:

```
AdministrationDomainName :: = [APPLICATION 2] CHOICE {
    NumericString,
    PrintableString}
```

The X.121 address is constructed in accordance with Recommendation X.121:

```
X121Address :: = NumericString
```

The terminal identifier is, for example, a Telex Answer back or a Teletex terminal identifier:

```
TerminalID :: = PrintableString
```

A *private domain name* identifies a Private MD which is connected to an ADMD:

```
PrivateDomainName :: = CHOICE {
    NumericString,
    PrintableString}
```

```
Organization Name :: = PrintableString
```

A unique UA identifier uniquely identifies an originator or recipient within an ADMD.

```
UniqueUAIdentifier :: = NumericString
```

```
PersonalName :: = SET {
    surName [0] IMPLICIT PrintableString,
    givenName [1] IMPLICIT PrintableString OPTIONAL,
    initials [2] IMPLICIT PrintableString OPTIONAL,
    generationQualifier [3] IMPLICIT PrintableString OPTIONAL}
```

```
OrganizationalUnit :: = PrintableString
```

The Domain Defined Attribute List contains the type and value of any non-standard attributes defined by the originator or recipient's MD:

```
DomainDefinedAttribute List :: = SEQUENCE OF DomainDefinedAttribute
```

```
DomainDefinedAttribute :: = SEQUENCE {
    type PrintableString,
    value PrintableString}
```

3.4.1.2 *Extension identifier*

An *extension identifier* is a shorthand way of identifying an individual recipient within the context of a particular message. When combined with an MPDU identifier, it unambiguously identifies a copy of the original User or Probe MPDU in a way that is more convenient and reliable than using O/R names. Values are assigned from the range that begins with one and ends with the number of recipients of the message as submitted:

```
ExtensionIdentifier :: = INTEGER
```

3.4.1.3 *Global domain identifier*

A *global domain identifier* unambiguously identifies an MD within the MHS and is used for identifying the source of trace information, report MPDUs, etc. In the case of an ADMD, it consists of the country name and Administration Domain name of the MD. For a PRMD, it consists of the country name and Administration Domain name of the associated Administration MD, plus a private domain identifier. The *private domain identifier* is a unique identification of the Private MD within the scope of the associated Administration MD, and may be identical to the PRMD's Private Domain Name:

```
GlobalDomainIdentifier :: = [APPLICATION 3] IMPLICIT SEQUENCE {
    CountryName,
    AdministrationDomainName,
    PrivateDomainIdentifier OPTIONAL}

PrivateDomainIdentifier :: = CHOICE {
    NumericString,
    PrintableString}
```

3.4.1.4 *Encoded information types*

The *encoded information types* of a message are the kind(s) of information that appear in its content. Both the *basic* encoded information types present and any *non-basic* parameters that are required are specified:

```
EncodedInformationTypes :: = [APPLICATION 5] IMPLICIT SET {
    [0] IMPLICIT BIT STRING {
        undefined(0), tLX(1), iA5text(2), g3Fax(3), tIF0(4),
        tTX(5), videotex(6), voice(7), sFD(8), tIF1(9)},
    [1] IMPLICIT G3NonBasicParams OPTIONAL,
    [2] IMPLICIT TeletexNonBasicParams OPTIONAL,
    [3] IMPLICIT PresentationCapabilities OPTIONAL,
-- other non-basic parameters are for further study --}
```

The basic encoded information types are defined as follows. The undefined type is any other than those defined in the following. The Telex (tLX) type is defined in Recommendation F.1 [8]. The teleprinter type (iA5text) is defined in Recommendation T.50. The Group 3 facsimile (g3 Fax) type is defined in Recommendations T.4 and T.30. The Text Interchange Format 0 (tIF0) type is defined in Recommendations T.5, T.6 and T.73. The Teletex (tTX) type is defined in Recommendations F.200, T.61 and T.60. The videotex type is defined in Recommendations T.100 and T.101. The Simple Formattable Document (sFD) type is defined in Recommendation X.420. The Text Interchange Format 1 (tIF1) type is defined in Recommendation T.73.

The non-basic parameters for Group 3 facsimile correspond to the three- or four-octet Facsimile Information Field (FIF) conveyed by the Digital Command Signal (DCS) defined in Recommendation T.30:

```
G3NonBasicParams :: = BIT STRING {
    twoDimensional(8),
    fineResolution(9),
    unlimitedLength(20),
    b4Length(21),
    a3Width(22),
    b4Width(23),
    uncompressed(30)}
```

The non-basic parameters for Teletex specify optional graphic character sets, optional character sets, optional page formats, miscellaneous optional terminal capabilities, and a private-use parameter, which correspond to the parameters of the non-basic terminal capability conveyed by the Command Document Start (CDS) defined in Recommendation T.62:

```
TeletexNonBasicParams :: = SET {
    graphicCharacterSets [0] IMPLICIT T61String OPTIONAL,
    controlCharacterSets [1] IMPLICIT T61String OPTIONAL,
    pageFormats [2] IMPLICIT OCTET STRING OPTIONAL,
    miscTerminalCapabilities [3] IMPLICIT T61String OPTIONAL,
    privateUse [4] IMPLICIT OCTET STRING OPTIONAL}
```

Where non-basic parameters are indicated, these parameters represent the logical "OR" of the non-basic parameters of each instance of the encoded information type in the message content. Thus this parameter only serves to indicate whether there is a content type incompatibility, or whether conversion is required. If conversion is required, the content must be inspected to determine which non-basic parameters apply to any instance of the encoded information.

The non-basic parameters for the Text Interchange Format 0 and 1 specify optional resolution, optional graphic character sets, optional control character sets and so on, which correspond to the parameters of the presentation capabilities defined in Recommendation T.73:

```
PresentationCapabilities :: = T73.PresentationCapabilities
```

3.4.1.5 *Trace information*

*Trace information* documents the passage of an MPDU through the MTL. It describes the action(s) taken by each MD through which the MPDU passes.

Each element of trace information includes the global domain identifier (see § 3.4.1.3) of the MD supplying the information:

```
TraceInformation :: = [APPLICATION 9] IMPLICIT SEQUENCE OF
    SEQUENCE {GlobalDomainIdentifier, DomainSuppliedInfo}
```

Each element of trace information includes the date and time the MPDU arrived in the MD. In the originating MD's case, this is the submission time. Additionally, if Deferred Delivery caused the MD to hold the message for a period of time, the time when it started to process the message for delivery or forwarding is also included. The trace information also specifies the action the MD took with respect to the MPDU. *Relayed* is the normal action of a relay MTAE. *Rerouted* indicates that an attempt had previously been made to route the MPDU to the MD identified by the previous Global Domain Identifier. If the MD subjects a UMPDU to conversion, the result of the conversion is also included:

```
DomainSuppliedInfo :: = SET {
    arrival [0] IMPLICIT Time,
    deferred [1] IMPLICIT Time OPTIONAL,
    action [2] IMPLICIT INTEGER {relayed(0), rerouted (1)},
    converted EncodedInformationTypes OPTIONAL,
    previous GlobalDomainIdentifier OPTIONAL}
```

3.4.1.6 *MPDU identifier*

An *MPDU identifier* is used to distinguish MPDUs within the MTS. The originating MTA assigns a String identifier to each MPDU it creates, choosing it in such a way that it unambiguously identifies, within the originating MD, the particular message submission event, probe event, or delivery report generation that provoked the MPDU. When combined with the originating MD's global domain identifier, the String identifier forms the MPDU identifier, which unambiguously identifies the submission event, probe event, or delivery report generation throughout the entire MTS:

```
MPDUIdentifier :: = [APPLICATION 4] IMPLICIT SEQUENCE {GlobalDomainIdentifier, IA5String}
```

When a User or Probe MPDU is copied for routing to recipients reachable by different paths through the MTS (see § 3.3.1.1), each copy bears the MPDU identifier of the original. The copies can be distinguished from one another by means of the responsibility flags (see § 3.4.2.1) and the corresponding extension identifiers, which specify to which recipient(s) each copy is to be delivered.

This identifier is related, but not necessarily identical, to the event identifiers given to UAEs at the submit and delivery service access points.

3.4.1.7 *Content type*

A *content type* parameter is supplied by the originating UA and identifies the convention which governs the structure of the contents. The only defined value identifies the P2 protocol for Interpersonal Messaging, which is specified in Recommendation X.420.

**ContentType :: = [APPLICATION 6] IMPLICIT INTEGER {p2(2)}**

3.4.1.8 *UA content identifier*

The *UA content identifier* is provided by the UA and carried back to the originator (in a Delivery Report MPDU) by the MTL. This parameter is limited to 16 characters in length:

**UAContentID :: = [APPLICATION 10] IMPLICIT PrintableString**

3.4.1.9 *Time*

A Time identifies a calendar date and time of day to be conveyed in the protocol. It must always include the time of day in terms of UTC (Coordinated Universal Time), and may optionally convey the local time when it is generated. The precision of the Time is to either one second or one minute, as selected by the entity which generates it.

**Time :: = UTCTime**

3.4.2 *User MPDU*

A User MPDU, whose formal definition is summarized in Figure 16/X.411, carries a message toward one or more of its recipient UAEs. It has two parts, envelope and content:

**UserMPDU :: = SEQUENCE {UMPDUEnvelope, UMPDUContent}**

The *UMPDU envelope* contains the information the MTL requires to route the message to its intended recipients. The *UMPDU content* is the UA-supplied information the message is intended to convey.

3.4.2.1 *UMPDU envelope*

The UMPDU envelope contains a number of parameters. It is constructed when the message is submitted, and may be changed as the User MPDU makes its way through the MTL:

```
UMPDUEnvelope :: = SET {
    MPDUIdentifier,
    originator ORName,
    original EncodedInformationTypes OPTIONAL,
    ContentType,
    UAContentID OPTIONAL,
    Priority DEFAULT normal,
    PerMessageFlag DEFAULT {},
    deferredDelivery [0] IMPLICIT Time OPTIONAL,
    [1] IMPLICIT SEQUENCE OF PerDomainBilateralInfo OPTIONAL,
    [2] IMPLICIT SEQUENCE OF RecipientInfo,
    TraceInformation}
```

The **MPDUIdentifier** distinguishes the message submission event that produced this MPDU from all other message submission events, probe events, or delivery report generations (see § 3.4.1.6).

The **originator** parameter identifies the originator of the message carried by the MPDU (see § 3.4.1.1).

The **ContentType** parameter identifies the type of the message content (see § 3.4.1.7).

The **original EncodedInformationTypes** is the value of the SUBMIT.Request's encoded-info.-types parameter. The absence of this parameter indicates that the encoded information types are undefined (see § 3.4.1.4).

The **UAContentID** parameter allows the originator to supply an identifier for the content. This is not delivered to the recipient, but is returned to the originator with any Notify event.

The **deferredDelivery Time** parameter allows the function of deferred delivery to be supported by an MTAE other than the originating MTAE, provided that there is an agreement to do so.

The **Priority** parameter specifies the relative priority of the message carried by the MPDU:

**Priority :: = [APPLICATION 7] IMPLICIT INTEGER {normal(0), nonUrgent(1), urgent(2)}**

The **PerMessageFlag**, a Bit String, specifies options that apply to all recipients of the message. The **discloseRecipients** bit indicates whether the O/R names of all recipients should be indicated to each recipient UA when the message carried by the MPDU is delivered. The **conversionProhibited** bit indicates whether conversion prohibition was requested in the SUBMIT.Request primitive. If this bit is not set, implicit conversion may be carried out. The **alternateRecipientAllowed** bit indicates whether the Alternate Recipient Allowed service was requested in the SUBMIT.Request primitive. The **contentReturnRequest** bit indicates whether the content of the message is to be returned with any Non-Delivery Notification.

**PerMessageFlag :: = [APPLICATION 8] IMPLICIT BIT STRING {**
**discloseRecipients(0),**
**conversionProhibited(1),**
**alternateRecipientAllowed(2),**
**contentReturnRequest(3)}**

The **PerDomainBilateralInfo** is information supplied by the originating MD and intended for one of the MDs the message will reach as it passes through the MTL. More than one element of domain bilateral information may be present; each is addressed to a particular MD:

**PerDomainBilateralInfo :: = SEQUENCE {**
**CountryName,**
**AdministrationDomainName,**
**BilateralInfo}**

**BilateralInfo :: = ANY**

The **RecipientInfo** contains those pieces of envelope information that may vary from one recipient to another. The first two pieces of such information are the O/R name and extension identifier of the recipient to whom the rest of the information applies. Also specified is a **PerRecipientFlag**, described below:

**RecipientInfo :: = SET {**
**recipient ORName,**
**[0] IMPLICIT ExtensionIdentifier,**
**[1] IMPLICIT PerRecipientFlag,**
**[2] IMPLICIT ExplicitConversion OPTIONAL}**

**PerRecipientFlag :: = BIT STRING** – – *see Figure 19/X.411*

| Responsibility Flag (Bit 0) | |
|---|---|
| Off: | 0 |
| On: | 1 |

| Report Request (Bits 1-2) | |
|---|---|
| *Not used:* | 00 |
| Basic: | 01 |
| Confirmed: | 10 |
| Audit-and-confirmed: | 11 |

| User Report Request (Bits 3-4) | |
|---|---|
| No report: | 00 |
| Basic: | 01 |
| Confirmed: | 10 |
| *Not used:* | 11 |

*Note* – Bits 5 to 7 are reserved and must be zero.

FIGURE 19/X.411

**Structure of Per-recipient Flag**

Explicit Conversion indicates the type of conversion which is explicitly requested by the originator. Two types of conversion may be explicitly requested: IA5 Text to Teletex; and Teletex to Telex. Other types are **for further study**. Conversions are performed as specified in Recommendation X.408.

**ExplicitConversion :: = INTEGER {iA5TextTeletex(0), teletexTelex(1)}**

The bits of the **PerRecipientFlag** are assigned as shown in Figure 19/X.411. The *Report Request* bits, set by the originating MTA, request one of the following three kinds of reports:

**Basic**: a delivery report is returned only in case of non-delivery, and it contains only the last item of trace information.

**Confirmed**: a delivery report is returned in every case, and it contains only the last item of trace information.

**Audit-and-confirmed**: a delivery report is returned in every case, and it contains all of the trace information.

The *User Report Request* bits indicate the kind of report, if any, the originator selected; the value $00_2$ indicates that he requested the Prevention of Non-delivery Notification service. The Report Request bits must specify at least the report level requested by the originator. If the Responsibility flag is set, the receiving MTA shall have the responsibility to either deliver the message carried by the MPDU to this particular recipient, or to relay it to another MTA for subsequent delivery.

The **TraceInformation** records information about the MPDU's passage through the MTL (see § 3.4.1.5).

### 3.4.2.2 *UMPDU content*

The UMPDU content is the UA-supplied information the message is intended to convey. Except when conversion is requested, the UMPDU content is not modified by the MTL but rather is passed transparently through it:

```
UMPDUContent :: = OCTET STRING
```

### 3.4.3 *Delivery report MPDU*

A Delivery report MPDU, whose formal definition is summarized in Figure 17/X.411, carries a delivery or non-delivery report toward a message's originator. It has two parts, envelope and content:

```
DeliveryReportMPDU :: = SEQUENCE {
    DeliveryReportEnvelope, DeliveryReportContent}
```

The *delivery report envelope* contains the information the MTL requires to route the report to the originator. The *delivery report content* is the system-supplied information the report is intended to convey. The Delivery Report may optionally also contain a parameter which is used to return billing information to the originating Administration Domain. An indication of whether the message was delivered to Administration-owned or privately owned equipment is also returned.

### 3.4.3.1 *Delivery report envelope*

The delivery report envelope contains a number of parameters. It is constructed when the report is originated, and may be changed as the Delivery Report MPDU makes its way through the MTL:

```
DeliveryReportEnvelope :: = SET {
    report MPDUIdentifier,
    originator ORName,
    TraceInformation}
```

The **report MPDUIdentifier** distinguishes this MPDU (not the *subject* UMPDU, whose fate is being reported) from all other MPDUs (see § 3.4.1.6).

The **originator** parameter identifies the originator of the subject message (see § 3.4.1.1).

The **TraceInformation** records information about the passage of this MPDU (not the subject UMPDU) through the MTL (see § 3.4.1.5).

3.4.3.2 *Delivery report content*

The delivery report content is the system-supplied information the report is intended to convey. It includes the **MPDUIdentifier** of the subject UMPDU. If an audit-and-confirmed delivery report was requested, it also includes the trace information present in the subject UMPDU when it reached the MD generating the report. If a content-returned non-delivery report was requested, the content of the report also includes the content of the subject UMPDU:

```
DeliveryReportcontent :: = SET {
    original MPDUIdentifier,
    intermediate TraceInformation OPTIONAL,
    UAContentID OPTIONAL,
    [0] IMPLICIT SEQUENCE OF ReportedRecipientInfo,
    returned [1] IMPLICIT UMPDUContent OPTIONAL,
    billingInformation [2] ANY OPTIONAL}
```

*Note* – The specification of the contents of the **billingInformation** parameter is by bilateral agreement.

The **ReportedRecipientInfo** is those pieces of information that may vary from one recipient to another. The first two pieces of such information are the O/R name and extension identifier of the recipient to whom the rest of the information applies:

```
ReportedRecipientInfo :: = SET {
    recipient [0] IMPLICIT ORName,
    [1] IMPLICIT ExtensionIdentifier,
    [2] IMPLICIT PerRecipientFlag,
    [3] IMPLICIT LastTraceInformation,
    intendedRecipient [4] IMPLICIT ORName OPTIONAL,
    [5] IMPLICIT SupplementaryInformation OPTIONAL}
```

The **PerRecipientFlag** specifies the report requested by the originator (see § 3.4.2.1).

The **LastTraceInformation** indicates the fate of the subject message. The information provided includes the date and time at which the subject UMPDU entered the MD making the report and, if the message underwent conversion, the result of that conversion:

```
LastTraceInformation :: = SET {
    arrival [0] IMPLICIT Time,
    converted EncodedInformationTypes OPTIONAL,
    [1] Report}
```

If delivery succeeded, the date and time at which the message was delivered to the recipient UA as well as an indication of whether the UA is owned by an Administration (public) or not (private) are specified. If delivery failed, the reason and some diagnostic information is given:

```
Report :: = CHOICE {
    [0] IMPLICIT DeliveredInfo,
    [1] IMPLICIT NonDeliveredInfo}

DeliveredInfo :: = SET {
    delivery [0] IMPLICIT Time,
    typeOfUA [1] IMPLICIT INTEGER {public(0), private(1)}DEFAULT public}

NonDeliveredInfo :: = SET {
    [0] IMPLICIT ReasonCode,
    [1] IMPLICIT DiagnosticCode OPTIONAL}
```

Transfer Failure indicates that, while the MTA was attempting to relay or deliver the message, some communication failure prevented it. Unable to Transfer indicates that, due to some problem with the message itself, the MTA could not relay or deliver it. ConversionNotPerformed indicates that a conversion necessary for the delivery of the message was unable to be performed. Further information on the nature of the problem is contained in the Diagnostic Code:

```
ReasonCode :: = INTEGER {
    transferFailure(0),
    unableToTransfer(1),
    conversionNotPerformed(2)}
```

```
DiagnosticCode :: = INTEGER {
    unrecognizedORName(0),
    ambiguousORName(1),
    mtaCongestion(2),
    loopDetected(3),
    uaUnavailable(4),
    maximumTimeExpired(5),
    encodedInformationTypesUnsupported(6),
    contentTooLong(7),
    conversionImpractical(8),
    conversionProhibited(9),
    implicitConversionNotRegistered(10),
    invalidParameters(11)}
```

Other reasons and diagnostic codes are for further study.

The **intendedRecipient** parameter is present if the subject message was diverted to an alternate recipient (see § 3.4.1.1).

The **SupplementaryInformation** is included for use by the Teletex Access Unit (see Recommendation X.430) and for possible use by Teletex/Telex conversion facilities. It is of limited length, but the exact limit is for further study. It may be used for Received Answer-back, Telex Transmission Duration, or Note and Received Recorded Message:

```
SupplementaryInformation :: = PrintableString
```

### 3.4.4 *Probe MPDU*

A Probe MPDU, whose formal definition is summarized in Figure 18/X.411, carries a request for information about the deliverability of a mesage toward one or more potential recipient UAs. It has just one part, the envelope:

```
ProbeMPDU :: = ProbeEnvelope
```

The *probe envelope* contains the information the MTL requires to route the probe to its destination.

An MTAE may generate a Probe MPDU either to establish that a particular, *subject* message can be delivered before sending it or to test in a general way the availability and capabilities of other MDs.

Under what conditions an originating MTAE may issue a Probe is for further study. To be considered are account settlement and trade-offs between storage and communication costs, including system and circuit dimensioning.

#### 3.4.4.1 *Probe envelope*

The probe envelope contains a number of parameters. It is constructed when the probe is issued, and may be changed as the Probe MPDU makes its way through the MTL:

```
ProbeEnvelope :: = SET {
    probe MPDUIdentifier,
    originator ORName,
    ContentType,
    UAContentID OPTIONAL,
    original EncodedInformationTypes OPTIONAL,
    TraceInformation,
    PerMessageFlag DEFAULT {},
    contentLength [0] IMPLICIT INTEGER OPTIONAL,
    [1] IMPLICIT SEQUENCE OF PerDomainBilateralInfo OPTIONAL,
    [2] IMPLICIT SEQUENCE OF RecipientInfo}
```

The **probe MPDUIdentifier** distinguishes the Probe event that resulted in the creation of this MPDU from all other probe events, submission events, and delivery report generations (see § 3.4.1.6).

The **originator** parameter identifies the originator of the subject message (see § 3.4.1.1).

The **original EncodedInformationTypes** is the encoded information types of the subject message. The absence of this parameter indicates that those types are undefined (see § 3.4.1.4).

The **TraceInformation** records information about this MPDU's passage through the MTL (see § 3.4.1.5).

The **PerMessageFlag** is as specified for the User MPDU (see § 3.4.2.1). However, only the **conversionProhibited** and **alternateRecipientAllowed** flags affect the deliverability of the message as tested by Probe.

The **PerDomainBilateralInfo** is as specified for the User MPDU (see § 3.4.2.1). It is included so that the effects of bilaterally agreed services, such as Closed User Group and Security Classification, can be checked before actual submission.

The **contentLength** is the length, in octets, of the content of the subject message.

The **RecipientInfo** is identical to that appearing in the UMPDU envelope (see § 3.4.2.1). It contains those pieces of envelope information that may vary from one recipient to another. The first two pieces of such information are the O/R name and extension identifier of the recipient to whom the rest of the information applies. Also specified is whether the current MTA has responsibility for probing delivery to this particular recipient, and the kind of delivery report (if any) requested by the user and by the originating MD.

## 3.5 *Management of associations*

### 3.5.1 *Establishing and releasing associations*

Associations between two MTAEs are created in accordance with bilateral agreements covering the following:

1) The maximum number of associations that may exist simultaneously.
2) Whether one-way or two-way alternate associations are used.
3) Which MTA has responsibility for establishing the associations.
4) Whether associations are permanently established or established and released as required.

If associations may be established as required, their establishment is a local decision based upon, for example, the time of day or the number and priority of the MPDUs awaiting transfer. The specific criteria used will reflect the Quality of Service bounds for message transfer (for example, the average delivery time). If an association may be released when no longer required, it need not be released as soon as there are no more MPDUs to transfer; rather, it may be left idle for a period of time in case further MPDUs arrive.

Associations are established by invoking the RTS' OPEN.Request primitive with the following parameters:

| **Parameter name** | **Parameter value** |
|---|---|
| Responder-address | The session address of the RTS of corresponding MTAE |
| Dialogue-mode | Monologue or two-way alternate, according to bilateral agreement |
| Initial-turn | With the initiator if, and only if, it has MPDUs to send (this only applies to two-way alternate associations) |
| Application-protocol | P1 |
| User-data | **CHOICE** {<br>**NULL,** – – *if no validation is required*<br>**[1] IMPLICIT SET** {<br>**mTAName [0] IMPLICIT IA5String,**<br>**password [1] ANY**}} |

The **mTAName** and **password** are omitted if some other method of validation is used (for example, calling party network address or a permanent circuit). Otherwise, the validation parameters are checked and an OPEN.Response issued with the following parameters:

| **Parameter name** | **Parameter value** |
|---|---|
| Disposition | Accepted or refused |
| User-data | If the association was accepted, the same as above |
| Refusal-reason | Unacceptable dialogue mode, Busy, Validation Failure. Other reasons are for further study |

Associations are released by invoking the CLOSE.Request primitive, which has no parameters.

3.5.2 *Transferring MPDUs*

Each MPDU is transferred as described in Section 3.3 by invoking the RTS' TRANSFER.Request primitive with the following parameters:

| **Parameter name** | **Parameter value** |
|---|---|
| APDU | The MPDU to be transferred |
| Transfer-time | Determined by local rule |

The following are the rules for assigning MPDUs to associations:

1) Each MPDU is assigned a priority based upon its type, and UPMDU priority in accordance with Table 1/X.411.
2) MPDUs are assigned to associations in accordance with their priorities.
3) Several associations may be used to carry MPDUs of the same priority.
4) On any one association, higher priority MPDUs are sent first.
5) On any one association, messages of the same priority are sent first-in-first-out (FIFO).

TABLE 1/X.411

**MPDU priorities**

| Priority | MPDU Type | MPDU Priority |
|---|---|---|
| 1 | SMPDU, UMPDU | Urgent |
| 2 | UMPDU | Normal |
| 3 | UMPDU | Non-urgent |

3.5.3 *Managing the turn*

If the association is two-way alternate, the Message Dispatcher without the turn may issue a TURN-PLEASE.Request primitive whose priority parameter reflects the highest priority MPDU awaiting transfer. When the Message Dispatcher requests the turn so that it can issue a CLOSE.Request primitive, the priority parameter should reflect the lowest priority MPDU the MTA is prepared to receive before releasing the association.

The Message Dispatcher that has the turn will issue a TURN-GIVE.Request primitive in response to a TURN-PLEASE.Indication when it has no MPDUs to transfer whose priorities are equal to or higher than that indicated in the TURN-PLEASE.Indication.

3.6 *Aspects relating to ADMD to PRMD connections*

The following restrictions apply to the use of the Message Transfer Protocol between a PRMD and an ADMD:

1) A PRMD will not act as a relay between two ADMDs. Therefore, a private domain identifier may only appear in the first or last item in a sequence of trace information.
2) On receipt of any instance of the trace information or MPDU identifier from a PRMD, the ADMD will check that it contains the correct global domain identifier (including private domain identifier).
3) The ADMD will Validate the Originator O/R-name on messages received from a PRMD as far as possible.

## 4 Submission and Delivery Protocol (P3)

### 4.1 *Introduction*

As specified in Recommendation X.400, the MTL comprises two kinds of entities, SDEs and MTAEs. This section describes the *Submission and Delivery Protocol (P3)*, the peer protocol that governs communication between an SDE and an MTAE in order to provide a UAE with the MTL services defined in § 2 of this Recommendation.

P3 makes use of the Reliable Transfer Server (RTS), defined in Recommendation X.410. The RTS is responsible for supporting associations between the SDE and MTAE, and for completely and reliably transferring protocol data units by means of those associations. For that purpose the RTS makes use of the Standard OSI protocols. P3 also makes use of the procedures and formats for remote operations, also defined in Recommendation X.410.

§ 4.2 reviews the concept of remote operations. § 4.3 specifies the remote operations of P3. § 4.4 describes how associations are managed in P3.

### 4.2 *Review of the remote operation protocol*

The purpose of P3 is to enable a UA that is remote from its MTA to obtain access to the services of the MTL. Such access includes the transfer of messages and responsibility for them from the SDE to the MTAE during submission, and from the MTAE to the SDE during delivery. The SDE and MTAE accomplish these and other objectives by invoking remote operations at their peers. (Where no ambiguity results, the adjective "remote" is omitted.)

The unit of exchange between an SDE and an MTAE is an Operation Protocol Data Unit (OPDU), which contains the information required to either invoke an operation or report the outcome of such an invocation. OPDUs are transferred by means of the RTS, which uses session services to prevent loss of all or any part of an OPDU. OPDUs are defined in Recommendation X.410, and their transfer by means of the RTS is described in § 4.4 below.

When an operation is invoked, an argument is conveyed to the peer entity. For the submission and delivery operations, for example, the argument includes the message itself. Some operations return results if they succeed and report errors if they fail. The invocation of each such operation causes one of the following to be returned to the invoker:

1) A Reject OPDU, indicating that the invocation failed, for example, because of a syntax error in the argument.
2) A ReturnError OPDU, indicating that the operation failed.
3) A ReturnResult OPDU, indicating that the operation succeeded.

An identifier is assigned to each invocation of an operation. This identifier is returned with any outcome report so that the outcome can be correlated with the invocation.

### 4.3 *Specification of SDE and MTAE operations*

The Submission and Delivery Protocol comprises the remote operations listed below and summarized in Table 2/X.411. The MTL service primitives to which each operation relates are also noted below:

Three *management* operations allow the SDE and MTAE to exchange information required to control the dialouge between them:

a) The *Register* operation allows the SDE and MTAE to exchange parameters specifying their current registration capabilities. This corresponds to REGISTER.Request/Confirmation.

b) The *Control* operation allows the SDE or MTAE to restrict the other's ability to invoke operations. This corresponds to CONTROL.Request/Confirmation/Response/Indication.

c) The *Change-Password* operation allows either the UA or the MTA to change the current value of its password.

*Note* – The LOGON and LOGOFF service elements of the MTL are not supported by Remote Operations, but are mapped directly onto RTS OPEN and CLOSE service primitives.

Three *submission* operations allow the SDE to submit messages with or without content to the MTAE:

a) The *Submit* operation allows the SDE to submit a complete message to the MTAE. This corresponds to SUBMIT.Request/Confirmation.

b) The *Probe* operation allows the SDE to submit a limited envelope to verify its delivery parameters. This corresponds to PROBE.Request/Confirmation.

c) The *Cancel* operation allows the SDE to request a cancellation of a previously successfully submitted Deferred Delivery message. This corresponds to CANCEL.Request/Confirmation.

The *Deliver* operation allows the MTAE to deliver a message to the SDE. This corresponds to DELIVER.Indication.

The *Notify* operation allows the MTAE to transfer delivery and non-delivery notifications to the SDE. This corresponds to NOTIFY.Indication.

The operations listed above are formally defined in Figure 20/X.411. The formal definitions specify, among other things, the data types of the operations' arguments and results. Wherever possible, data types defined in P1 are used in the definition of P3. Thus the data types used but not defined here (for example, **EncodedInformationTypes**) are to be found in § 3. In addition, P3 data types are designed to exhibit maximum commonality with the corresponding P1 data types. For example, **SubmitEnvelope** is structured much like **UMPDUEnvelope**.

TABLE 2/X.411

**SDE and MTAE operations**

| Operation | Argument | Outcome | Information returned with result |
|---|---|---|---|
| Register | Information about deliverable messages, default controls, and UA parameter settings | Result or error | *None* |
| Control | Operations and messages to be held | Result or error | Information about held operations and messages |
| Change-Password | Old and new passwords | Result or error | *None* |
| Submit | Envelope and content | Result or error | Event ID, time, and content identifier |
| Probe | Envelope | Result or error | Event ID, time, and content identifier |
| Cancel | Event ID | Result or error | *None* |
| Deliver | Envelope, content, event ID and time | Error | *Not Applicable* |
| Notify | Delivery or non-delivery notification | Error | *Not Applicable* |

```
P3 DEFINITIONS :: =
BEGIN
-- The following types, used in P3 DEFINITIONS are defined in P1 (Section 3).
    -- P1.ORName                    P1.UMPDUContent
    -- P1.Priority                  P1.ContentType
    -- P1.EncodedInformationTypes   P1.Report
    -- P1.UAContentID               P1.SupplementaryInformation
    -- P1.PerMessageFlag            P1.X121Address
    -- P1.ExplicitConversion        P1.Time
-- the following macros are used, as defined in X.410
OPERATION                           :: = RemoteOperations.OPERATION
ERROR                               :: = RemoteOperations.ERROR
-- operations
register        OPERATION -- invoked by SDE
                ARGUMENT SET {
                    deliverable P1.EncodedInformationTypes OPTIONAL,
                    P1.ORName OPTIONAL,
                    [0] Address OPTIONAL,
                    maximumContentLength [1] INTEGER OPTIONAL, -- in octets
                    [2] DefaultControlSettings OPTIONAL}
                RESULT NULL
                ERRORS {registerRejection}
                :: = 1
control         OPERATION -- invoked by either SDE or MTAE
                ARGUMENT SET {
                    restrict [0] IMPLICIT BOOLEAN DEFAULT TRUE,
                        -- if FALSE, all restrictions are removed
                        -- if TRUE, any other set members are used to update
                        -- the prevailing restrictions
                    permissible [1] IMPLICIT OperationTypes OPTIONAL,
                    lowest P1.Priority OPTIONAL,
                    permissible P1.EncodedInformationTypes OPTIONAL,
                    maximumContentLength [2] IMPLICIT INTEGER OPTIONAL -- in octets --}
                RESULT SET {
                    held [0] IMPLICIT OperationTypes OPTIONAL,
                    messagesHeld [1] IMPLICIT BIT STRING {
                        longContent(0), lowPriority(1)} OPTIONAL,
                    held P1.EncodedInformationTypes OPTIONAL}
                ERRORS {serviceElementNotSubscribed} -- not generated by SDE --
                :: = 2
```

FIGURE 20/X.411 (Part 1 of 5)

**Formal definition of P3 operations**

```
-- P3 DEFINITIONS continued --
changePassword  OPERATION -- invoked by SDE or MTAE
                ARGUMENT SET {
                    oldPassword [0] ANY,
                    newPassword [1] ANY}
                RESULT NULL
                ERRORS {
                    oldPasswordWrong, newPasswordNotAcceptable}
                :: = 8
submit          OPERATION -- invoked by SDE
                ARGUMENT SEQUENCE {
                    SubmitEnvelope,
                    Content}
                RESULT SET {
                    submit EventID,
                    submission [0] IMPLICIT P1.Time,
                    P1.UAContentId OPTIONAL}
                ERRORS {
                    controlViolation, originatorInvalid,
                    recipientImproperlySpecified, serviceElementNotSubscribed}
                :: = 3
probe           OPERATION -- invoked by SDE
                ARGUMENT ProbeEnvelope
                RESULT SET {
                    probe EventID,
                    probe [0] IMPLICIT P1.Time,
                    P1.UAContentID OPTIONAL}
                ERRORS {
                    controlViolation, originatorInvalid,
                    recipientImproperlySpecified, serviceElementNotSubscribed}
                :: = 4
cancel          OPERATION -- invoked by SDE
                ARGUMENT submit EventID
                RESULT NULL
                ERRORS {
                    invalidEventID, messageDelivered, messageTransferred}
                :: = 7
deliver         OPERATION -- invoked by MTAE
                ARGUMENT SEQUENCE {
                    deliver EventID,
                    delivery P1.Time,
                    DeliverEnvelope,
                    Content}
                ERRORS {controlViolation}
                :: = 5
```

FIGURE 20/X.411 (Part 2 of 5)

**Formal definition of P3 operations**

```
-- P3 DEFINITIONS continued --
notify              OPERATION -- invoked by MTAE
                    ARGUMENT SET {
                        submitOrProbe EventID,
                        P1.UAContentID OPTIONAL,
                        SEQUENCE OF NotifyRecipientInfo,
                        returned [0] IMPLICIT Content OPTIONAL}
                    ERRORS {controlViolation}
                    :: = 6
-- errors
controlViolation                ERROR
                                PARAMETER NULL
                                :: = 1
originatorInvalid               ERROR
                                PARAMETER NULL
                                :: = 2
recipientImproperlySpecified    ERROR
                                PARAMETER SEQUENCE OF P1.ORName
                                :: = 3
serviceElementNotSubscribed     ERROR
                                PARAMETER NULL
                                :: = 4
oldPasswordWrong                ERROR
                                PARAMETER NULL
                                :: = 5
newPasswordNotAcceptable        ERROR
                                PARAMETER NULL
                                :: = 6
invalidEventID                  ERROR
                                PARAMETER NULL
                                :: = 7
messageDelivered                ERROR
                                PARAMETER NULL
                                :: = 8
messageTransferred              ERROR
                                PARAMETER NULL
                                :: = 9
registerRejection               ERROR
                                PARAMETER NULL
                                :: = 10
```

FIGURE 20/X.411 (Part 3 of 5)

**Formal definition of P3 operations**

```
-- P3 DEFINITIONS continued --
-- types
Address                 ::= CHOICE { -- other forms are for further study
                                [0] IMPLICIT SEQUENCE {
                                P1.X121Address OPTIONAL,
                                tSAP-ID PrintableString OPTIONAL -- up to 16 digits --}}
DefaultControlSettings  ::= SET {
                                permissible [0] IMPLICIT OperationTypes OPTIONAL,
                                lowest P1.Priority OPTIONAL,
                                permissible P1.EncodedInformationTypes OPTIONAL,
                                maximumContentLength [1] IMPLICIT INTEGER OPTIONAL, -- in octets
                                restrict [2] IMPLICIT BOOLEAN DEFAULT TRUE
                                    -- if FALSE, there are no restrictions other than
                                    --      the registered parameters
                                    -- if TRUE, the other set members, if any, are
                                    --      used as default control settings -- }
OperationTypes          ::= BIT STRING {probeOrNotify(0), submitOrDeliver(1)}
SubmitEnvelope          ::= SET {
                                originator P1.ORName,
                                P1.EncodedInformationTypes,
                                P1.ContentType,
                                P1.UAContentID OPTIONAL,
                                deferredDelivery [0] IMPLICIT Time OPTIONAL,
                                P1.Priority DEFAULT normal,
                                P1.PerMessageFlag DEFAULT {},
                                [1] IMPLICIT SEQUENCE OF SubmitRecipientInfo}
SubmitRecipientInfo     ::= SET {
                                recipient P1.ORName,
                                [0] IMPLICIT UsersReportRequest,
                                [1] IMPLICIT P1.ExplicitConversion}
UsersReportRequest      ::= BIT STRING {confirmed(3), basic(4)}
Content                 ::= P1.UMPDUContent
EventID                 ::= P1.MPDUIdentifier
ProbeEnvelope           ::= SET {
                                originator P1.ORName,
                                P1.EncodedInformationTypes,
                                P1.ContentType,
                                P1.UAContentID OPTIONAL,
                                contentLength [0] IMPLICIT INTEGER OPTIONAL,
                                P1.PerMessageFlag DEFAULT {},
                                [3] IMPLICIT SEQUENCE OF SubmitRecipientInfo}
```

FIGURE 20/X.411 (Part 4 of 5)

**Formal definition of P3 operations**

```
-- P3 DEFINITIONS continued --

DeliverEnvelope          ::= SET {
                                 [0] IMPLICIT P1.ContentType,
                                 originator P1.ORName,
                                 original [1] IMPLICIT P1.EncodedInformationTypes,
                                 P1.Priority DEFAULT normal,
                                 [2] IMPLICIT DeliveryFlags,
                                 otherRecipients [3] IMPLICIT SEQUENCE OF P1.ORName OPTIONAL,
                                 thisRecipient [4] IMPLICIT P1.ORName,
                                 intendedRecipient [5] IMPLICIT P1.ORName OPTIONAL,
                                 converted [6] IMPLICIT P1.EncodedInformationTypes OPTIONAL,
                                 submission [7] IMPLICIT P1.Time}

NotifyRecipientInfo      ::= SET {
                                 recipient [0] IMPLICIT P1.ORName,
                                 [1] P1.Report,
                                 converted P1.EncodedInformationTypes OPTIONAL,
                                 intendedRecipient [2] IMPLICIT P1.ORName OPTIONAL,
                                 [3] IMPLICIT P1.SupplementaryInformation DEFAULT NULL}

DeliveryFlags            ::= BIT STRING {conversionProhibited(1)}

END -- of P3 DEFINITIONS --
```

FIGURE 20/X.411 (Part 5 of 5)

**Formal definition of P3 operations**

### 4.3.1 *Management operations*

The management of the P3 interaction between an SDE and an MTAE is accomplished by means of the Register, Control, and Change-Password operations.

#### 4.3.1.1 *Register operation*

The Register operation allows the UA to request long-term changes to its registration parameters. The deliverable encoded information types, the maximum content length, the UA's network address, the UA's O/R Name, and the default control settings can be altered. The latter allows the UA to specify what control restrictions will be in force whenever an association is established. These restrictions must be a subset of those implied by the other parameters of Register.

The following data types used in the Register operation correspond to the similarly named parameters of the REGISTER primitives (see § 2.2.4):

**deliverable EncodedInformationTypes**
**ORName**
**Address**
**maximumContentLength**
**DefaultControlSettings**

The outcome of the Register operation corresponds to the success-indication parameter of the REGISTER.Confirmation primitive. Where the Register operation returns an error, the particular error returned conveys the Failure-reason parameter of the REGISTER.Confirmation primitive.

*Note* – The procedures for performing the Alternate Recipient Assignment service element are not specified in the P3 protocol and are left to the Administration providing the service.

#### 4.3.1.2 *Control operation*

The Control operation allows the MTAE or SDE to temporarily restrict the operations that may be invoked by the other. Among the restrictions that can be imposed are the following:

1) Deliver or submission is permitted.
2) Messages whose contents' lengths exceed a specified value may not be transferred.
3) Messages whose priorities are below a specified value may not be transferred.
4) Only messages with specified encoded information types may be transferred.
5) Notifications or probes may be transferred.

The following data types used in the Control operation correspond to the similarly named parameters of the CONTROL primitives (see §§ 2.2.5 and 2.2.6):

**lowest Priority**
**permissible EncodedInformationTypes**
**maximumContentLength**

The **permissible Operation Types** data type is used to convey the information described in relation to the Msgs, Notifications, and Probes parameters of CONTROL.

The result of the Control operation corresponds to the Messages-held parameter of the CONTROL primitives.

The outcome of the Control operation corresponds to the success-indication paramater of the CONTROL.Confirmation primitive. Where the Control Operation returns an error, the particular error returned conveys the Failure-reason parameter of the CONTROL.Confirmation primitive.

The **restrict** data type if set to *false*, removes all restrictions currently in force by the invoker of the Control operation. If set to *true*, the restrictions being invoked are contained in the remaining parameter.

The error **serviceElementNotSubscribed** may be returned by the MTAE when it is unable to honour a "Control" operation because the user has not subscribed to the "Hold for Delivery" service element. The SDE will never generate this error and must obey Control restrictions imposed by the MTAE.

#### 4.3.1.3 *Change-Password operation*

The Change-Password operation allows either the UAE or MTAE to change its current password. No operation involving passwords should be invoked after invoking a Change-Password and before receiving the result.

The following data types used in the Change-Password operation correspond to the similarly named parameters of the CHANGE-PASSWORD primitives (see §§ 2.2.12 and 2.2.13):

**oldPassword**
**newPassword**

The outcome of the Change-Password operation corresponds to the success-indication parameter of the CHANGE-PASSWORD primitives. Where the Change-Password operation returns an error, the particular error returned conveys the Failure-reason parameter of the CHANGE-PASSWORD primitives.

### 4.3.2 *Submission operations*

The Submit and Probe operations convey the information necessary to request the various service elements of the MTL's message submission service.

4.3.2.1 *Submit operation*

The Submit operation contains both message envelope parameters and the message's content. The MTAE accepts or rejects as a whole the message sent by the SDE. Any intermediate checkpoints generated during the transfer are the responsibility of lower layers and are not visible in P3.

The MTAE indicates its acceptance or rejection of the Submit operation (and thus of the message) by means of the result, error, or reject return. If the operation succeeds, its result is the submission identifier and time. If the operation fails, the error encountered is specified.

The following data types used in the Submit operation correspond to the similarly named parameter of the SUBMIT primitives (see § 2.2.7):

**originator ORName**
**recipient ORName**
**Content**
**ContentType**
**EncodedInformationTypes**
**Priority**
**deferredDelivery Time**
**UAContentID**
**ExplicitConversion**
**submit EventID**
**submission Time**

Within the **PerMessageFlag** data type used in the Submit operation, the following named bits correspond to the similarly named parameters of the SUBMIT primitives:

**conversionProhibited**
**discloseRecipients**
**alternateRecipientAllowed**
**contentReturnRequest**

The **UsersReportRequest** data type corresponds to the NDN-suppress and Delivery-notice parameters of the SUBMIT.Request primitive. It has the value {} where NDN-suppress is requested, {**confirmed**} where Delivery-notice is requested, and {**basic**} where neither is requested.

The outcome of the Submit operation corresponds to the Success-indication parameter of the SUBMIT.Confirmation primitive. Where the Submit operation returns an error, the particular error returned conveys the Failure-reason parameter of the SUBMIT.Confirmation primitive.

4.3.2.2 *Probe operation*

The Probe operation allows the SDE to send an abbreviated message containing only some of the envelope parameters. If the probe is accepted, the MTAE will construct a probe MPDU with the parameters provided.

The MTAE indicates its acceptance or rejection of the Probe operation (and thus of the envelope) by means of the result, error, or reject return. If the operation succeeds, its result is the probe identifier and time. If the operation fails, the error encountered is specified.

The following data types used in the Probe operation correspond to the similarly named parameters of the PROBE primitives (see § 2.2.8):

**originator ORName**
**recipient ORName**
**ContentType**
**EncodedInformationTypes**
**ExplicitConversion**
**UAContentID**
**contentLength**
**probe EventID**
**probe Time**

Within the **PerMessageFlag** data type used in the Probe operation, the following named bits correspond to the similarly named parameters of the PROBE primitives:

**conversionProhibited**
**alternateRecipientAllowed**

Although the **UsersReportRequest** is included, there is no corresponding parameter of the service primitive. It always has the value {**confirmed**}.

The outcome of the Probe operation corresponds to the Success-indication parameter of the PROBE.Confirmation primitive. Where the Probe operation returns an error, the particular error returned conveys the Failure-reason parameter of the PROBE.Confirmation primitive.

#### 4.3.2.3 *Cancel operation*

The Cancel operation allows the SDE to request that the MTAE stop delivery or transfer of a previously Submitted deferred delivery message. The **EventID** is the **submit EventID** that corresponds to that supplied by the MTAE in the Submit RESULT generated by the Submit operation requesting deferred delivery.

The data type **submit EventID** used in the Cancel operation corresponds to the similarly named parameter of the CANCEL.Request primitive (see Section 2.2.11). A **NULL** result indicates that the Cancel operation was successful. The errors defined correspond to the Failure-reason parameter of the CANCEL.Confirmation primitive.

### 4.3.3 *Deliver operation*

The MTAE initiates the delivery of an arrived message provided the UA has not, by means of a previous Control operation, indicated its unwillingness to receive messages of its length and priority.

The argument of the Deliver operation is the message to be delivered. The acknowledgement of successful delivery (including the attendant transfer of responsibility) is provided by the Session Layer by means of the completion of the activity carrying the protocol data unit that invoked the Deliver operation. Thus the outcome of this operation need not be explicitly reported.

The following data types used in the Deliver operation correspond to the similarly named parameters of the DELIVER.Indication primitive (see § 2.2.9):

**deliver EventID**
**delivery Time**
**Content**
**ContentType**
**originator ORName**
**original EncodedInformationTypes**
**Priority**
**otherRecipients** (equivalent to Other-recipient-O/R-names)
**thisRecipient ORName**
**intendedRecipient ORName**
**converted EncodedInformationTypes**
**submission Time**

The data type **DeliveryFlags** is used to convey the information described in relation to the Conversion-prohibited parameter of the DELIVER.Indication primitive.

There is no result for a Deliver operation. The error defined allows the SDE to indicate that the Deliver was received in violation of a Control restriction.

### 4.3.4 *Notify operation*

The Notify operation provides an indication to the SDE of successful or unsuccessful delivery of a message. It reflects an event identifier for correlation with the previous Submit or Probe operation, and a sequence of recipient information indicating success or failure for one or more recipients. Notify may also carry the returned message content to the originator.

The following data types used in the Notify operation correspond to the similarly named parameters of the NOTIFY.Indication primitive (see § 2.2.10):

**submitOrProbe EventID**
**UAContentID**
**returned Content**
**recipient ORName**
**converted EncodedInformationTypes**
**intendedRecipient ORName**
**SupplementaryInformation**

The data type **Report** (defined in § 3) is used to convey the information described in relation to the following parameters of the NOTIFY.Indication primitive:

Notification-type
Non-delivery-reason
Delivery-time
Type-of-UA

There is no result for a Notify operation. The error defined allows the SDE to indicate that the Notify was received in violation of a Control restriction.

## 4.4 *Management of associations*

### 4.4.1 *Establishing and releasing associations*

The SDE establishes associations in response to LOGON.Request primitives at the MTL. The SDE may establish more than one association as a result of a single LOGON.Request. The number of simultaneous associations to be used is subject to agreement between the UA and the MTA.

The MTAE establishes associations as a result of some local rule (for example, the time of day or the number of messages awaiting transfer). The OPEN.Confirmation and OPEN.Indication contain an indication of the number of messages and total number of octets awaiting delivery, for each priority.

Associations are established by invoking the RTS' OPEN.Request primitive with the following parameters:

| **Parameter name** | **Parameter value** |
|---|---|
| Responder-address | The session address of the RTS of the corresponding SDE or MTAE |
| Dialogue-mode | Monologue or two-way alternate, according to bilateral agreement |
| Initial-turn | With the initiator if, and only if, it has OPDUs to send (this only applies to two-way alternate associations) |
| Application-protocol | P3 |
| **User-data** | **SET** {<br>**P3Identifier,**<br>**messagesWaiting [1] SET** {<br>**urgent [0] IMPLICIT DeliveryQueue,**<br>**normal [1] IMPLICIT DeliveryQueue,**<br>**nonUrgent [2] IMPLICIT DeliveryQueue}OPTIONAL,**<br>*-- not present when the SDE opens the association, or when*<br>*-- the user has not subscribed to Hold for Delivery*<br>**password [2] ANY**}<br><br>**P3Identifier :: = CHOICE** {<br>**P1.ORName,** *-- for the UA*<br>**mTAName [0] IMPLICIT IA5STRING**}<br><br>**DeliveryQueue :: = SET** {<br>**messages [0] IMPLICIT INTEGER,**<br>**octets [1] IMPLICIT INTEGER OPTIONAL**} |

Upon receipt of the OPEN.Indication, a LOGON.Indication is generated. When the LOGON.Response is received, an OPEN.Response is issued with the following parameters:

| **Parameter name** | **Parameter value** |
|---|---|
| Disposition | Accepted or refused |
| User-data | If the association was accepted, the same as above |
| Refusal-reason | Unacceptable dialogue-mode, Busy, Validation Failure. Other reasons are for further study |

Associations are terminated by means of the CLOSE.Request primitive. This may be issued in response to a LOGOFF.Request, or if one of several associations between the two entities has no further use.

### 4.4.2 *Transferring OPDUs*

The entity that possesses the turn may transfer an OPDU by issuing a TRANSFER.Request primitive with the following parameters:

| **Parameter name** | **Parameter value** |
|---|---|
| APDU | OPDU |
| Transfer-time | Determined by a local rule |

When the RTS returns an untransferred OPDU to the UA (by means of EXCEPTION.Indication), the action taken depends upon the OPDU returned as follows:

| **OPDU returned** | **Action taken** |
|---|---|
| Invoke (Register) | Close the association |
| Invoke (Control) | Close the association |
| Invoke (Change-Password) | Issue CHANGE-PASSWORD.Confirmation indicating failure |
| Invoke (Submit) | Issue SUBMIT.Confirmation indicating failure |
| Invoke (Probe) | Issue PROBE.Confirmation indicating failure |
| Invoke (Cancel) | Issue CANCEL.Confirmation indicating failure |
| ReturnResult | Close the association |
| ReturnError | Close the association |

When the RTS returns an untransferred OPDU to the MTA (by means of EXCEPTION.Indication), the action taken depends upon the OPDU returned as follows:

| **OPDU returned** | **Action taken** |
|---|---|
| Invoke (Control) | Close the association |
| Invoke (Deliver) | Generate a non-delivery notification |
| Invoke (Notify) | Discard the OPDU |
| ReturnResult | Close the association |
| ReturnError | Close the association |

### 4.4.3 *Managing the turn*

If the association is two-way alternate, the entity without the turn may issue a TURN-PLEASE.Request primitive whose priority parameter reflects the highest priority OPDU awaiting transfer:

| **Parameter name** | **Parameter value** |
|---|---|
| Priority | 1 = Invoke (Control)<br>2 = Invoke (Submit or Deliver; urgent message)<br>3 = Invoke (Change-Password, Register, Probe, or Notify), ReturnResult, or ReturnError<br>4 = Invoke (Cancel, Submit, or Deliver; normal message)<br>5 = Invoke (Submit or Deliver; non-urgent message) |

When the entity requests the turn so that it can issue a CLOSE.Request primitive, the priority parameter should reflect the lowest priority OPDU the entity is prepared to receive before releasing the association.

The entity that has the turn will issue a TURN-GIVE.Request primitive in response to a TURN-PLEASE.Indication when it has no OPDUs to transfer whose priorities are equal to or higher than that indicated in the TURN-PLEASE.Indication.

ANNEX A

(to Recommendation X.411)

**Abbreviations**

The following abbreviations are used in this Recommendation.

| | |
|---|---|
| ADMD | Administration Management Domain |
| APDU | Application Protocol Data Unit |
| DCS | Digital Command Signal |
| FIF | Facsimile Information Field |
| FIFO | First-In-First-Out |
| MD | Management Domain |
| MHS | Message Handling System |
| MPDU | Message Protocol Data Unit |
| MT | Message Transfer |
| MTA | Message Transfer Agent |
| MTAE | Message Transfer Agent Entity |
| MTL | Message Transfer Layer |
| MTS | Message Transfer System |
| OPDU | Operation Protocol Data Unit |
| O/R | Originator/Recipient |
| PRMD | Private Management Domain |
| P1 | Message Transfer Protocol |
| P3 | Submission and Delivery Protocol |
| RTS | Reliable Transfer Server |
| SDE | Submission and Delivery Entity |
| SFD | Simple Formattable Document |
| SMPDU | Service Message Protocol Data Unit |
| TIF0 | Text Interchange Format 0 |
| TIF1 | Text Interchange Format 1 |
| TSAP | Transport Service Access Point |
| UA | User Agent |
| UAE | User Agent Entity |
| UAL | User Agent Layer |
| UMPDU | User Message Protocol Data Unit |
| UTC | Coordinated Universal Time |

ANNEX B

(to Recommendation X.411)

**Summary of application-wide identifiers**

| *Module* | *ID Value* | *Name* |
|---|---|---|
| P1 | 0 | **ORName** |
| P1 | 1 | **CountryName** |
| P1 | 2 | **AdministrationDomainName** |
| P1 | 3 | **GlobalDomainIdentifier** |
| P1 | 4 | **MPDUIdentifier** |
| P1 | 5 | **EncodedInformationTypes** |
| P1 | 6 | **ContentType** |
| P1 | 7 | **Priority** |
| P1 | 8 | **PerMessageFlag** |
| P1 | 9 | **TraceInformation** |
| P1 | 10 | **UAContentID** |
| P2 | 11 | **IPMessageID** |

APPENDIX I

(to Recommendation X.411)

**List of Terms**

The following terms are contained in this Recommendation. References to definitions, implicit or explicit, are included where appropriate and where available.

## A

**Access Termination Service**
cont. in X.411; impl. def. in X.411

**Address**
cont. in X.411; impl. def. in X.411

**Administration Management Domain**
cont. in X.411; impl. def. in X.400

**Alternate-recipient-allowed**
cont. in X.411; impl. def. in X.411

**Alternate Recipient Allowed**
cont. in X.411; impl. def. in X.411

**Alternate Recipient Assignment**
cont. in X.411; impl. def. in X.400

**Application-protocol**
cont. in X.411; impl. def. in X.411

**Application Protocol Data Unit**
cont. in X.411; impl. def. in X.411

**Association Manager**
cont. in X.411; impl. def. in X.411

## C

**Cancel**
cont. in X.411; impl. def. in X.411

**Cancel Deferred Delivery Service**
cont. in X.411; impl. def. in X.411

**Cancel Operation**
cont. in X.411; impl. def. in X.411

**Change-Password Operation**
cont. in X.411; impl. def. in X.411

**Common Data Types**
cont. in X.411; impl. def. in X.411

**Confirmation**
cont. in X.411; impl. def. in X.411

**Content**
cont. in X.411; impl. def. in X.411

**Content-length**
cont. in X.411; impl. def. in X.411

**Content-return**
cont. in X.411; impl. def. in X.411

**Content-type**
cont. in X.411; impl. def. in X.411

**Content-Type**
cont. in X.411; impl. def. in X.400

**Content Type Indication**
cont. in X.411; impl. def. in X.400

**Control**
cont. in X.411; impl. def. in X.411

**Control Operation**
cont. in X.411; impl. def. in X.411

**Conversion-prohibited**
cont. in X.411; impl. def. in X.411

**Conversion Prohibition**
cont. in X.411; impl. def. in X.400

**Converted-encoded-information-type**
cont. in X.411; impl. def. in X.411

## D

**Default-control-settings**
cont. in X.411; impl. def. in X.411

**Deferred Delivery**
cont. in X.411; impl. def. in X.400

**Deferred-delivery-time**
cont. in X.411; impl. def. in X.411

**Deliver**
cont. in X.411; impl. def. in X.411

**Deliverable-encoded-information-type**
cont. in X.411; impl. def. in X.411

**Deliver-event-id**
cont. in X.411; impl. def. in X.411

**Delivery-notice**
cont. in X.411; impl. def. in X.411
Delivery Notification
cont. in X.411; impl. def. in X.400

**Deliver Operation**
cont. in X.411; impl. def. in X.411

**Deliver Report**
cont. in X.411; impl. def. in X.411

**Delivery Report Content**
cont. in X.411; impl. def. in X.411

**Delivery Report Envelope**
cont. in X.411; impl. def. in X.411

**Delivery-time**
cont. in X.411; impl. def. in X.411

**Dialogue-mode**
cont. in X.411; impl. def. in X.411

**Digital Command Signal**
cont. in X.411

# N



# O



# P



# R



# S

**Service Primitives**
cont. in X.411; impl. def. in X.411

**Standard Attribute**
cont. in X.411; impl. def. in X.411

**Submission**
cont. in X.411; impl. def. in X.411

**Submission and Delivery Entity**
cont. in X.411; impl. def. in X.410

**Submission and Delivery Protocol**
cont. in X.411; impl. def. in X.410

**Submission-time**
cont. in X.411; impl. def. in X.411

**Submit**
cont. in X.411; impl. def. in X.411

**Submit-event-id**
cont. in X.411; impl. def. in X.411

**Submit Operation**
cont. in X.411; impl. def. in X.411

**Submit-or-Probe-event-id**
cont. in X.411; impl. def. in X.411

**Success-indication**
cont. in X.411; impl. def. in X.411

**Supplementary-information**
cont. in X.411; impl. def. in X.411

## T

**Telematic Address**
cont. in X.411; impl. def. in X.411

**This-recipient-O/R-name**
cont. in X.411; impl. def. in X.411

**This Recipient**
cont. in X.411; impl. def. in X.411

**Trace Information**
cont. in X.411; impl. def. in X.411

**Transfer-time**
cont. in X.411; impl. def. in X.411

**Type-of-UA**
cont. in X.411; impl. def. in X.411

## U

**UA-content-id**
cont. in X.411; impl. def. in X.411

**User-data**
cont. in X.411; impl. def. in X.411

**User-initiated Access Establishment Service**
cont. in X.411; impl. def. in X.411

**User Agent**
cont. in X.411; impl. def. in X.400

**User Agent Entity**
cont. in X.411; impl. def. in X.400

**User Agent Layer**
cont. in X.411; impl. def. in X.400

**User Message Protocol Data Unit**
cont. in X.411; impl. def. in X.411

**Recommendation X.420**

### MESSAGE HANDLING SYSTEMS: INTERPERSONAL MESSAGING USER AGENT LAYER

*(Malaga-Torremolinos, 1984)*

The establishment in various countries of telematic services and computer-based store-and-forward message services in association with public data networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT,

*considering*

(a) the need for interpersonal messaging and message transfer services;

(b) the need to transfer messages of different types having a large variety of formats;

(c) that Recommendation X.1 includes specific user classes of service and X.10 categories of access, and Recommendation X.2 defines optional user facilities for public data networks;

(d) that the F-series of Recommendations defines telematic services and that the T-series of Recommendations defines terminal equipment and control procedures for telematic services;

(e) that Recommendation X.200 defines the reference model of open systems interconnection for CCITT applications;

(f) that a set of Recommendations describes various aspects of message handling: X.400, X.401, X.408, X.409, X.410, X.411, X.420 and X.430;

*unanimously declares*

(1) that general aspects of the interpersonal messaging user agent layer are described in § 1;

(2) that the model of the interpersonal messaging user agent layer is described in § 2;

(3) that the structure and format of interpersonal messaging user agent protocol data units are described in § 3;

(4) that the operation of interpersonal messaging user agent entities is described in § 4;

(5) that simple formattable documents are described in § 5.

TABLE OF CONTENTS



## 1 Purpose and scope

This Recommendation is one of a series of Recommendations and describes the Interpersonal Messaging User Agent Layer for the Message Handling System (MHS). This Recommendation defines the conceptual operation of UA entities within the User Agent Layer (UAL) for the Interpersonal Messaging (IPM) Service, and the syntax and semantics of the peer protocol between them.

Other Recommendations in the series contain additional information essential to a thorough understanding of the MHS. Recommendation X.401 outlines its basic service elements and optional user facilities. Recommendation X.400 describes the MHS model and defines the MHS service elements. Recommendation X.411 specifies the protocol aspects of the Message Transfer Layer (MTL). Recommendation X.409 defines the notation and representational technique used to specify and encode MHS protocols. Recommendation X.408 specifies the algorithms the MHS uses when converting between different types of encoded information. Recommendation X.430 describes how Teletex terminals access the MHS. Recommendation X.410 describes the general techniques used for MHS protocols and the way in which standard Open Systems Interconnection (OSI) protocols are used to support MHS applications.

§ 2 of this Recommendation describes the model of the IPM UAL. § 3 defines the structure and format of message contents (UAPDUs) exchanged by UA entities (UAEs) in the IPM UAL. § 4 describes the operation of an IPM UAE in providing the IPM Service. § 5 defines the representation used for conveying "simple formattable documents" (SFDs) which can be contained in the body of IP-messages exchanged by IPM Service users.

## 2 Model of the Interpersonal Messaging User Agent Layer

The IPM Service provides its users with the capability to communicate by sending and receiving Interpersonal Messages (IP-messages). The provision of the IPM Service can be described abstractly using the layered description of the Interpersonal Messaging System outlined in § 5.3 of Recommendation X.400 and illustrated in Figure 1/X.420 below. As described, the services available to an originator/recipient making use of the IPM UAL are those of the Message Transfer Layer plus those provided by the cooperation of UA entities. UAEs cooperate with one another by means of a peer protocol called the *Interpersonal Messaging Protocol (P2).* This protocol essentially consists of:

1) The definition of a set of protocol elements, or components, each having a standardized syntax and semantics, which can be used to construct *User Agent Protocol Data Units (UAPDUs).* These UAPDUs comprise the contents of the messages exchanged between IPM UAEs.
2) The operations relating to the exchange of these protocol elements that an IPM UA must perform.
3) The rules which an IPM UA must follow in using the MTL Service in providing the IPM Service.

Two types of IPM contents, or UAPDUs, are defined: *IP-message UAPDUs (IM-UAPDUs)* and *IPM-status-report UAPDUs (SR-UAPDUs)* (see Figure 2/X.420). An IM-UAPDU carries an IP-message generated by and transferred between originators and recipients (see note). The IM-UAPDU conveys both the *heading* and the *body* of the IP-message. The heading of an IP-message is the control information that characterizes it. (For example, it contains elements such as "subject", "to", and "cc".) The IP-message *body* is the information the user wishes to communicate (consisting, for example, of paragraphs of text, voice comments, facsimile diagrams, and combinations of these).

*Note* – In general, there is a one-to-one correspondence between IP-messages and IM-UAPDUs. However, it should be noted that in some cases (for example, involving provision of Blind Copy Recipient Indication – see § 4.2.2.1) more than one IM-UAPDU may be associated with the same IP-message (that is, containing the same IP-message-id).

An IPM-status-report is conveyed in an SR-UAPDU and consists of information generated by an IPM UAE and transferred to another IPM UAE. This status information may be for use by the peer UAE or may be conveyed to a user.

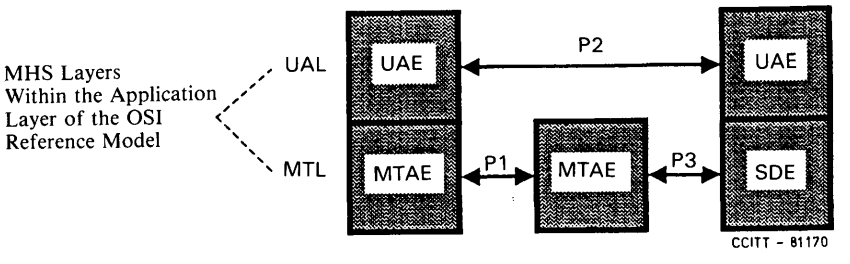
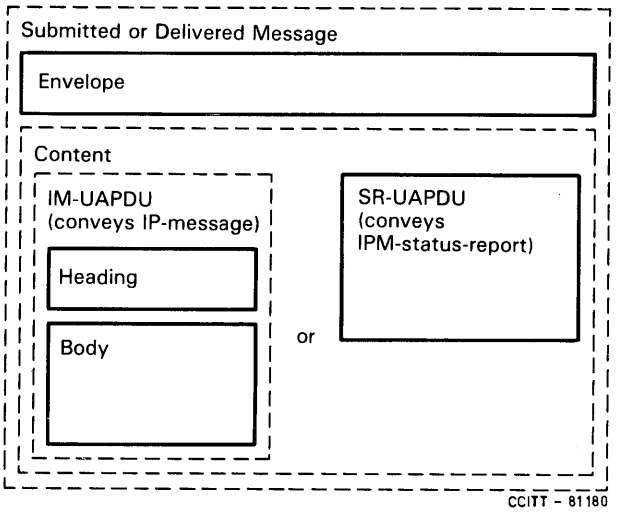



FIGURE 1/X.420.

**Layered Description of the Interpersonal Messaging System**




*Note* — Boxes in this figure represent the nested structure of a submitted or delivered message with embedded IM-UAPDU/SR-UAPDU. Dashed boxes represent structure only; solid boxes represent structure and also indicate the information carried.

FIGURE 2/X.420.
**Messages Exchanged by IPM UASs**

## 3 Structure and format of UAPDUs

The two distinct UA Protocol Data Unit (UAPDU) types are defined as follows:

```
UAPDU :: = CHOICE {
    [0] IMPLICIT IM-UAPDU,
    [1] IMPLICIT SR-UAPDU}
```

This section specifies the structure of each UAPDU type using the notation defined in Recommendation X.409. The binary representation of each UAPDU can be inferred from its notational description with the aid of that Recommendation.

The following defined data types, from Recommendation X.409, are used in the specification of the P2 protocol: UTC Time, IA5 String, T.61 String, Videotex String, and Printable String. A set of application-wide identifiers is defined for the MHS protocols (including P1, P2 and P3). The assignment of values for these identifiers is contained in Annex B of Recommendation X.411.

### 3.1 *Common data types*

The following data types figure in the definition of more than one UAPDU.

#### 3.1.1 *IP-message-Id*

An *IP-Message-ID* identifies a particular IP-message and is used in providing the IP-message Identification service element (see § 4.2.1.2 of Recommendation X.400). It is intended to be a unique and unambiguous identifier and is assumed to be specified as recommended in the service element definition.

An IP-Message-Id contains a Printable String and may also include an O/R name. If present, the O/R name denotes the user (or UA) who generated this identifier. The Printable String is assigned by the generating user (or UA) and is intended to be unique within that user's context. When the user's O/R name is present, therefore, the identifier is globally unique:

```
IPMessageId :: = [APPLICATION 11] IMPLICIT SET {ORName OPTIONAL, PrintableString}

ORName :: = P1.ORName
```

### 3.1.2 *Time*

A **Time** identifies a calender date and time of day to be conveyed in the protocol. It must always include the time of day in terms of UTC (Universal Coordinated Time), and may optionally convey the local time of the UAE which generates it. The precision of the Time is to either one second or to one minute, as selected by the generating UAE:

```
Time :: = UTCTime
```

## 3.2 *IM-UAPDU*

An IM-UAPDU, whose formal definition is summarized in Figure 3/X.420, has two parts, the heading and body:

```
IM-UAPDU :: = SEQUENCE{Heading, Body}
```

### 3.2.1 *Heading*

The heading of an IM-UAPDU contains a number of components. Each is used in the provision of an IPM service element:

```
Heading :: = SET{
    IPMessageId,
    originator              [0]  IMPLICIT ORDescriptor OPTIONAL,
    authorizingUsers        [1]  IMPLICIT SEQUENCE OF ORDescriptor OPTIONAL,  -- only if not the originator
    primaryRecipients       [2]  IMPLICIT SEQUENCE OF Recipient OPTIONAL,
    copyRecipients          [3]  IMPLICIT SEQUENCE OF Recipient OPTIONAL,
    blindCopyRecipients     [4]  IMPLICIT SEQUENCE OF Recipient OPTIONAL,
    inReplyTo               [5]  IMPLICIT IPMessageId OPTIONAL,
                                 -- omitted if not in reply to a previous message
    obsoletes               [6]  IMPLICIT SEQUENCE OF IPMessageId OPTIONAL,
    crossReferences         [7]  IMPLICIT SEQUENCE OF IPMessageId OPTIONAL,
    subject                 [8]  CHOICE {T61String} OPTIONAL,
    expiryDate              [9]  IMPLICIT Time OPTIONAL,
                                 -- if omitted, expiry date is never
    replyBy                 [10] IMPLICIT Time OPTIONAL,
    replyToUsers            [11] IMPLICIT SEQUENCE OF ORDescriptor OPTIONAL,
                                 -- each O/R descriptor must contain an O/R name
    importance              [12] IMPLICIT INTEGER{low(0), normal(1), high(2)} DEFAULT normal,
    sensitivity             [13] IMPLICIT INTEGER{personal(1), private(2), companyConfidential(3)}
                                 OPTIONAL,
    autoforwarded           [14] IMPLICIT BOOLEAN DEFAULT FALSE
                                 -- indicates that the forwarded message body part(s) were
                                 -- auto-forwarded- -}
```

Table 1/X.420 shows, for each component of the heading, the corresponding IPM service element (defined in Recommendation X.400) which it is used to provide.

The **originator** component identifies the user who submits an IP-message. The **authorizingUsers** are those users who authorized the sending of the IP-message. An *O/R descriptor*, which is used to represent these components, can include an O/R name, free-form name, and telephone number.

The O/R name is that which is needed to identify the user within the IPMS. As described elsewhere in this section, there are circumstances involving the provision of certain service elements where the O/R name cannot be omitted. Additionally or alternatively, a *free-form name* denoting the user can be included. This component allows the user to include a familiar, user-friendly, or other name form not subject to the constraints of the O/R name. It is intended to be of limited length and is not intended to carry arbitrary information. The O/R descriptor can also optionally include a telephone number which could, for example, be used to access the user:

```
ORDescriptor :: = SET{ -- at least one of the first two members must be present
    ORName OPTIONAL,
    freeformName [0] IMPLICIT T61 String OPTIONAL,
    telephoneNumber [1] IMPLICIT PrintableString OPTIONAL}
```

The **primaryRecipients**, **copyRecipients**, and **blindCopyRecipients** are the users specified by the originator to receive this IP-message in those particular roles. Each is denoted by a **Recipient** component.

TABLE 1/X.420

**Use of IM-UAPDU heading components in provision of IPM services**

| IM-UAPDU heading component(s) | IPM service elements(s) |
|---|---|
| **IPMessageId** | IP message identification |
| **originator** | Originator indication |
| **authorizingUsers** | Authorizing users indication |
| **primaryRecipients**<br>**copyRecipients** | Primary and copy recipients indication<br>Reply request indication<br>Receipt notification<br>Non-receipt notification |
| **blindCopyRecipients** | Blind copy recipient indication<br>Reply request indication<br>Receipt notification<br>Non-receipt notification |
| **inReplyTo** | Replying IP-message indication |
| **obsoletes** | Obsoleting indication |
| **crossReferences** | Cross referencing indication |
| **subject** | Subject indication |
| **expiryDate** | Expiry date indication |
| **replyBy**<br>**replyToUsers** | Reply request indication |
| **importance** | Importance indication |
| **sensitivity** | Sensitivity indication |
| **autoforwarded** | Autoforwarded indication |

A **Recipient** component conveys an identification of a recipient which is an O/R descriptor. In addition, it includes per-recipient requests applying to that user which are the **reportRequest** and **replyRequest**. The **reportRequest** can indicate requests for **receiptNotification**, **nonReceiptNotification** and for return of this IP-message with a returned non-receipt notification (**returnIPmessage**). When **receiptNotification** is selected, **nonReceiptNotification** shall also be selected. The **replyRequest** denotes a request to the recipient to reply to this IP-message. If any of these per-recipient requests are selected by the originator for a particular recipient, then the O/R descriptor identifying that recipient *must* include an O/R name component in order for the request to be successfully recognized and acted upon.

```
Recipient :: = SET {
    [0]IMPLICIT ORDescriptor,
    reportRequest [1] IMPLICIT BIT STRING {
        receiptNotification(0), nonReceiptNotification(1), returnIPMessage(2)} DEFAULT {},
            -- if requested, the O/R descriptor must contain an O/R name
    replyRequest [2] IMPLICIT BOOLEAN DEFAULT FALSE
        -- if true, the O/R descriptor must contain an O/R name-- }
```

The **inReplyTo** component identifies a previous IP-message to which this IP-message is a reply.

The **obsoletes** and **crossReferences** components identify any IP-messages which are obsoleted or cross-referenced, respectively, by this IP-message.

The **subject** component conveys information the originator specified as the subject of the IP-message.

The **expiryDate** indicates the date and time by which the originator considers the IP-message to be no longer valid and useful. The time by which the originator would like to have a reply sent is specified as the **replyByTime**. The **replyToUsers** are those to whom the originator wishes to have any reply to this IP-message sent. The O/R Descriptor of each such potential recipient of the reply must include an O/R name component in order for the reply to be successfully sent by means of the IPMS.

The **importance** component is an indication of the IP-message's importance as specified by the originator. The importance may be **low**, **normal** or **high**. The originator's classification of the IP-message's **sensitivity** can also be conveyed. The sensitivity may be **personal**, **private**, or **company confidential**.

If the body of the IP-message contains another IP-message which was **auto-forwarded**, this is indicated.

### 3.2.2 *Body*

The body of an IM-UAPDU consists of one or more independent parts:

```
Body :: = SEQUENCE OF BodyPart
```

Each part carries with it an indication of its type. Where necessary the representation of a particular body part type is structured to include, along with the actual body part information, a set of parameters which are associated with or are required for the interpretation of the body part information. The various parts of a particular IP-message body may be of different types:

```
BodyPart :: = CHOICE {
    [0] IMPLICIT IA5Text,
    [1] IMPLICIT TLX,
    [2] IMPLICIT Voice,
    [3] IMPLICIT G3Fax,
    [4] IMPLICIT TIF0,
    [5] IMPLICIT TTX,
    [6] IMPLICIT Videotex,
    [7] NationallyDefined,
    [8] IMPLICIT Encrypted,
    [9] IMPLICIT ForwardedIPMessage,
   [10] IMPLICIT SFD,
   [11] IMPLICIT TIF1}
```

A **IA5Text** body part contains a string of characters, together with an indication of which repertoire (ITA2 or IA5) is in use. In either case, the characters are *represented* using the appropriate IA5 bit combinations:

```
IA5Text :: = SEQUENCE {
    SET {repertoire [0] IMPLICIT INTEGER {ia5(5), ita2(2)} DEFAULT ia5
        -- additional members of this Set are a possible future extension--},
    IA5String}
```

A **TLX** body part uses the five-bit code assignments of ITA2 for conveying telex information. The details of the representation of this body part are for further study:

```
TLX :: = for further study
```

A **Voice** body part contains a bit string representing digitized voice. The encoding method, and the other information which must be conveyed, are for further study:

```
Voice :: = SEQUENCE {
    SET, -- members are for further study
    BIT STRING}
```

A **G3Fax** body part contains a sequence of bit strings, each representing a page of Group 3 facsimile information encoded according to Recommendation T.4. The number of pages is optionally conveyed, as are the applicable non-basic parameters, if any:

```
G3Fax :: = SEQUENCE {
    SET {
        numberOfPages [0] IMPLICIT INTEGER OPTIONAL,
        [1] IMPLICIT P1.G3NonBasicParamsOPTIONAL},
    SEQUENCE OF BIT STRING}
```

A **TIF0** body part contains a document whose structure is defined in Recommendation T.73 and which conforms to **TIF** (Text Interchange Format). 0 application rules.

*Note* – TIF.0 is the Text Interchange Format defined in Recommendation T.73 which is used by, and to communicate with, Group 4 facsimile class 1 terminals.

```
TIF0 :: = T73Document
```

A **T73Document** is a sequence of protocol elements describing the layout structure of a document as defined in Recommendation T.73:

```
T73Document :: = SEQUENCE OF T73 ProtocolElement
```

A **TTX** body part represents a Teletex document, and contains a sequence of strings of T.61 characters, each string representing a page of that document. The number of pages is optionally conveyed, as are an indication of whether or not the document is telex-compatible and the applicable non-basic parameters, if any:

```
TTX :: = SEQUENCE {
    SET {
        numberOfPages [0] IMPLICIT INTEGER OPTIONAL,
        telexCompatible [1] IMPLICIT BOOLEAN DEFAULT FALSE,
        [2] IMPLICIT P1.TeletexNonBasicParams OPTIONAL},
    SEQUENCE OF T61 String}
```

A **Videotex** body part contains a string encoded using the encoding scheme of Recommendations T.100 or T.101. Further properties which may have to be conveyed are for further study:

```
Videotex :: = SEQUENCE {
    SET, -- members are for further study
    VideotexString}
```

**NationallyDefined** body part(s) can be included. However, definition of such a body part is outside the scope of this Recommendation:

```
NationallyDefined :: = ANY
```

An **Encrypted** body part represents another body part which has been subjected to encryption. The ciphertext itself is represented by a bit string. Other information which may have to be conveyed, such as an identifier for the encryption algorithm, are for further study:

```
Encrypted :: = SEQUENCE {
    SET, -- members are for further study
    BIT STRING}
```

A **ForwardedIPMessage** body part represents an IP-message which is contained in another IP-message body for the purpose of conveying it to a further set of recipients. It includes, optionally, the delivery information supplied when the IP-message was originally delivered:

```
ForwardedIPMessage :: = SEQUENCE {
    SET {
        delivery [0] IMPLICIT Time OPTIONAL,
        [1] IMPLICIT DeliveryInformation OPTIONAL},
    IM-UAPDU}

DeliveryInformation :: = P3.DeliverEnvelope
    -- This merely reuses a data type definition, and does not imply that the information was ever carried in P3.
```

An **SFD** body part represents an instance of a *Simple Formattable Document*, as defined in § 5 of this Recommendation:

```
SFD :: = SFD.Document
```

A **TIF1** body part contains a document whose structure is defined in Recommendation T.73, and which conforms to TIF.1 application rules.

*Note* – TIF.1 is the Text Interchange Format defined in Recommendation T.73 which can be used by mixed mode Teletex and Group 4 facsimile class 2 and 3 terminals.

```
TIF1 :: = T73Document
```

## 3.3 *SR-UAPDU*

The SR-UAPDU, whose formal definition is summarized in Figure 4/X.420, is used to return notification of receipt or non-receipt of an IP-message to its originator:

```
SR-UAPDU :: = SET {
    [0] CHOICE {
        nonReceipt          [0]  IMPLICIT NonReceiptInformation,
        receipt             [1]  IMPLICIT ReceiptInformation},
    reported                IPMessageId,
    actualRecipient         [1]  IMPLICIT ORDescriptor OPTIONAL,
    intendedRecipient       [2]  IMPLICIT ORDescriptor OPTIONAL,
        -- only present if not actual recipient. The O/R Descriptor must contain an O/R Name
    converted               P1.EncodedInformationTypes OPTIONAL}
```

A **NonReceiptInformation** component is included if, and only if, notification of non-receipt is being conveyed. Its components include a **reason** which indicates either **uaeInitiatedDiscard** (when the UAE has discarded the IP-message for local reasons) or **autoforwarded** (when the UAE has autoforwarded the IP-message to another UAE). In cases of UAE discard of an IP-message, the **nonReceiptQualifier** conveys an indication of **expired** (if discard occurred due to arrival of the expiry date), **obsoleted** (if the IP-message was obsoleted by another one), or **subscriptionTerminated**. In the case of autoforwarding, the **comments** component optionally conveys a string of characters, supplied by the UAE that performed the autoforwarding. If the originator requested return of the IP-message upon non-receipt, the **returned** component conveys the entire IM-UAPDU that was delivered:

```
NonReceiptInformation :: = SET {
    reason                  [0]  IMPLICIT INTEGER {uaeInitiated Discard(0), autoForwarded (1)},
    nonReceiptQualifier     [1]  IMPLICIT INTEGER {expired(0), obsoleted(1), subscriptionTerminated(2)}
                                 OPTIONAL,
    comments                [2]  IMPLICIT PrintableString OPTIONAL, -- on auto-forward
    returned                [3]  IMPLICIT IM-UAPDU OPTIONAL}
```

A **ReceiptInformation** component is included if, and only if, notification of receipt is being conveyed. Its **receipt Time** component indicates the time at which the IM-UAPDU was actually received. The **typeOfReceipt** will indicate either **explicit** (in the case where the recipient voluntarily and explicitly authorized sending the notification of receipt) or **automatic** (in the case where the UAE automatically generated the notification when the IP-message was received). The use of **SupplementaryInformation** in the SR-UAPDU is reserved for the communication of responses from telex and telematic terminals (for example, answerback):

```
ReceiptInformation :: = SET {
    receipt [0] IMPLICIT Time,
    typeOfReceipt [1] IMPLICIT INTEGER {explicit(0),automatic(1)} DEFAULT explicit,
    [2] IMPLICIT P1.SupplementaryInformation OPTIONAL}
```

The **reported IPMessageId** identifies the IP-message whose fate is being reported by this SR-UAPDU.

The **actualRecipient** component identifies the UAE to which the IM-UAPDU was delivered, and which is generating this IPM-status-report. If the IM-UAPDU was delivered to the originally specified recipient, then this component conveys exactly the corresponding O/R Descriptor from the **Recipient** component which identified the actual recipient in the delivered IM-UAPDU. If the IM-UAPDU was delivered to an alternate recipient, then this component is an O/R Descriptor which identifies the actual recipient. It must, however, include the actual recipient's O/R Name.

The **intendedRecipient** component is returned only if the IM-UAPDU was actually delivered to an alternate recipient. This component is then the O/R Descriptor that identifies the originally intended recipient, exactly as conveyed in the corresponding **Recipient** component of the IM-UAPDU being reported on.

The **converted EncodedInformationTypes** component conveys the new **EncodedInformationTypes** if conversion took place on the IM-UAPDU.

```
P2 DEFINITIONS :: =
BEGIN

-- P2 makes use of types defined in the following modules:
--      P1: X.411, § 3.4
--      P3: X.411, § 4.3
--      SFD: this Recommendation, § 5
--      T73: T.73, § 5

UAPDU                   :: = CHOICE {
                                 [0] IMPLICIT IM-UAPDU,
                                 [1] IMPLICIT SR-UAPDU}

-- IP-message UAPDU

IM-UAPDU                :: = SEQUENCE {Heading, Body}

-- heading

Heading                 :: = SET {
    IPMessageId,
    originator          [0] IMPLICIT ORDescriptor OPTIONAL,
    authorizingUsers    [1] IMPLICIT SEQUENCE OF ORDescriptor OPTIONAL,
                                 -- only if not the originator
    primaryRecipients   [2] IMPLICIT SEQUENCE OF Recipient OPTIONAL,
    copyRecipients      [3] IMPLICIT SEQUENCE OF Recipient OPTIONAL,
    blindCopyRecipients [4] IMPLICIT SEQUENCE OF Recipient OPTIONAL,
    inReplyTo           [5] IMPLICIT IPMessageId OPTIONAL,
                                 -- omitted if not in reply to a previous message
    obsoletes           [6] IMPLICIT SEQUENCE OF IPMessageId OPTIONAL,
    crossReferences     [7] IMPLICIT SEQUENCE OF IPMessageId OPTIONAL,
    subject             [8] CHOICE {T61String} OPTIONAL,
    expiryDate          [9] IMPLICIT Time OPTIONAL,
                             -- if omitted, expiry date is never
    replyBy             [10] IMPLICIT Time OPTIONAL,
    replyToUsers        [11] IMPLICIT SEQUENCE OF ORDescriptor OPTIONAL,
                                 -- each O/R descriptor must contain an O/R name
    importance          [12] IMPLICIT INTEGER {low(0), normal(1), high(2)} DEFAULT normal,
    sensitivity         [13] IMPLICIT INTEGER {personal(1), private(2), companyConfidential(3)}
                             OPTIONAL,
    autoforwarded       [14] IMPLICIT BOOLEAN DEFAULT FALSE
      -- indicates that the forwarded message body part(s) were autoforwarded-- -}

IP MessageId            :: = [APPLICATION 11] IMPLICIT SET {
                                 ORName OPTIONAL,
                                 PrintableString}

ORName                  :: = P1.ORName

-- P2 definitions to be continued
```

FIGURE 3/X.420 (Part 1 of 3)

**Formal definition of IM-UAPDU**

```
-- P2 definitions continued

ORDescriptor    ::= SET {-- at least one of the first two members must be present
                        ORName OPTIONAL,
                        freeformName [0] IMPLICIT T61String OPTIONAL,
                        telephoneNumber [1] IMPLICIT PrintableString OPTIONAL}

Recipient       ::= SET {
                        [0] IMPLICIT ORDescriptor,
                        reportRequest [1] IMPLICIT BIT STRING {
                            receiptNotification(0),
                            nonReceiptNotification(1),
                            returnIPMessage(2)} DEFAULT {},
                            -- if requested, the O/R descriptor must contain an O/R name
                        replyRequest [2] IMPLICIT BOOLEAN DEFAULT FALSE
                            -- if true, the O/R descriptor must contain an O/R name --}

-- body

Body            ::= SEQUENCE OF BodyPart

BodyPart        ::= CHOICE {
                        [0] IMPLICIT IA5Text
                        [1] IMPLICIT TLX,
                        [2] IMPLICIT Voice,
                        [3] IMPLICIT G3Fax,
                        [4] IMPLICIT TIF0,
                        [5] IMPLICIT TTX,
                        [6] IMPLICIT Videotex,
                        [7] NationallyDefined,
                        [8] IMPLICIT Encrypted,
                        [9] IMPLICIT ForwardedIPMessage,
                        [10] IMPLICIT SFD,
                        [11] IMPLICIT TIF1}

-- body part types

IA5Text         ::= SEQUENCE {
                        SET {repertoire [0] IMPLICIT INTEGER {ia5(5), ita2(2)}
                            DEFAULTia5
                            -- additional members of this Set are a possible future extension --},
                        IA5String}

TLX             ::= for further study

Voice           ::= SEQUENCE {
                        SET, -- members are for further study
                        BIT STRING}

-- P2 definitions to be continued
```

FIGURE 3/X.420 (Part 2 of 3)

**Formal definition of IM-UAPDU**

```
-- P2 definitions continued

G3Fax                    ::= SEQUENCE {
                                 SET {
                                     numberOfPages [0] IMPLICIT INTEGER OPTIONAL,
                                     [1] IMPLICIT P1.G3NonBasicParams OPTIONAL},
                                 SEQUENCE OF BIT STRING}

TIF0                     ::= T73Document

T73Document              ::= SEQUENCE OF T73.ProtocolElement

TTX                      ::= SEQUENCE {
                                 SET {
                                     numberOfPages [0] IMPLICIT INTEGER OPTIONAL,
                                     telexCompatible [1] IMPLICIT BOOLEAN DEFAULT FALSE,
                                     [2] IMPLICIT P1.TeletexNonBasicParams OPTIONAL},
                                 SEQUENCE OF T61String}

Videotex                 ::= SEQUENCE {
                                 SET, -- members are for further study
                                 VideotexString}

NationallyDefined        ::= ANY

Encrypted                ::= SEQUENCE {
                                 SET, -- members are for further study
                                 BIT STRING}

ForwardedIPMessage       ::= SEQUENCE {
                                 SET {
                                     delivery [0] IMPLICIT Time OPTIONAL,
                                     [1] IMPLICIT DeliveryInformation OPTIONAL},
                                 IM-UAPDU}

DeliveryInformation      ::= P3.DeliverEnvelope
                             -- This merely reuses a data type definition, and does not imply that the information
                             -- was ever carried in P3.

SFD                      ::= SFD.Document

-- note that SFD and T73 Document use the same space of application-wide tags which is different from that used for
-- other MHS protocols

TIF1                     ::= T73Document

-- P2 definitions to be continued
```

FIGURE 3/X.420 (Part 3 of 3)

**Formal definition of IM-UAPDU**

```
-- P2 definitions continued

-- IPM-status-report UAPDU

SR-UAPDU                ::= SET {
                                [0] CHOICE {
                                    nonReceipt      [0] IMPLICIT NonReceiptInformation,
                                    receipt         [1] IMPLICIT ReceiptInformation},
                                reported            IPMessageId,
                                actualRecipient     [1] IMPLICIT ORDescriptor OPTIONAL,
                                intendedRecipient   [2] IMPLICIT ORDescriptor OPTIONAL,
                                    -- only present if not actual recipient the O/R descriptor must contain an
                                    -- O/R named
                                converted           P1.EncodedInformationTypes OPTIONAL}

NonReceiptInformation   ::= SET {
                                reason [0] IMPLICIT
                                    INTEGER {uaeInitiatedDiscard(0), autoForwarded(1)},
                                nonReceiptQualifier [1] IMPLICIT
                                    INTEGER {expired(0), obsoleted(1), subscriptionTerminated(2)}
                                    OPTIONAL,
                                comments [2] IMPLICIT PrintableString OPTIONAL,
                                    -- on auto-forward
                                returned [3] IMPLICIT IM-UAPDU OPTIONAL}

ReceiptInformation      ::= SET {
                                receipt [0] IMPLICIT Time,
                                typeOfReceipt [1] IMPLICIT INTEGER {
                                    explicit(0),automatic(1)} DEFAULT explicit,
                                [2] IMPLICIT P1.SupplementaryInformation OPTIONAL}

END     -- of P2 definitions
```

FIGURE 4/X.420

**Formal definition of SR-UAPDU**

## 4 Operation of IPM User Agent Entities

To provide the IPM Service, IPM UAEs must perform operations which support specific service elements, as well as certain management functions. For each service element, the required operations, if any, of the originator's and the recipient's UAE are specified. In addition, the UAE management operations which involve the MTL are also described.

The operation of an IPM UAE is described in terms of:

1) Its interactions with its originating or recipient IPM Service user.
2) Its interactions with the underlying MTL Service which it uses to communicate with peer UAEs.

Interactions with the MTL are described using the MTL service primitives defined in Recommendation X.411. Interactions with the IPM Service user are described in general terms, recognizing the fact that the details of this interface are dependent upon the locally provided UA functions, which will vary widely among different UAs. It is not the subject of this Recommendation to specify such local UA functionality, nor is the description of UAE operation contained herein intended to specify or constrain any part of an IPM Service implementation.

### 4.1 *Provision of the basic Interpersonal Messaging Service*

The basic IPM Service (described in § 4.2.1 of Recommendation X.400) enables an originating user to send an IP-message and have it received by the specified recipient. As part of this basic service, an IP-message identification and the body part, along with an indication of its body part type, are conveyed, the MT Service is accessed, and each of the basic service elements of the MT Service (described in § 4.1.1 of Recommendation X.400) is available to transfer the IP-message.

#### 4.1.1 *MT Service access*

IPM UAEs utilize the Message Transfer Service as a means of transferring IP-messages, and thus must have access to it. The Access Management service element (see § 4.1.1.8 of Recommendation X.400), which is part of the basic MT Service, is utilized by UAEs for this purpose.

The MTL primitives which are associated with access management are summarized in Table 2/X.420.

The specific manner in which these service primitives are (or are not) used between a UAE and the MTL is dependent upon the procedures which are required by the particular Management Domain providing the MT Service being accessed.

#### 4.1.2 *Sending and receipt of IP-messages*

Provision of the basic service of sending and receiving IP-messages requires action by both the originator's and the recipient's UAE. The originator's UAE performs the following:

1) Constructs an IM-UAPDU containing the **IPMessageId** and **Body** components in the P2 syntax.
2) Issues a SUBMIT.Request primitive to the MTL, supplying the constructed IM-UAPDU as the content and the parameters of the SUBMIT.Request which correspond to the basic MT Service (see Column A of Table 3/X.420).
3) If the SUBMIT.Confirmation primitive returned by the MTL indicates that the submission of the IM-UAPDU failed, the UAE must make the originator aware of the failure. The failure-reason parameter indicated by the primitive must also be available to the originator.
4) If the SUBMIT.Confirmation primitive indicates that the submission was successful, the primitive provides submission time and submit-event-id parameters which can be used to correlate further notifications with this IM-UAPDU (see Note 1 below).
5) After a successful submission, it is possible that the UAE will receive a NOTIFY.Indication primitive signalling that the IM-UAPDU was not delivered. The UAE must make the originator aware that the non-delivery of this IP-message occurred. In addition, in the case of the basic MT Service, the NOTIFY.Indication primitive conveys information in the recipient-O/R-names and non-delivery-reason parameters which must be made available to the originator (see Note 1).

At the recipient's UAE, when the IM-UAPDU is delivered, a DELIVER.Indication primitive is received from the MTL with the parameters of the basic MT Service (originator-O/R-name, recipient-O/R-name, content-type, original-encoded-information-types, submission-time and delivery-time). The IM-UAPDU is the content parameter of the DELIVER.Indication and contains the **IPMessageId** and the **Body** components. All of this information, conveyed as DELIVER.Indication parameters, must be made accessible to the recipient when he receives the IP-message (see Note 2).

*Note 1* – It must be possible to correlate a notification with the previous IM-UAPDU (or Probe; see § 4.3.2) to which it refers. Different mechanisms for achieving this correlation may be used. For example, the submit-event-id parameters supplied in the SUBMIT.Confirmation and NOTIFY.Indication primitives may be used to associate the two events. Alternatively, the value of the **IPMessageId** component of the IM-UAPDU might be supplied as the UA-content-id parameter of the SUBMIT.Request. In this case, the corresponding UA-content-id parameter returned by the NOTIFY.Indication can be used for correlation. (It should be noted that the UA-content-id parameter is limited in length to 16 characters.)

*Note 2* – There are many different ways in which the recipient may access and receive the IP-message. For example, the UAE may automatically provide the IP-message to the recipient. Alternatively, a UAE may provide the recipient with the capability to selectively (for example, at any time and in any order) receive IP-messages which have been delivered to its UAE.

TABLE 2/X.420

**Summary of MTL Primitives Associated with Access Management**

| Primitive | Reference in Rec. X.411 |
|---|---|
| (UAL) LOGON.Request | § 2.2.1 |
| (UAL) LOGON.Confirmation | § 2.2.1 |
| (MTL) LOGON.Indication | § 2.2.2 |
| (MTL) LOGON.Response | § 2.2.2 |
| LOGOFF.Request | § 2.2.3 |
| LOGOFF.Confirmation | § 2.2.3 |
| REGISTER.Request | § 2.2.4 |
| REGISTER.Confirmation | § 2.2.4 |
| (UAL) CHANGE-PASSWORD.Request | § 2.2.12 |
| (UAL) CHANGE-PASSWORD.Confirmation | § 2.2.12 |
| (MTL) CHANGE-PASSWORD.Indication | § 2.2.13 |

### 4.2 *Provision of optional IPM service elements invoked in Sending/Receiving IP-messages*

A number of IPM service elements exist which can be selected or invoked by the user as options of sending and/or receiving an IP-message. The operation of the UAEs in providing each of these service elements is described in terms of actions the UAEs perform in addition to those already being performed for the basic service of sending and receiving.

#### 4.2.1 *Provision of the Submission, Delivery and Conversion MT service elements*

The Submission, Delivery and Conversion MT service elements are described in §§ 4.1.2-4.1.3 of Recommendation X.400. With the exception of Deferred Delivery Cancellation, each of these service elements is provided as an option of the basic service of sending and receiving IP-messages, as described in the following sections. Provision of Deferred Delivery Cancellation is described in § 4.3.1.

TABLE 3/X.420

**Parameters used in MTL SUBMIT.Request primitive**

| Parameter | **A**: IM-UAPDU (basic service) | **B**: SR-UAPDU | **C**: IM-UAPDU (auto-forwarding) |
|---|---|---|---|
| Recipient-O/R-names | O/R-names of UAEs to whom IM-UAPDU is being sent | O/R-name of originator of IM-UAPDU being reported on | O/R-name of UAE to whom IM-UAPDU is being auto-forwarded |
| Originator-O/R-name | O/R-name of UAE sending the IM-UAPDU | O/R-name of this recipient of IM-UAPDU being reported on | O/R-name of UAE auto-forwarding the IM-UAPDU |
| Content | IM-UAPDU | SR-UAPDU | IM-UAPDU |
| Content-type | Indicates "P2" | Indicates "P2" | Indicates "P2" |
| Encoded-information-types | Set according to the body part type of the IM-UAPDU body | Unspecified (thus precluding conversion) | Unspecified (thus precluding conversion) |
| NDN-suppress | See Note a | Yes | Yes |
| Priority | See Note a | Same as for IM-UAPDU being reported on (from DELIVER.Indication) | Same as for IM-UAPDU being auto-forwarded (from DELIVER.Indication) |
| Deferred-delivery-time | See Note a | – | – |
| Delivery-notice | See Note a | No | No |
| Conversion-prohibited | See Note a | Yes | Yes |
| Disclose-recipients | No | No | No |
| Alternate-recipient-allowed | See Note a | No | No |
| Content-return | See Note a | No | No |
| UA-content-id | See Note b | – | – |
| Explicit-conversion | See Note a | – | – |

*Note a* – These parameters are not mandatory in the MTL SUBMIT.Request, and are not related to the basic MT Service. Thus they are not specified in the SUBMIT.Request when the basic IPM Service is provided. (They may, however, be specified in the provision of the Submission and Delivery and Conversion Message Transfer service elements, as described in § 4.2.1.)

*Note b* – See Note 1 in § 4.1.2.

4.2.1.1 *Grade of Delivery, Multi-destination Delivery, Disclosure of Other Recipients, and Explicit Conversion*

The MT service elements Grade of Delivery, Multi-destination Delivery, Disclosure of Other Recipients and Explicit Conversion are made available to IPM Service users by cooperating originator and recipient UAE actions as follows:

1) The originator's UAE specifies the corresponding parameter(s) of the SUBMIT.Request primitive when the IM-UAPDU is sent.
2) The recipient's UAE makes the corresponding parameter(s) of the DELIVER.Indication primitive accessible to the recipient when the IP-message is received.

Table 4/X.420 lists the service elements in this set, along with the corresponding parameters.

*Note* – In the case that Multi-destination Delivery is requested, a number of NOTIFY.Indication events relating to one IM-UAPDU may be generated by the MTL. Any such notification may refer to delivery or non-delivery to one or more of the originally specified recipients and must be correlated with the original IM-UAPDU as described in Note 1 in § 4.1.

TABLE 4/X.420

**Service Elements and Corresponding Parameters**

| Service element | Parameter of SUBMIT.Request | Parameter(s) of DELIVER.Indication |
|---|---|---|
| Grade of Delivery Selection | Priority | Priority |
| Multi-destination Delivery | Recipient-O/R-names | Other-recipient-O/R-names (see also Note a) |
| Disclosure of Other Recipients | Disclose-recipients | Other-recipient-O/R-names (see also Note b) |
| Explicit Conversion | Explicit-conversion | Original-encoded-information-types<br>Converted-encoded-information-types |

*Note a* – When multi-destination delivery is invoked, other recipients will be indicated to each recipient only if the Disclosure of Other Recipients service element was also invoked.

*Note b* – The presence of this parameter indicates that Disclosure of Other Recipients was selected by the originator and provided by the MTL.

4.2.1.2 *Deferred Delivery, Prevention of Non-delivery Notification, and Conversion Prohibition*

Each of the MT service elements Deferred Delivery, Prevention of Non-delivery Notification, and Conversion Prohibition is made available to IPM Service users by means of the originating UAE's action. For each of these elements, the originating UAE specifies a corresponding parameter of the SUBMIT.Request primitive when the IM-UAPDU is sent. The associated parameters are shown in Table 5/X.420.

4.2.1.3 *Implicit Conversion*

The Implicit Conversion MT service element is made available to IPM Service users by means of the recipient UAE's actions. A DELIVER.Indication containing an IM-UAPDU upon which implicit conversion has been performed will convey an original-encoded-information-types parameter. This information may be made accessible to the recipient when the IP-message and its other associated information are received.

TABLE 5/X.420

**Service Elements and Corresponding Parameters**

| Service element | Corresponding SUBMIT.Request parameter |
|---|---|
| Deferred Delivery | Deferred-delivery-time |
| Prevention of Non-delivery Notification | NDN-suppress (see Note a) |
| Conversion Prohibition | Conversion-prohibited |

*Note a* – If the IM-UAPDU is being transferred to more than one recipient, a different value for this parameter may be specified for each individual recipient.

4.2.1.4 *Alternate Recipient Allowed*

To make the Alternate Recipient Allowed MT service element available to users of the IPM Service, the following originator and recipient UAE actions are performed:

1) The originating UAE supplies the alternate-recipient-allowed parameter in the SUBMIT.Request primitive when the IM-UAPDU is sent.
2) When an IM-UAPDU is actually delivered to an alternate recipient's UAE, the DELIVER.Indication primitive will indicate the O/R name of the originally intended recipient in the intended-recipient-O/R-name parameter.
3) If, as a result of this service element being requested, an attempt is made to deliver to an alternate recipient, and this attempt fails, an appropriate NOTIFY.Indication, signifying non-delivery, will be issued to the originator's UAE (unless Prevention of Non-delivery Notification was requested; see § 4.2.1.2). This has parameters which identify the intended as well as the alternate recipient (intended-recipient-O/R-name and recipient-O/R-names), and this information must be available to the originator.
4) Where delivery actually takes place to an alternate recipient, and where delivery notification (see § 4.2.1.6) has also been requested, a NOTIFY.Indication signifying delivery will be issued to the originator's UAE. The recipient-O/R-names parameter will reflect the actual (alternate) recipient, while the intended-recipient-O/R-name parameter will be that originally supplied with the SUBMIT.Request.
5) Other actions by the alternate recipient's UAE are required if a receipt or non-receipt notification for the IM-UAPDU must be generated (see §§ 4.2.2.2-4.2.2.3).

4.2.1.5 *Delivery Notification*

The Delivery Notification MT service element is made available to IPM Service users by the following originating UAE actions:

1) The UAE requests the notification by specifying the delivery-notice parameter of the SUBMIT.Request primitive when the IM-UAPDU is sent.
2) If delivery of the IM-UAPDU occurs and is signalled by means of a NOTIFY.Indication primitive to the originating UAE, the UAE must make the originator aware that the notification has occurred. The notification information which is conveyed by the NOTIFY.Indication (recipient-O/R-names, delivery-time, and type-of-UA parameters and, if applicable, converted-encoded-info-types, intended-recipient-O/R-name, and supplementary-information parameters) must be available to the originator.

*Note* – It must be possible to correlate a notification with the previous IM-UAPDU to which it refers. Examples of different mechanisms to achieve this correlation are described in Note 1 of § 4.1 of this Recommendation.

4.2.1.6 *Return of Content*

The Return of Content MT service element is made available to IPM Service users by the following UAE actions:

1) The UAE specifies the request for the service in the corresponding content-return parameter of the SUBMIT.Request primitive when the IM-UAPDU is sent.

2) If delivery of the IM-UAPDU fails, the returned-content parameter of the NOTIFY.Indication primitive will convey the returned IM-UAPDU. The UAE must make the information in the returned IM-UAPDU available to the originator. (Prevention of Non-delivery Notification must not have been selected when the IM-UAPDU was submitted.)

4.2.2 *Provision of the Cooperating UA Action service elements*

The Cooperating UA Action service elements are described in § 4.2.3 of Recommendation X.400.

4.2.2.1 *Blind Copy Recipient Indication*

To provide this service element, the originator's UAE:

1) Constructs two (or possibly more) IM-UAPDUs in the P2 syntax. These IM-UAPDUs are identical except for the inclusion of and contents of the **blindCopyRecipients** component.

2) One constructed IM-UAPDU, which does not contain any **blindCopyRecipients** component, is sent by issuing a SUBMIT.Request primitive to only those recipients specified as primary and copy recipients of the IP-message (that is, only the O/R names of the primary and copy recipients are supplied in the recipient-O/R-names parameter of the SUBMIT.Request).

3) The blind copy recipients are sent a different IM-UAPDU. The extent to which the blind copy recipients are disclosed to one another is a local matter. For example, the UAE may send the same IM-UAPDU to all the blind copy recipients, may send a separate IM-UAPDU to each blind copy recipient individually, or may choose some other grouping. In addition, in the **blindCopyRecipients** component of the IM-UAPDU being sent to some grouping of blind copy recipients, the UAE may insert only the names of the blind copy recipients receiving that particular copy of the IM-UAPDU, or the names of all the blind copy recipients, or some other combination of blind copy recipient names.

When the IM-UAPDUs are delivered by means of the DELIVER.Indication primitive, the IM-UAPDUs sent to UAEs of the primary and secondary recipients do not contain the **blindCopyRecipients** component and thus these recipients are not aware of the blind copy recipients. The UAE of each blind copy recipient is informed of some set of the blind copy recipients including himself by means of the **blindCopyRecipients** component which the originator's UAE supplied to him in his copy of the IM-UAPDU.

4.2.2.2 *Non-receipt Notification*

Provision of Non-receipt Notification requires action by both the originator's and the intended recipient's UAE:

1) When the originator requests non-receipt notification for a particular recipient of an IP-message (either in addition to, or independent of, receipt notification), the originator's UAE specifies the request in the **reportRequest** component for that recipient in the IM-UAPDU which is sent to the recipient by means of a SUBMIT.Request primitive. This may include a request to return the IP-message with the non-receipt notification.

2) The IM-UAPDU will be delivered to each of its recipients' UAEs by means of a DELIVER.Indication primitive. For each such recipient for which non-receipt notification has been requested, the UAE will generate a non-receipt notification upon the following conditions:

   a) In the event that the IM-UAPDU is not received by the originator-specified recipient but is auto-forwarded by the recipient's UAE to another UAE.

   b) In the event that, before the recipient receives the IP-message, the UAE disposes of the IM-UAPDU so that there is no potential for the IP-message ever to be received.

3) In either case, to generate the notification, the UAE constructs an IPM Status Report (SR-UAPDU) including as components the **IPMessageId** which is identical to that of the IM-UAPDU being reported on, and information relating to the reason for non-receipt (see Column A of Table 6/X.420). The UAE also includes the **intendedRecipient** component if, and only if, the actual recipient differs from that specified by the originator. This component carries the O/R name which was conveyed in the intended-recipient-O/R-name parameter of the DELIVER.Indication. If return of the IP-message was requested, then the IM-UAPDU is conveyed as the **returned** component of the SR-UAPDU. The UAE issues a SUBMIT.Request primitive to send the constructed SR-UAPDU to the originator, supplying this IPM status report as the content parameter. The other parameters of the SUBMIT.Request primitive are specified as shown in Column B of Table 3/X.420.

4) At the originator's UAE, a DELIVER.Indication from the MTL conveys the SR-UAPDU. The UAE must make the originator aware that the non-receipt notification has occurred, and the information in the components of the IPM status report must be available to the originator.

*Note* – In the format of the IM-UAPDU, the **reportRequest** component is a part of the **Recipient** component, which also includes the **ORDescriptor** (see § 3.2.1). When the **reportRequest** is included, the **ORDescriptor** component must include the recipient's **ORName**. The O/R name is required in this component because in its absence it would be difficult or impossible for the recipient UAE to recognize and extract the **Recipient** component of the IM-UAPDU which applies to him. Thus the recipient UAE would be unable to complete the intent of this notification service.

4.2.2.3 *Receipt Notification*

Provision of the Receipt Notification service element requires action by both the originator's UAE and the recipient's UAE (and possibly by the recipient himself):

1) When the originator requests Receipt Notification (note that this request can be specified individually for each recipient of the IM-UAPDU and that a request for recipt notification also implies a request for non-receipt notification), the originator's UAE specifies the request in the **reportRequest** component for that recipient in the IM-UAPDU which is sent to the recipient by means of a SUBMIT.Request primitive.

2) The IM-UAPDU will be delivered to each of its recipients' UAEs by means of a DELIVER.Indication primitive. For each such recipient for whom receipt notification has been requested, one of two situations can exist. In one case, when the IM-UAPDU is received by the recipient, the UAE will automatically generate a SR-UAPDU to convey the receipt notification and send it back to the originator. In the alternative case, the UAE will inform the recipient, when he receives the IP-message, that receipt notification has been requested. The UAE will wait until the recipient has performed some explicit action authorizing return of the receipt notification before the SR-UAPDU conveying the notification is actually generated and sent.

3) In either case, to generate the notification, the UAE constructs an SR-UAPDU including components indicating the **Reported IPMessageId** of the IM-UAPDU to which the notification applies, the **typeOfReceipt** (for example, automatic or explicit, as described in Action 2 above) and the **receipt Time** (see Column B of Table 6/X.420). The **intendedRecipient** component is also included if, and only if, the actual recipient differs from that specified by the originator. It carries the **ORDescriptor** which was conveyed in the **Recipient** component of the delivered IM-UAPDU which corresponds to the intended-recipient-O/R-name parameter of the DELIVER.Indication. To send the notification, the UAE issues a SUBMIT.Request primitive with the SR-UAPDU as the content. The other parameters of the SUBMIT.Request primitive are specified as shown in Column B of Table 3/X.420.

4) At the originator's UAE, a DELIVER.Indication from the MTL conveys the SR-UAPDU. The UAE must make the originator aware that the receipt notification has occurred, and the information included in the **receipt Time** and **supplementaryInformation** components of the SR-UAPDU must be available to the originator.

*Note* – In the format of the IM-UAPDU, the **ORDescriptor** component which accompanies the receipt notification request must include the recipient's O/R name (see Note in § 4.2.2.2).

TABLE 6/X.420

**Components of SR-UAPDU**

| Components | A: Non-receipt | B: Receipt | C: Auto-forwarded Non-receipt |
|---|---|---|---|
| **reported IPMessageId** | See Note a | See Note a | See Note a |
| **actualRecipient** | See Note b | See Note b | See Note b |
| **intendedRecipient** | See Note c | See Note c | See Note c |
| **Converted** | See Note d | See Note d | See Note d |
| **receipt Time** | – | UAE-supplied time of receipt | – |
| **typeOfReceipt** | – | Indicates "explicit" or "automatic" | – |
| **reason** | Indicates "UAE initiated discard" | – | Indicates "auto-forwarded" |
| **supplementaryInformation** | – | UAE-supplied supplementary information | – |
| **nonReceiptQualifier** | Indicates "expired" "obsoleted", or "subscription terminated" | – | – |
| **Comments** | – | – | Comments by intended recipient if supplied |
| **returned** | IM-UAPDU being reported upon if requested by originator | – | IM-UAPDU being reported upon if requested by originator |

*Note a* – This is the same identifier conveyed in the IM-UAPDU to which the report refers.

*Note b* – O/R Descriptor of the user whose UAE actually accepted delivery of the IM-UAPDU being reported on and is generating this report.

*Note c* – O/R Descriptor of the originally intended recipient, obtained from the corresponding **Recipient** component of the IM-UAPDU being reported upon.

*Note d* – If conversion occurred in the UAL, this parameter is the new **EncodedInformationTypes**. If conversion occurred in the MTL but not in the UAL, this is the converted-encoded-information-types parameter of the DELIVER.Indication that conveyed the IM-UAPDU being reported upon.

4.2.2.4 *Auto-forwarded Indication*

Provision of the Autoforwarded Indication service element requires the following cooperating UA actions:

1) When a recipient UAE receives an IM-UAPDU by means of a DELIVER.Indication primitive, the fact that the body contains an auto-forwarded IP-message (IM-UAPDU) is indicated by the **auto-forwarded** IM-UAPDU component. If the auto-forwarded IP-message contains a request for receipt and/or non-receipt notification, these requests are ignored by the recipient UAE to which the IP-message is auto-forwarded, and no notification is generated.
2) It is possible that the recipient UAE itself has been instructed to perform auto-forwarding. However, to guarantee that "looping" of auto-forwarded IP-messages does not occur, the UAE must not auto-forward such an IM-UAPDU containing an auto-forwarded body part.
3) If the UAE receives an IM-UAPDU which contains an auto-forwarded IP-message and is not performing auto-forwarding itself, then it must indicate to the recipient when the IP-message is received that it contains an IP-message in the body.

4.2.3 *Provision of the Cooperating UA Information Conveying service elements*

The Information Conveying service elements are described in § 4.2.4 of Recommendation X.400.

4.2.3.1 *Information Conveying service elements applying on a per-IP-message basis*

A number of the cooperating UA Information Conveying service elements are applicable on a per-IP-message basis. That is, when the service element is specified, it applies to the IP-message and all of its recipients. This type of service element is distinguished from the type of service element which can be specified on a per-recipient basis, that is, individually for each of the multiple recipients of a single IP-message.

The Information Conveying elements which are in the per-IP-message category are the following:

- Originator Indication
- Authorizing Users Indication
- Primary and Copy Recipients Indication
- Expiry Date Indication
- Cross-referencing Indication
- Importance Indication
- Obsoleting Indication
- Sensitivity Indication
- Subject Indication
- Replying IP-message Indication
- Forwarded IP-message Indication
- Body Part Encryption Indication.

To provide the Forwarded IP-message Indication and Body Part Encryption Indication, the originator's UAE includes the appropriate body parts of types (**Forwarded IPMessage** and **Encrypted**) in the IM-UAPDU which is constructed to be sent to the recipient's UAE. For each of the other service elements of this type, the originator's UAE includes the corresponding heading component (see Table 1/X.420) in the IM-UAPDU. The IM-UAPDU is sent to the recipient's UAE by means of a SUBMIT.Request.

The IM-UAPDU arrives at each recipient's UAE by means of the DELIVER.Indication primitive. The UAE then conveys the corresponding information from the IM-UAPDU to the recipient when the IP-message is received.

4.2.3.2 *Reply Request Indication*

The Reply Request Indication service element can apply to each recipient individually in a multi-recipient IP-message. To provide this service:

1) The originator's UAE includes in the IM-UAPDU the **replyRequest** component for each specified recipient for which the Reply Request is being invoked.

2) For each recipient to which the IM-UAPDU is delivered, the recipient UAE must extract the **replyRequest** component if present which relates to itself and convey this information to the recipient when the IP-message is received.

*Note* – In the format of the IM-UAPDU, the **replyRequest** component is a part of the **Recipient** component which also includes the O/R Descriptor (see § 3.2.1). When the **replyRequest** is included, the O/R Descriptor component must include the recipient's O/R name. The O/R name is required in this component because in its absence the recipient UAE might not be able to recognize and extract the **Recipient** component of the IM-UAPDU which applies to him. The UAE would thus not be able to complete the intent of the Reply Request Indication service element.

3) If the originator specifies the date by which the reply is to be sent and/or a list of user(s) to whom the reply is to be sent, the originator's UAE includes the appropriate component(s) (**replyBy** and **replyToUsers**) in the IM-UAPDU. These extracted and communicated to the recipient by the recipient's UAE when the IP-message is received. The **replyToUsers** component, when present, includes all of the O/R names of the users to whom the reply is to be sent.

#### 4.2.3.3 *Multi-part body*

The Multi-part Body service element provides the capability for the body of the IP-message to consist of several different parts which may be of different body types. In providing this service element:

1) The originator's UAE constructs the IM-UAPDU by including the various body parts along with their type indication in the **Body** component.

2) When the UAE sends the IM-UAPDU by issuing the SUBMIT.Request primitive, it must specify in the encoded-information-types parameter the types of the multiple body parts, including the "OR"-ed non-basic parameters for the body part types which have such characteristics (for example, teletex and G3 facsimile). For example, if the body consists of multiple teletex and G3 facsimile body parts, the non-basic parameters applying to each will be embedded in each body part. The originating UAE must "OR" the non-basic parameters for the teletex body parts and must "OR" the non-basic parameters for the G3 facsimile body parts to obtain a description of the "worst case" situation regarding body type compatibility. These "OR"-ed values for the two body types are supplied in the encoded-information-types parameter of the SUBMIT.Request.

3) When an IM-UAPDU with a multi-part body is delivered to a recipient UAE, and if no conversions have been performed in the MTS, then the encoded-information-types parameter will be the same as that supplied by the originating UAE. If some conversion has taken place, the encoded-information-types parameter will indicate the new body part types resulting from the conversion(s), and the original-encoded-information-types parameter will indicate what type of body parts were originally sent.

*Note* – Since a mapping of the original encoded information types to the corresponding converted body parts is not provided, it will not always be possible for the recipient to determine which body parts were actually converted.

4) When an IP-message consists of a single G3 facsimile or teletex body part, the non-basic parameters need not be duplicated in the **BodyPart** component of the IM-UAPDU but can be conveyed in the encoded-information-types parameter of the SUBMIT.Request primitive when the IM-UAPDU is sent. Upon delivery of the IM-UAPDU, the non-basic parameters are supplied to the recipient's UAE in the encoded-information-types parameter of the DELIVER.Indication primitive. Absence of the non-basic parameters from both the **BodyPart** component of the IM-UAPDU and the encoded-information-types parameters (when the encoded-information-types parameters indicate a G3 facsimile or teletex body part) indicates that the body part is of the basic G3 or teletex type.

### 4.3 *Provision of optional IPM service elements invoked independently of Sending/Receiving IP-messages*

A set of the IPM service elements are not selected in association with the basic service of sending and receiving an IP-message, but provide other capabilities and are invoked independently. This set of elements consists of the four MT service elements: Deferred Delivery Cancellation, Probe, Hold for Delivery, and Alternate Recipient Assignment.

4.3.1 *Deferred Delivery Cancellation*

The Deferred Delivery Cancellation service element (described in § 4.1.2.10 of Recommendation X.400) is provided through the actions of the UAE of the originator requesting cancellation of a previously submitted IP-message for which deferred delivery was requested. To perform this service element, the originator UAE's actions are as follows:

1) If Blind Copy Recipients Indication was not requested when the IP-message for which cancellation is desired was originally sent, then the UAE issues a single CANCEL.Request primitive; the submit-event-id parameter identifies the IM-UAPDU conveying the IP-message which is being cancelled (see note below).

2) In the case that Blind Copy Recipients Indication was specified when the IP-message was sent, the UAE issues a CANCEL.Request corresponding to each of the two (or possibly more) IM-UAPDUs which were submitted to the MTL and each of which conveys a copy of the IP-message (see § 4.2.2.1); each CANCEL.Request conveys a submit-event-id parameter which identifies an IM-UAPDU being cancelled (see note below).

3) The MTL will indicate the results of the cancellation attempt by means of a CANCEL.Confirmation primitive (or multiple CANCEL.Confirmation primitives, where Blind Copy Recipients were specified). The UAE must utilize the submit-event-id parameter returned by the CANCEL.Confirmation primitive to determine to which previously issued CANCEL.Request the Confirmation applies. For IP-messages sent with Blind Copy Recipients, the results of the cancellation attempt may not be the same for all of the specified recipients. Thus the UAE must also be able to correlate (via the submit-event-id) the individual CANCEL primitives with the O/R name of the intended recipient of each IM-UAPDU conveying a copy of the IP-message. The originator must be informed about the results of the cancellation attempt for each recipient. In the event of failure to cancel delivery of the IP-message to a recipient, the failure reason (specified by the failure-reason parameter of the CANCEL.Confirmation primitive) must also be made available to the originator.

*Note* – Since use of the submit-event-id is the only mechanism available for identifying the IM-UAPDUs to be cancelled, the UAE must maintain these (possibly multiple) identifiers corresponding to any IP-message for which it intends to provide Deferred Delivery Cancellation.

4.3.2 *Probe*

The Probe service element (described in § 4.1.4.1 of Recommendation X.400 is provided through the UAE of the IPMS user who initiates the Probe request. In providing this service element:

1) The UAE issues a PROBE.Request primitive to the MTL supplying the parameter values as indicated in Table 7/X.420.

2) If the PROBE.Confirmation primitive returned by the MTL indicates that the request for a PROBE was not accepted, then the UAE must make the Probe originator aware of the failure. The failure reason parameter indicated by the primitive must also be available to the originator.

3) If the PROBE.Confirmation primitive indicates that the PROBE.Request was accepted, the Confirmation provides a probe-time and a probe-event-id parameter which can be used to correlate further notifications with this Probe (see Note below).

4) For a PROBE which was accepted, the UAE will receive a NOTIFY.Indication primitive signalling either "delivery" or "non-delivery"; if delivery is indicated, the UAE should inform the PROBE originator that an IP-message submitted with the same set of parameters as the PROBE could have been successfully delivered; if "non-delivery" is indicated, the UAE should inform the originator that a submitted IP-message with the same parameters would have resulted in a "non-delivery".

*Note* – It must be possible to correlate a notification with the previous PROBE to which it refers. Examples of different mechanisms to achieve correlation are described in Note 1 of § 4.1 of this Recommendation.

4.3.3 *Hold for Delivery*

The Hold for Delivery service element is provided through the UAE of the recipient who is requesting that his IP-messages be held (or released) for delivery. To provide this service element:

1) The UAE issues a CONTROL.Request primitive, supplying the appropriate parameter values (permissible-encoded-information-types, maximum-content-length, lowest-priority-msgs, notifications) which correspond to the recipient's request.

2) The UAE must inform the recipient of the results of the request for this service element, as indicated by the CONTROL.Confirmation primitive returned by the MTL. The information conveyed by the messages-held parameter of the CONTROL.Confirmation must also be available to the recipient.

TABLE 7/X.420

**Parameters used in MTL PROBE.Request primitive**

| Parameter | Value Specified by UAE |
|---|---|
| Recipient-O/R-names | O/R name of recipient(s) specified by originator |
| Originator-O/R-name | O/R name of UAE (user) issuing the Probe |
| Content-type | Indicates "P2" |
| Encoded-information-types | As specified by originator |
| Conversion-prohibited | As specified by originator |
| Alternate-recipient-allowed | As specified by originator |
| Content-length | As specified by originator |
| UA-content-id | As specified by originator |

4.3.4 *Alternate Recipient Assignment*

The MT service element of Alternate Recipient Assignment (see § 4.1.5.1 of Recommendation X.400) will not, in general, be made available to every user of the IPM Service. The criteria for allowing UAEs to be assigned as alternate recipients and the procedures for performing such assignment are determined by the Management Domain providing the MT Service. These procedures may, in some cases, involve management interactions between the MTL and the UAE but no MTL primitives or specific IPM UAE operations for this purpose are currently defined.

Following assignment of a UAE as an alternate recipient, messages will be delivered to the UAE if the agreed upon conditions are met. § 4.2.1.4 describes the UAE actions associated with actual occurrence of delivery to an alternate recipient.

4.4 *IPM UAE management operations*

There are certain functions, necessary for the proper operation of the IPMS, which UAEs must perform in addition to those functions related directly and/or exclusively to the provision of a particular IPM service element. Such "management functions" may require interaction between cooperating UAEs or between a UAE and the MTL (see note below).

The principal such function of IPM UAEs is to deal with MTL-invoked "flow control", which is utilized by the MTL as a means of managing and protecting its resources. Such flow control is represented in the MTL Service by issuance of the (MTL) CONTROL.Indication primitive (see § 2.2.6 of Recommendation X.411), which conveys information (by means of the maximum-content-length, lowest-priority, probes, and msgs parameters) instructing the UAE concerning the maximum length, priority, and type of messages it is willing to accept. Upon receiving such an indication, a UAE should adjust its operation to observe the new constraints imposed by the MTL and should issue a CONTROL.Response primitive, signalling by means of the messages-held parameter whether, as a result of the new restrictions, it has messages or probes that it must wait to submit.

*Note* – There may be, of course, UAE management functions that involve no interaction with other entities. Like all strictly local functions, these are not addressed in these MHS Recommendations.

## 5 Simple Formattable Documents

### 5.1 *Introduction*

An important application of the Interpersonal Messaging Service is the exchange of what can be described as formattable (but not yet formatted) documents. Since the physical characteristics (for example, line length) of user's equipment varies widely, information formatted for the originator's ouput device may appear quite different – even nearly unreadable – when output on a recipient's device.

The problem described above is solved by permitting the originator to specify the *logical* structure of a document but not its detailed *layout* structure. As part of his description of the former, the originator may specify any aspects of the layout he feels he must control to preserve the full meaning of the document. At the same time, however, he delegates to the recipient (more precisely, to his UA) full responsibility for all the remaining layout decisions. An example of the result of providing such a capability is illustrated in Figures 5/X.420 and 6/X.420. A document that appears to the originator as depicted in Figure 5/X.420 would appear to a recipient as depicted in Figure 6/X.420.

This section defines a *Simple Formattable Document (SFD)* suitable as a basis for the exchange of simple, character-encoded textual information. An SFD can be conveyed as a particular body type in the body of IP-messages exchanged by means of the IPM Service. § 5.2 describes the abstract structure of such a document. § 5.3 describes its concrete encoding. § 5.4 gives an example of both.

The establishment in various countries of telematic services and computer-based store-and-forward message services in association with public data networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT, considering:

(a) the need for interpersonal messaging and message transfer services;

(b) the need to transfer messages of different types having a large variety of formats;

FIGURE 5/X.420

**An example of a four-paragraph document as rendered by the originator's UA**

The establishment in various countries of telematic services and computer-based store-and-forward message services in association with public data networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT, considering:

(a) the need for interpersonal messaging and message transfer services;

(b) the need to transfer messages of different types having a large variety of formats;

FIGURE 6/X.420

**The same four-paragraph document illustrated in Figure 5/X.420 as rendered by a recipient's UA**

### 5.2 *Structure of a Simple Formattable Document*

A Simple Formattable Document comprises character-encoded information organized as a sequence of paragraphs. The abstract structure of such a document is described in detail in this section. Both the overall structure of the document and the structure of its component parts – called content portions and specific logical objects – are described.

5.2.1 *Overall structure*

A (simple formattable) *document* is a series of *paragraphs*, each of which in turn is a series of *content portions*. The only form of content portion currently defined, called *character text*, is a series of characters. A document and each of its paragraphs are referred to as *specific logical objects*. (Generic logical objects, which describe classes of specific logical objects, are not supported in SFDs.)

Each content portion or specific logical object has one or more *attributes*, each of which is either variable or unspecified, as summarized in Table 8/X.420. A *variable* attribute is one that can be controlled by the entity that constructs the document. An *unspecified* attribute is one that can be controlled only by the entity that renders the document (that is, outputs to some physical medium for presentation). The value of an unspecified attribute is also *known* only to the rendering entity and, in general, varies from one such entity to the next. Unspecified attributes lie at the heart of the SFD concept, since they provide the rendering entity with the freedom it needs to render the document on a particular output medium. In the context of the IPMS, the entity that constructs the SFD is the originator's UAE. The rendering entity is the UAE of the recipient.

TABLE 8/X.420

**SFD Protocol elements and what they represent**

| Protocol element | Variable attributes | Exemples of unspecified attributes |
|---|---|---|
| Document<br>(a logical object) | Page heading | Output medium dimensions<br>Top, bottom, left and right margins<br>Presence and position of page numbers<br>Number of columns per page<br>How page breaks are chosen |
| Paragraph<br>(a logical object) | Alignment<br>Left indentation<br>Bottom blank lines<br>Graphic rendition | Line height<br>How line breaks are chosen |
| Character Text<br>(a content portion) | Graphic rendition | Font<br>Pitch |

5.2.2 *Content portions*

A content portion is a portion of the document (for example, a fragment of character text) that must be output as part of the rendering process. It embodies syntactic, rather than semantic, information. (The latter is conveyed by means of specific logical objects; see § 5.2.3.)

5.2.2.1 *Character text*

A character text content portion is an ordered set of zero or more of the graphic characters defined by Recommendation T.61. Like other characters, spaces are rendered wherever they occur. Also admitted are the character sequences CR-LF, which begins a new line, and CR-FF, which begins a new page.

*Note* – These character sequences are included to permit tabular or other special formats that cannot be achieved by the current SFD structural elements. However, use of these sequences is discouraged where not absolutely necessary as it deprives the recipient of the flexibility provided by SFD representation.

A character text content portion has one variable attribute, *graphic rendition*, which can have either of two values, normal and underlined.

Among a character text content portion's unspecified attributes are font and pitch.

### 5.2.3 *Specific logical objects*

A specific logical object aggregates content portions or other specific logical objects so as to influence the output of the former during the rendering process. It embodies semantic, rather than syntactic, information. Two specific logical objects, document and paragraph, are defined below.

#### 5.2.3.1 *Document*

A *document* is an ordered set of zero or more paragraphs.

A document has one variable attribute, *page heading*, which is an ordered set of zero or more of the graphic characters defined by Recommendation T.61. The page heading appears within the top margin of every page (or screen's worth) of the rendered document.

Among a document's unspecified attributes are the dimensions of the output medium on which it is to be rendered; the size of the top, bottom, left and right margins of the page (or screen); whether and where page numbers appear on the page; the number of columns on the page; and exactly how page breaks are positioned with respect to paragraph boundaries.

#### 5.2.3.2 *Paragraph*

A *paragraph* is an ordered set of zero or more character text content portions and has four variable attributes.

The *alignment attribute* indicates how the characters rendered on each line of the paragraph are to be positioned with respect to the page's left and right margins. *Left alignment* signifies that the first character of each line is to appear at the left margin and that no attempt should be made to place the last character of the line at the right margin. *Centered* signifies that the first and last characters should be equi-distant from the left and right margins, respectively. *Justified* signifies that the first and last characters are to be placed at the left and right margins, respectively.

The *left indentation* attribute indicates by how many character boxes the paragraph's left margin is to exceed the fixed left margin that would otherwise be chosen when rendering the paragraph.

The *bottom blank lines* attributes indicate whether zero or one blank line is to be rendered following the last line of the paragraph.

The *graphic rendition* attribute indicates whether the initial graphic rendition for each of the paragraph's component content portions is to be normal or underlined.

Among a paragraph's unspecified attributes are line height and how line breaks are positioned with respect to word boundaries.

## 5.3 *Specification of SFD Document Interchange Format*

When transferred between open systems (for example, by means of the IPM Service), a Simple Formattable Document is represented as a sequence of octets. The structure of such a sequence of octets – the *SFD Document Interchange Format (SFD-DIF)* – is described in this section. It is the SFD-DIF that governs the SFD body part type contained in IP-messages transferred through the IPMS (see § 3.2.2).

The SFD-DIF conforms to Recommendation X.409. Thus the standard notation for data types and values is used in its description. The resulting formal definition of the SFD-DIF is summarized in Figure 7/X.420 and explained piece by piece in the subsections that follow. The binary representation of an SFD-DIF instance can be inferred from the notational description with the aid of Recommendation X.409.

```
SFD DEFINITIONS ::=
BEGIN

Document                ::= SEQUENCE OF ProtocolElement

ProtocolElement         ::= CHOICE {
                                textUnit [3] IMPLICIT TextUnit,
                                specificLogicalDescriptor [5] IMPLICIT LogicalDescriptor}

-- text units

TextUnit                ::= SEQUENCE {
                                contentPortionAttributes ContentPortionAttributes,
                                textInformation TextInformation}

ContentPortionAttributes ::= SET {-- none at present--}

TextInformation         ::= CHOICE {T61String}

-- logical descriptor

LogicalDescriptor       ::= SEQUENCE
                                {LogicalObjectType,LogicalDescriptorBody

LogicalObjectType       ::= INTEGER {document(0), paragraph(1)}

LogicalDescriptorBody   ::= SET {
                                -- variable attributes (if object is document)
                                   pageHeading [3] IMPLICITT61String OPTIONAL,
                                -- variable attributes (if object is paragraph)
                                   layoutDirectives [4] IMPLICIT LayoutDirectives OPTIONAL,
                                   presentationDirectives [5] IMPLICIT
                                       PresentationDirectives OPTIONAL,
                                -- default variable attributes for subordinate objects (if any)
                                   defaultValueLists [6] IMPLICIT SEQUENCE {
                                       defaultValueList} OPTIONAL}

LayoutDirectives        ::= SET {
                                leftIndentation [0] Offset OPTIONAL,
                                bottomBlankLines [3] Offset OPTIONAL}

Offset                  ::= CHOICE {
                                [1] IMPLICIT INTEGER}

PresentationDirectives  ::= SET {
                                alignment [0] IMPLICIT Alignment OPTIONAL,
                                graphicRendition [1] IMPLICIT GraphicRendition OPTIONAL}

Alignment               ::= INTEGER {leftAligned(0), centered(2), justified(3)}

GraphicRendition        ::= SEQUENCE OF GraphicRenditionAspect

GraphicRenditionAspect  ::= INTEGER-- an SGR parameter value; see Recommendation T.61

DefaultValueList        ::= CHOICE {
                                paragraphAttributes [1] IMPLICIT ParagraphAttributes}
```

FIGURE 7/X.420 (Part 1 of 2)

**Formal definition of the SFD Document Interchange Format**

```
ParagraphAttributes     ::= SET {
                                layoutDirectives < Attribute OPTIONAL,
                                presentationDirectives < Attribute OPTIONAL}
Attribute               ::= CHOICE {
                                layoutDirectives [0] IMPLICIT LayoutDirectives,
                                presentationDirectives [1] IMPLICIT PresentationDirectives}
END
```

FIGURE 7 (Part 2 of 2)

**Formal definition of the SFD Document Interchange Format**

5.3.1 *General structure*

The representation of an SFD comprises a series of *protocol elements*, each of which is either a text unit or a specific logical descriptor:

```
Document ::= SEQUENCE OF ProtocolElement

ProtocolElement ::= Choice {
    textUnit [3] IMPLICIT TextUnit,
    specificLogicalDescriptor [5] IMPLICIT LogicalDescriptor}
```

A *text unit* represents a content portion. A *specific logical descriptor* represent a logical object.

The order of protocol elements within the Sequence is significant. The first element is the specific logical descriptor for the document. The specific logical descriptors for the document's paragraphs appear in the order in which the corresponding paragraphs appear in the document. Immediately following each paragraph's descriptor are the text units for the paragraph's content portions. The text units appear in the order in which the corresponding content portions appear in the paragraph.

*Note* – The structure of SFD protocol elements follows closely that of the protocol elements of the Document Interchange Protocol for the Telematic Services, which is specified in Recommendation T.73. In particular, the structure of text units conforms to that Recommendation, and the structure of specific logical descriptors parallels that of the specific layout descriptors defined by Recommendation T.73. Logical descriptors implement some of the logical document structure alluded to in Recommendation T.73.

5.3.2 *Text units*

A text unit specifies certain attributes of the content portion that the text unit represents, as well as the content portion itself. At present no attributes are defined, and the only type of content portion defined is character text (see § 5.2.2.1):

```
TextUnit :: = SEQUENCE {
    contentPortionAttributes ContentPortionAttributes,
    textInformation TextInformation}

ContentPortionAttributes :: = SET {– – none at present– –}

TextInformation :: = CHOICE {T61String}
```

The initial graphic rendition for the character text is that of the paragraph of which the character text is a part. The graphic rendition can be altered within the T.61 String as prescribed by Recommendation T.61.

5.3.3 *Specific logical descriptors*

A specific logical descriptor has two parts, logical object type and body. The former indicates whether the descriptor represents the document or a paragraph. The latter optionally specifies variable attributes of the object and/or, if the object is the document, default values for variable attributes of subordinate objects (that is, paragraphs):

```
LogicalDescriptor       ::= SEQUENCE {LogicalObjectType, LogicalDescriptorBody}

LogicalObjectType       ::= INTEGER {document(0), paragraph(1)}

LogicalDescriptorBody   ::= SET {
                              -- variable attributes (if object is document)
                                 pageHeading [3] IMPLICIT T61String OPTIONAL,
                              -- variable attributes (if object is paragraph)
                                 layoutDirectives [4] IMPLICIT
                                     LayoutDirectives OPTIONAL,
                                 presentationDirectives [5] IMPLICIT
                                     PresentationDirectives OPTIONAL,
                              -- default variable attributes for subordinate objects (if any)
                                 defaultValueLists [6] IMPLICIT SEQUENCE {DefaultValueList}
                                 OPTIONAL}
```

As indicated in § 5.2.3.1, the document has a single variable attribute: page heading. As indicated in § 5.2.3.2, a paragraph has four variable attributes: left indentation, bottom blank lines, alignment, and graphic rendition. These attributes are classified as *layout directives* or *presentation directives* as indicated below. They are called "directives" because they direct the rendering, or output, of the logical object they describe:

```
LayoutDirectives        ::= SET {
                                 leftIndentation [0] Offset OPTIONAL,
                                 bottomBlankLines [3] Offset OPTIONAL}

Offset                  ::= CHOICE { [1] IMPLICIT INTEGER}

PresentationDirectives  ::= SET {
                                 alignment [0] IMPLICIT Alignment OPTIONAL,
                                 graphicRendition [1] IMPLICIT GraphicRendition OPTIONAL}

Alignment               ::= INTEGER {
                                 leftAligned(0), centered(2), justified(3)}

GraphicRendition        ::= SEQUENCE OF GraphicRenditionAspect

GraphicRenditionAspect  ::= INTEGER -- an SGR parameter value; see Recommendation T.61
```

If the present object is the document, default values for variable attributes of paragraphs may be specified as follows:

```
DefaultValueList        ::= CHOICE {
                                 paragraphAttributes [1] IMPLICIT ParagraphAttributes}

ParagraphAttributes     ::= SET {
                                 layoutDirectives < Attribute OPTIONAL,
                                 presentationDirectives < Attribute OPTIONAL}

Attribute               ::= CHOICE {
                                 layoutDirectives [0] IMPLICIT LayoutDirectives,
                                 presentationDirectives [1] IMPLICIT PresentationDirectives}
```

If a particular attribute of a logical object is specified neither explicitly (that is, in the object's specific logical descriptor) nor implicitly (that is, as a default value in the subordinate object's specific logical descriptor), a fixed value is used as follows:

| | |
|---|---|
| Heading: | "" (that is, none) |
| Left indentation: | 0 |
| Bottom blank lines: | 1 |
| Alignment: | left aligned |
| Graphic rendition: | normal |

## 5.4 *Examples of use of SFD Document Interchange Format*

The Document Interchange Format for Simple Formattable Documents, specified in the preceding section, is illustrated in the present section. A simple, four-paragraph document provides the basis for the example.

The document in question is shown in Figure 5/X.420 as it might be rendered by the originator's UA. The same document is shown again in Figure 6/X.420, this time as it might be rendered by a recipient's UA equipped with a wider output medium.

Two of the possibly many alternative representations of this document are presented in the following two sections. One contains significant detail of the document's structure, encoded using the capabilities of the SFD-DIF. The other is a simpler representation which embeds characteristics of the layout with the document text.

### 5.4.1 *Structured encoding*

If the originator's UA provides him with a sophisticated text editor – one for example, that allows him to explicity indicate paragraph boundaries, indentation and alignment – it might produce on his behalf the SFD-DIF instance shown below. In this highly structured encoding, each paragraph is represented by a separate logical descriptor, and all of the document's "significant" characteristics are captured in the encoding and thus will be evident to the recipient when the document is rendered by his UA:

```
{
    specificLogicalDescriptor {document, {
        defaultValueLists {paragraphAttributes {alignment justified}}}},
    specificLogicalDescriptor {paragraph, {}},
    textUnit {{},
        "The establishment in various countries of telematic services and computer-based store-and-
        forward message services in association with public data networks creates a need to produce
        standards to facilitate international message exchange between subscribers to such services."},
    specificLogicalDescriptor {paragraph, {}},
    textUnit {{},
        "The CCITT, considering:"},
    specificLogicalDescriptor {paragraph, {
        presentationDirectives {alignment leftAligned},
        layoutDirectives {leftIndentation 5}}},
    textUnit {{},
        "(a) the need for interpersonal messaging and message transfer services;"},
    specificLogicalDescriptor {paragraph, {
        presentationDirectives {alignment leftAligned},
        layoutDirectives {leftIndentation 5}}},
    textUnit {{},
        "(b) the need to transfer messages of different types having a large variety of formats;"}}
```

*Note 1* – The values of the T.61 Strings that appear above contain no carriage return/line feed sequences. Those that seem to be multi-line are so rendered only for the purposes of illustration.

*Note 2* – The (human-readable) standard notation for data values is used above to described a particular instance of the Document Interchange Format. It is the corresponding (machine-readable) standard representation that is transferred between open systems.

### 5.4.2 *Unstructured encoding*

The originator's UA might, for some reason, elect to produce a largely unstructured encoding of the document. In particular, it might model the entire document as a single paragraph, producing the SFD-DIF instance shown below. To ensure that the document's "significant" characteristics are apparent to the recipient

when the document is rendered by his UA, paragraph boundaries and spacing must be indicated by carriage return/line feed sequences, and paragraph indentation by spaces. Items (a) and (b) in the document cannot be shown flush left (that is, unjustified) as intended, since the entire document must be specified either flush left or justified. Thus this particular "significant" document characteristic is lost:

```
{
    specificLogicalDescriptor {document, {
        defaultValueLists {paragraphAttributes {alignment justified}}}},
    specificLogicalDescriptor {paragraph, {}},
    textUnit {{},
        "The establishment in various countries of telematic services and computer-based store-and-
        forward message services in association with public data networks creates a need to produce
        standards to facilitate international exchange between subscribers to such services.↓↓The CCITT,
        considering;↓↓°°°° (a) the need for interpersonal messaging and message transfer
        services;↓↓°°°° (b) the need to transfer messages of different types having a large variety of
        formats;"}
```

*Note 1* – The value of the T.61 String that appears above contains a carriage return/line feed sequence only where the symbol "↓" appears. At other points where it appears to be multi-line, the T.61 String is so rendered only for the purposes of illustration.

*Note 2* – Throughout the value of the T.61 String that appears above, spaces of which the reader is to take special note are rendered by the symbol "°".

*Note 3* – The (human-readable) standard notation for data values is used above to describe a particular instance of the Document Interchange Format. It is the corresponding (machine-readable) standard representation that is transferred between open systems.

## ANNEX A

(to Recommendation X.420)

### Abbreviations

The following abbreviations are used in this Recommendation.

| | |
|---|---|
| CC | Courtesy copy |
| CR | Carriage return |
| DIF | Document interchange format |
| G3Fax | Group 3 Facsimile |
| IM-UAPDU | IP-message user agent protocol data unit |
| IP | Interpersonal |
| IPM | Interpersonal messaging |
| LF | Line feed |
| MHS | Message handling system |
| MT | Message transfer |
| MTL | Message transfer layer |
| NDN | Non-delivery notification |
| OSI | Open Systems Interconnection |
| P2 | Interpersonal messaging protocol |
| SFD | Simple formattable document |
| SR-UAPDU | Status report user agent protocol data unit |
| TIF.0 | Text interchange format 0 |
| TIF.1 | Text interchange format 1 |
| TLX | Telex |
| TTX | Teletex |
| UA | User agent |
| UAE | User agent entity |
| UAL | User agent layer |
| UAPDU | User agent protocol data unit |
| UTC | Coordinated universal time |

APPENDIX I

(to Recommendation X.420)

**List of terms**

The following terms are contained in this Recommendation. References to definitions, explicit or implicit, are included where appropriate and available.

## A

**Alternate-recipient-allowed**
cont. in X.420; impl. def. in X.420

**Alternate Recipient Allowed**
cont. in X.420; impl. def. in X.400

**Alternate Recipient Assignment**
cont. in X.420; impl. def. in X.400

**Application Layer**
cont. in X.420; impl. def. in X.200

**Attributes**
cont. in X.420; impl. def. in X.420

**Authorizing Users Indication**
cont. in X.420; impl. def. in X.400

**Auto-forwarded**
cont. in X.420; impl. def. in X.420

**Auto-forwarded Indication**
cont. in X.420; impl. def. in X.400

## B

**Basic Interpersonal Messaging Service**
cont. in X.420; impl. def. in X.400

**Blind Copy Recipient Indication**
cont. in X.420; impl. def. in X.400

**Body**
cont. in X.420; impl. def. in X.400

**Body Part**
cont. in X.420; impl. def. in X.400

**Body Part Encryption Indication**
cont. in X.420; impl. def. in X.400

**Body Part Type**
cont. in X.420; impl. def. in X.420

**Bottom Blank Lines**
cont. in X.420; impl. def. in X.420

## C

**Character Text**
cont. in X.420; impl. def. in X.420

**Character Text Body**
cont. in X.420; impl. def. in X.420

**Common Data Type**
cont. in X.420; impl. def. in X.420

**Confirmation**
cont. in X.420

**Content**
cont. in X.420; impl. def. in X.400

**Content Portion**
cont. in X.420; impl. def. in X.420

**Content-return**
cont. in X.420; impl. def. in X.420

**Conversion**
cont. in X.420; impl. def. in X.408

**Conversion-prohibited**
cont. in X.420; impl. def. in X.420

**Conversion Prohibition**
cont. in X.420; impl. def. in X.400

**Cross-referencing Indication**
cont. in X.420; impl. def. in X.400

## D

**Data Element**
cont. in X.420; impl. def. in X.408

**Deferred-delivery-time**
cont. in X.420; impl. def. in X.420

**Deferred Delivery**
cont. in X.420; impl. def. in X.400

**Deferred Delivery Cancellation**
cont. in X.420; impl. def. in X.400

**Deliver**
cont. in X.420; impl. def. in X.420

**Delivery-notice**
cont. in X.420; impl. def. in X.420

**Delivery Notification**
cont. in X.420; impl. def. in X.400

**Delivery-time**
cont. in X.420; impl. def. in X.420

**Disclosure-recipients**
cont. in X.420; impl. def. in X.420

**Disclosure of Other Recipients**
cont. in X.420; impl. def. in X.400

**Document**
cont. in X.420

## E

**Encoded-information**
cont. in X.420; impl. def. in X.420

**Envelope**
cont. in X.420; impl. def. in X.400

## F



## G



## H



## I



## L



## M



## N

## O



## P



## R



## S

**Subject Indication**
cont. in X.420; impl. def. in X.400

**Submission and Delivery**
cont. in X.420; impl. def. in X.420

**Submission-time**
cont. in X.420; impl. def. in X.420

**Submit**
cont. in X.420

**Supplementary-information**
cont. in X.420; impl. def. in X.420

## T

**Teletex**
cont. in X.420; impl. def. in X.200

**Telex**
cont. in X.420

**Text Unit**
cont. in X.420; impl. def. in X.420

## U

**UA-content-ID**
cont. in X.420; impl. def. in X.420

**Unspecified Attrribute**
cont. in X.420; impl. def. in X.420

**Unstructured Encoding**
cont. in X.420; impl. def. in X.420

**User Agent**
cont. in X.420; impl. def. in X.400

**User Agent Entity**
cont. in X.420; impl. def. in X.400

**User Agent Layer**
cont. in X.420; impl. def. in X.420

**User Agent Protocol Data Unit**
cont. in X.420; impl. def. in X.420

## V

**Variable Attribute**
cont. in X.420; impl. def. in X.420

*Note* – Terms with several words combined with a hyphen are generally names of service primitive parameters.

**Recommendation X.430**

### MESSAGE HANDLING SYSTEMS: ACCESS PROTOCOL FOR TELETEX TERMINALS

*(Malaga-Torremolinos, 1984)*

The establishment in various countries of telematic services and computer-based store-and-forward message service in association with public data networks creates a need to produce standards to facilitate international message exchange between subscribers to such services.

The CCITT,

*considering*

(a) the need for interpersonal messaging and message transfer services;

(b) the need to transfer messages of different types having a large variety of formats;

(c) that within the X-series of Recommendations services and optional user facilities for public data networks are defined;

(d) that the F-series of Recommendations defines telematic services and that the T-series of Recommendations defines terminal equipment and control procedures for telematic services;

(e) that the V-series of Recommendations provides the means for data communication over the telephone network;

(f) that a set of Recommendations describes various aspects of message handling: X.400, X.401, X.408, X.409, X.410, X.411 and X.420,

*unanimously declares*

that this Recommendation describes the access protocol to be used by Teletex terminals when making additional use of the message handling system.

CONTENTS



## 1 Purpose and scope

This Recommendation is one of a series of Recommendations and describes the access protocol for Teletex Terminals to the Message Handling System (MHS). This Recommendation specifies the protocol to be used by Teletex (TTX) terminals when accessing the MHS for the purpose of providing the Interpersonal Messaging (IPM) Service to their users. The IPM service elements are made available to users of the Teletex Service.

Other Recommendations in the series contain additional information essential to a thorough understanding of the MHS. Recommendation X.401 outlines the basic service elements and optional user facilities of the IPM and Message Transfer (MT) Services. Recommendation X.400 describes the MHS model and defines the IPM and MT service elements. Recommendation X.411 specifies the protocol aspects of the MT Service. Recommendation X.420 specifies the protocol aspects of the IPM Service. Recommendation X.409 defines the notation and representational technique used to specify and encode MHS protocols. Recommendation X.408 specifies the algorithms the MHS uses when converting between different types of encoded information. Recommendation X.410 describes general techniques used for MHS protocols and the way in which standard Open Systems Interconnection (OSI) protocols are used to support MHS applications.

§ 2 of this Recommendation defines the Teletex Access Model, the actions of the protocol used by the Teletex terminal, and the operation of the *Teletex Access Unit (TTXAU)*. § 3 describes the various action elements used. § 4 provides details on the control procedures defined. § 5 defines the error recovery procedures. § 6 describes the encoding and formatting of control information. § 7 lists the appropriate references.

## 2 Teletex access model

### 2.1 *Field of application*

The TTXAU is an additional functionality provided within the Message Handling System to accommodate a Teletex terminal. The purpose of a Teletex Access Unit (TTXAU) is to aid the user of a Teletex terminal in gaining access to the features of the IPM Service. The TTXAU, which is associated with a Message Transfer Agent (MTA), provides the Teletex terminal with access to the Message Transfer System (MTS).

An instance of a TTXAU logically supports a single Teletex terminal. As such it does not perform any message switching or routing functions between Teletex terminals. Communication between the Teletex terminal and the TTXAU is governed by the *Teletex Access Protocol (P5)*, which uses human-readable encoding.

A Teletex terminal shall be addressable by at least a Form 2 address (see X.400, § 3.3.3) and may also be addressed by one or more variants of Form 1 address.

The ability to negotiate the use of Teletex standardized options in documents submitted to the MTS is for further study.

The TTXAU may also provide a *Document Storage (DS)* facility to accept delivery of messages from the MTS for Teletex users. Document Storage is basically defined as a Teletex terminal storage extension facility located in the TTXAU allowing reservation of a specific amount of storage for an individual User-ID. Users of Teletex terminals may also be registered as users of DS. Messages from the MTS for distribution to these registered DS users will be stored in an individual DS (*output message*) until either the Teletex user specifically requests that they be output to the Teletex terminal, or the TTXAU succeeds in outputting the message to the Teletex terminal.

*Note 1* – for the present, one User-ID is mapped onto one Terminal Identifier in the Teletex Service.

*Note 2* – The Hold for Delivery service element defined in Recommendation X.400 (§ 4) is distinguished from the Document Storage facility defined in this Recommendation. The Hold for Delivery service element provides temporary storage (message "buffering") until delivery can occur. Only after a message has been successfully transferred to the recipient's UA is delivery complete and a delivery notification returned. The Document storage facility is a user subscribed extension of the Teletex terminal storage and may be used to store messages for an extended period of time. Unlike the Hold for Delivery service element, delivery is considered complete when a message is successfully transferred to Document Storage at which time delivery notification is returned (if requested).

Figure 1/X.430 illustrates the TTXAU/DS as it relates to both the Teletex and IPM Services.

The Teletex terminal and TTXAU may communicate with one another by means of the control procedures of the Teletex Service. The provisions of the Teletex Service are as defined in Recommendations F.200, T.60, T.61, T.62, T.70, T.71 and T.90. Either the Teletex terminal or the TTXAU may initiate the call. The Teletex Access Protocol comes into operation after a call is established. P5 itself is based upon Recommendation T.62, with the provisions of Recommendations F.200 and T.90 taken into account.

### 2.2 *Teletex Access Protocol actions*

The Teletex Access Protocol comprises several *actions*, each of which is carried out independently. One or more actions can occur after call establishment.

#### 2.2.1 *Message Sending action*

The *Message Sending* action is initiated by the Teletex terminal. In this action, the Teletex terminal sends to the TTXAU the following:

1) Control information related to IPM service elements, which includes MT service elements describing the submitted message.
2) One or more Teletex documents, which constitute the body of the content of a message being submitted.

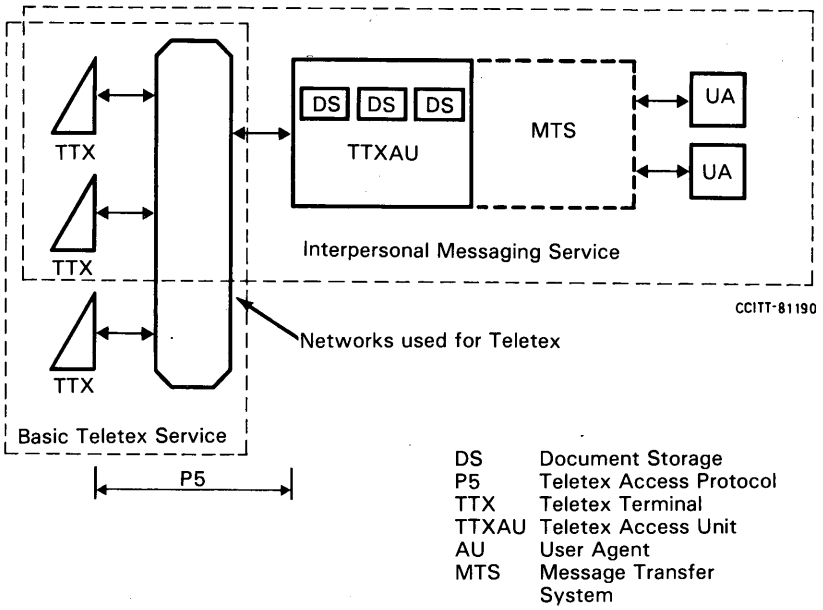



FIGURE 1/X.430
**Teletex Access Protocol environment**

2.2.2 *Probe action*

The *Probe* action is initiated by the Teletex terminal. The purpose of this action is to determine whether a message with particular submission information could be delivered. The Teletex terminal sends to the TTXAU control information related to MT service elements that describe the message being probed.

2.2.3 *Message Delivery action*

The *Message Delivery* action is initiated by the TTXAU. In this action, the TTXAU sends to the Teletex terminal the following:

1) Control information related to IPM service elements which includes MT service elements describing the delivered message. An indication that receipt notification is required may be included in the control information.
2) One or more Teletex documents, which constitute the body of the content of a message being delivered.

2.2.4 *Notification action*

The *Notification* action is initiated by the TTXAU. In this action, the TTXAU sends a Delivery or Receipt Status Notification to the Teletex terminal.

The *Delivery Status Notification* indicates one of the following:

1) That a message has not been accepted for submission.
2) That a message has been successfully delivered.
3) That the attempt to deliver a message has failed. Besides the non-delivery reasons common to all messages, the reasons for non-delivery in this context include Teletex-specific reasons reflecting local problems between the TTXAU and the Teletex terminal.
4) That a message has been stored in the DS of the destination User-ID.

The *Receipt Status Notification* indicates one of the following:

1) That the message has been stored in the Teletex terminal's storage and that the notification has been automatically generated by the TTXAU, after successful output of the message to the Teletex terminal, for example, delivery or output.
2) That the message has been explicitly acknowledged by the recipient by way of the Receipt Acknowledgment action.

### 2.2.5 *Document Storage Management action*

The *Document Storage Management* action may be initiated by the Teletex terminal to determine whether output messages are available for output from the DS. The TTXAU, either in response to the above request or based upon an automatic mechanism, will provide a DS Report to the Teletex terminal containing information on the messages being held in the DS for the User-ID. The Teletex terminal may request the output of one or more of the messages identified as residing in the DS.

### 2.2.6 *Output action*

The *Output* action may be initiated by the TTXAU to send to the Teletex Terminal the output message(s) held in the DS in response to the Output Request action element if the retrieval mode is registered. The TTXAU tries to send output messages from the DS automatically to the Teletex terminal if auto-output mode has been registered and the conditions for automatic output are met.

### 2.2.7 *Registration action*

The *Registration* action is initiated by the Teletex terminal. In this action, the Teletex terminal sends to the TTXAU control information concerning the definition or modification of various aspects of the permanent relationship between the Teletex terminal and the TTXAU (for example, supported types of encoded information, the DS retrieval mode, or the DS auto-output mode).

### 2.2.8 *Receipt Acknowledgment action*

The *Receipt Acknowledgment* action may be initiated by the Teletex terminal user when a received message contains a request for receipt notification and the message has been presented to the user.

### 2.2.9 *Exception Reporting action*

The *Exception Reporting* action is initiated by the TTXAU to indicate the receipt of an invalid action element. The TTXAU may also initiate this Exception Reporting action to indicate that a partially received action element has been discarded.

## 2.3 *Operation of TTXAU*

This section describes the operations that a TTXAU has to carry out, in reaction either to an action initiated by the Teletex terminal or to an event arriving from the MTA.

An instance of the TTXAU is logically considered to be a single entity, acting with the Teletex terminal on one hand and with an MTA on the other hand. However, conceptually the TTXAU and the MTA together have to be considered a single system, that is the interface between the two is not formally defined within the context of this Recommendation.

The Teletex Access Unit represents the Teletex terminal to the MHS. The structure of the TTXAU in the layered representation is shown in Figure 2/X.430.

Some operations of the TTXAU are determined by the existence of a DS facility in the TTXAU associated with the Teletex terminal.

§ 2.3.1 describes operations which are independent of the selection of the DS facility. § 2.3.2 describes operations where DS has not been selected. § 2.3.3 describes operations where DS has been selected.

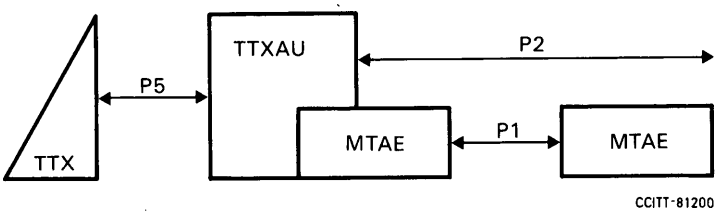



FIGURE 2/X.430
**TTXAU protocol overview**

2.3.1 *General operations*

2.3.1.1 *Registration*

Upon receipt of a Registration action, the TTXAU updates the parameters of the permanent relationship between itself and the Teletex terminal.

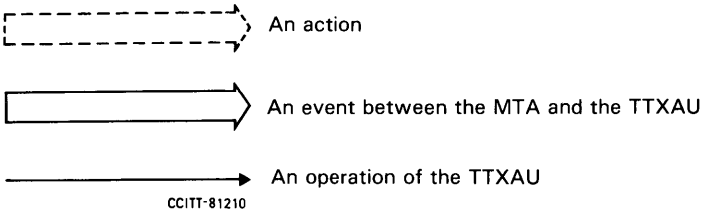
2.3.1.2 *Exception reporting*

Upon receipt of any action initiated by the Teletex terminal, syntactical errors are reported by the TTXAU to the Teletex terminal by means of an Exception Reporting action.

2.3.2 *Operations without DS facility*

2.3.2.1 *Message Sending and Probe*

The operations associated with Message Sending are described in Figure 4/X.430. The symbols used in this and subsequent action-related figures are explained in Figure 3/X.430.




Conditional actions or operations are shown with dashed lines.
Numbers enclosed in parentheses are references to notes.

FIGURE 3/X.430
**Legend for action figures**

2.3.2.2 *Message Delivery*

The operations associated with Message Delivery are described in Figure 5/X.430.

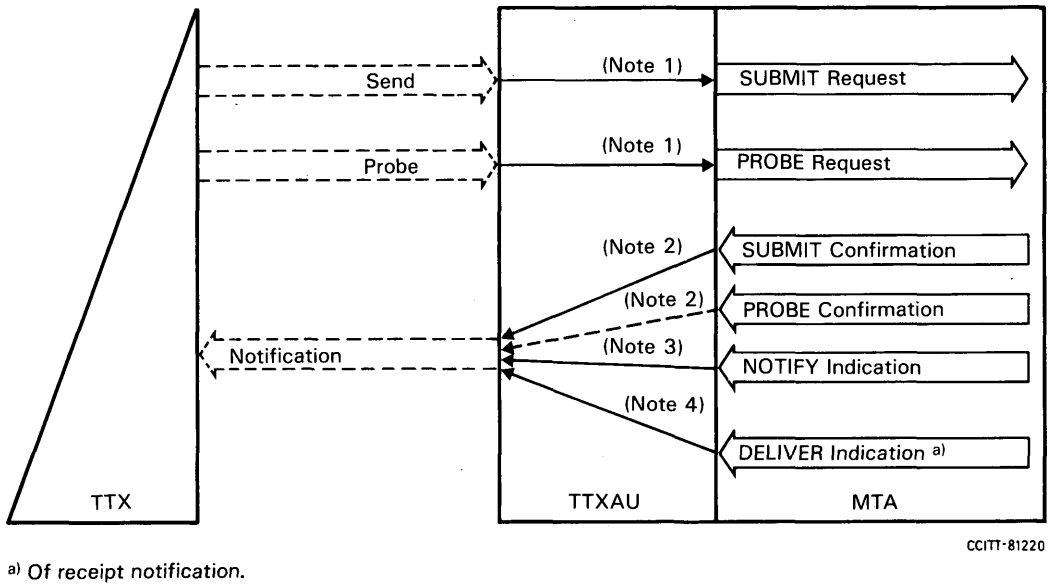
2.3.3 *Operations with DS Facility*

2.3.3.1 *Message Sending and Probe*

The operations associated with Message Sending are described in Figure 6/X.430.

2.3.3.2 *Message Output*

The operations associated with Message Output are described in Figure 7/X.430.




a) Of receipt notification.

*Note 1* — Upon receipt of a Send or Probe action from the teletex terminal, the TTXAU issues a SUBMIT.Request or PROBE.Request to the MTA. A SUBMIT.Request may result in a SUBMIT.Confirmation, NOTIFY.Indication, and/or DELIVER.Indication being returned by the MTA. A PROBE.Request may result in a PROBE.Confirmation and/or NOTIFY.Indication being returned by the MTA.

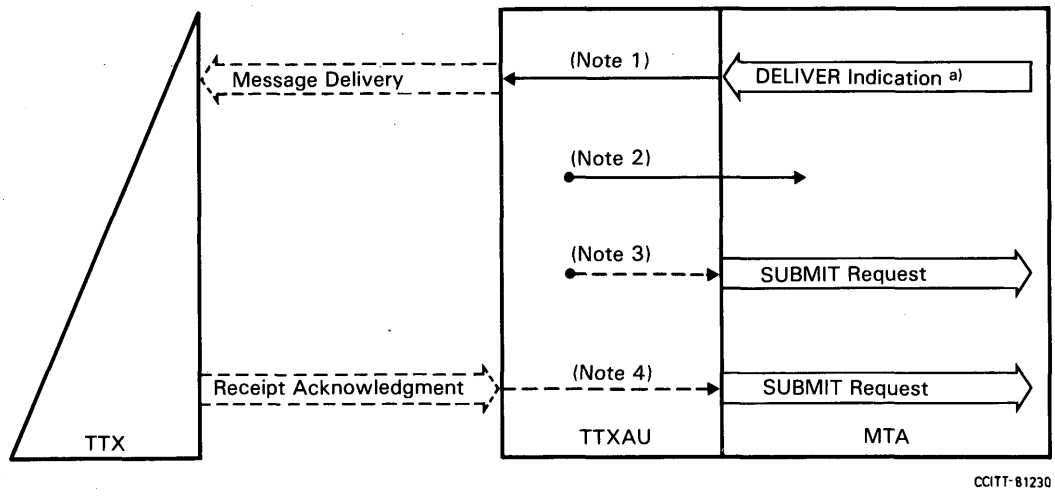
*Note 2* — A SUBMIT.Confirmation indicating failure of the SUBMIT.Request (see Section 2.2.7 of Recommendation X.411) causes the TTXAU to initiate a Notification action to the Teletex terminal.

*Note 3* — A NOTIFY.Indication indicating either delivery or non-delivery of the message (see Section 2.2.10 of Recommendation X.411) or result of a Probe, causes the TTXAU to initiate a Notification action to the Teletex terminal.

*Note 4* — A DELIVER.Indication containing a receipt report (see Section 3.3 of Recommendation X.420) causes the TTXAU to initiate a Notification action to the Teletex terminal.

FIGURE 4/X.430

**Actions related to Message Sending and Probe**




a) Of a user message.

*Note 1* — Upon receipt of a DELIVER.Indication relating to an IP-message, the TTXAU initiates the Message Delivery action.

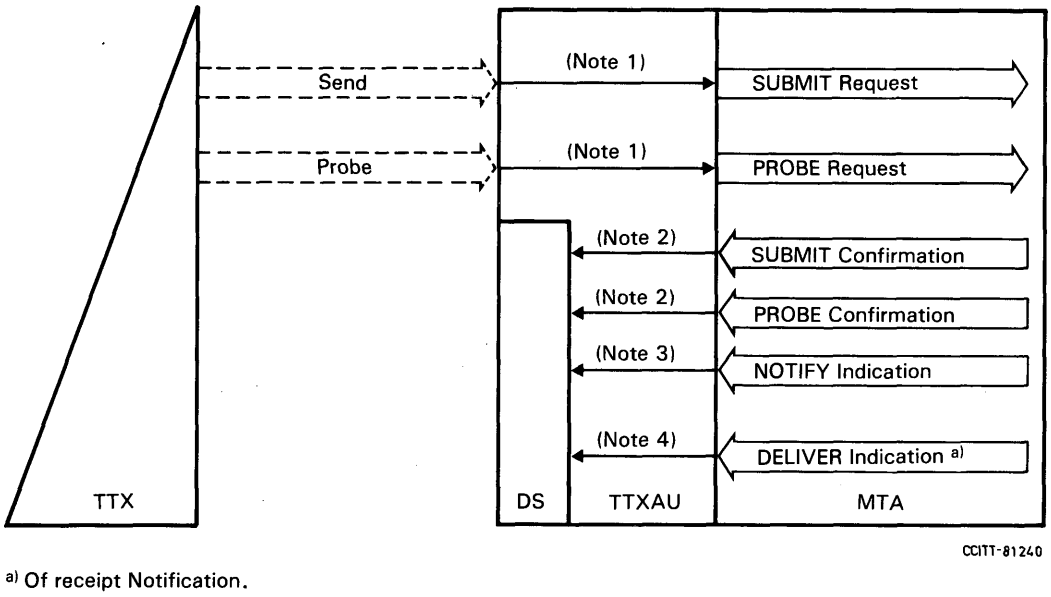
*Note 2* — The TTXAU informs the MTA of the success or failure of the Message Delivery action, prompting the MTA to return either a delivery report, if requested, or a non-delivery report. The circumstances that cause a message to be declared undeliverable are specified in Recommendation X.411.

*Note 3* — If by a subscription arrangement between the Teletex terminal and the TTXAU, the TTXAU is conditioned to automatically return an auto-receipt status report upon successful delivery of a message, the TTXAU generates and submits reports to the MTA. The action of the TTXAU when submission or delivery of this report is unsuccessful is not defined.

*Note 4* — Upon receipt of a Receipt Acknowledgment action from the Teletex terminal, the TTXAU submits the explicit receipt status report to the MTA. The SUBMIT.Request may result in a SUBMIT.Confirmation and/or NOTIFY.Indication being returned by the MTA, in which case the TTXAU's action is described in § 2.3.2.1.

FIGURE 5/X.430

**Actions related to Message Delivery**




a) Of receipt Notification.

*Note 1* — Upon receipt of a Send or Probe action from the Teletex terminal, the TTXAU issues a SUBMIT or PROBE.Request to the MTA. A SUBMIT.Request may result in a SUBMIT.Confirmation, NOTIFY.Indication, and/or DELIVER.Indication being returned by the MTA. A PROBE.Request may result in a PROBE.Confirmation and/or NOTIFY.Indication being returned by the MTA.

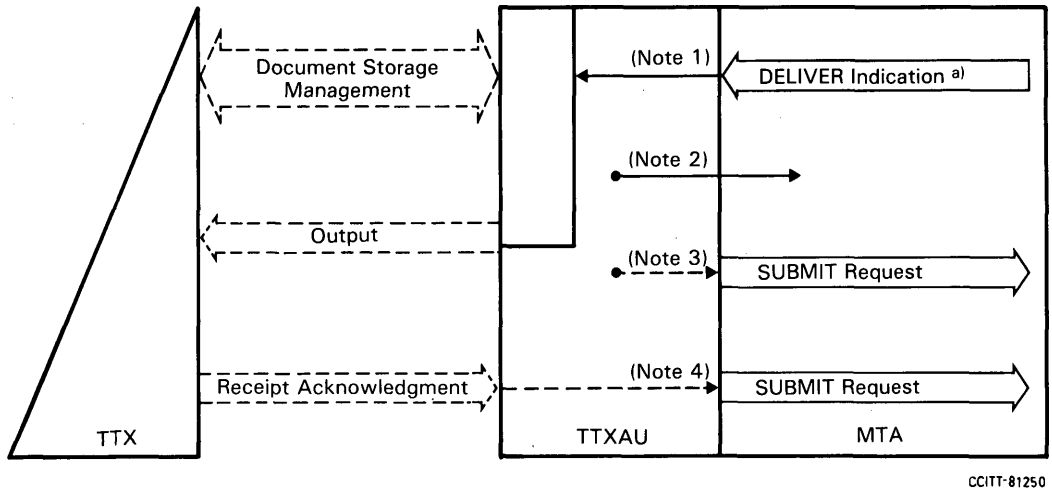
*Note 2* — A SUBMIT.Confirmation indicating failure of the SUBMIT.Request (see Section 2.2.7 of Recommendation X.411) causes the TTXAU to direct a Notification message to the DS.

*Note 3* — A NOTIFY.Indication indicating either delivery or non-delivery of the message or result of a Probe causes the TTXAU to direct a Notification message to the DS (see § 2.2.10 of Recommendation X.411).

*Note 4* — A DELIVER.Indication containing a receipt report causes the TTXAU to direct a Notification message to the DS (see § 3.3 of Recommendation X. 420).

FIGURE 6/X.430

**Actions related to Message Sending and Document Storage**




a) Of a user-message.

*Note 1* — Upon receipt of delivery indication from the MTA, the TTXAU will store the message in the DS. (The message may be an IP-message or a Notification.) According to the subscription arrangement between the Teletex terminal and the TTXAU, the TTXAU either automatically initiates the output action or performs Output under the control of the Document Storage Management action.

*Note 2* — Once the message is in the DS, the TTXAU wll indicate to the MTA that the message was delivered to the DS.

*Note 3* — If by a subscription arrangement between the Teletex terminal and the TTXAU, the TTXAU is conditioned to automatically return a receipt status report upon successful delivery of a message, the TTXAU generates and submits this report to the MTA. The action of the TTXAU when submission or delivery of this report is unsuccessful is not defined.

*Note 4* — Upon receipt of a Receipt Acknowledgment action from the Teletex terminal, the TTXAU submits the explicit receipt status report to the MTA. The SUBMIT.Request may result in a SUBMIT.Confirmation and/or NOTIFY.Indication being returned by the MTA, in which case the TTXAU's action is described in § 2.3.3.1.

FIGURE 7/X.430

**Actions related to Message Output and Document Storage**

## 3 Description of action elements

An action consists of one or more action elements. An *action element* is composed of either a single document containing control information, or a sequence of documents, the first of which contains control information.

This section specifies the control information of the action elements used in each action of the Teletex Access Protocol. These action elements are summarized in Table 1/X.430. The status of each action element is either *mandatory (M)* or *conditional (C)* from two perspectives, that of the Teletex terminal and that of the TTXAU.

The various components of an action element are defined by referring to the relevant MHS Recommendations when appropriate.

*Note* – In the following description, only the first instance of an action element component is referenced.

### 3.1 *Message Sending action*

One action element is used in the Message Sending action. The Send action element is used by the Teletex terminal to request submission of a message.

A Send action element that makes use of IPM parameters Primary and Copy recipients in conjunction with Blind Copy recipients should result in the TTXAU creating separate submit requests to maintain the integrity of the Blind Copy recipients not being revealed to the Primary and Copy recipients.

TABLE 1/X.430

**Teletex Access Protocol action elements**

| Action | Action elements | TTX status | TTXAU status | Direction of transfer TTX ⟵ ⟶ TTXAU |
|---|---|---|---|---|
| Message sending | Send | M | M | → |
| Probe | Probe | C | M | → |
| Message delivery [a] | Message delivery | C | M | ← |
| Notification | Delivery Status Notification<br>Receipt Status Notification | M<br>C | M<br>M | ←<br>← |
| Document Storage Management | DS Query<br>DS Report<br>Output Request | C<br>C<br>C | C<br>C<br>C | →<br>←<br>→ |
| Output [a] | Output Message | C | M | ← |
| Registration | Registration | C | M | → |
| Receipt Acknowledgment | Explicit Receipt Acknowledgment | C | M | → |
| Exception Reporting | Exception Reporting | M | M | ← |

[a] A message may arrive at a Teletex terminal as a result of either a Message Delivery or Output action. The Message Delivery action is applicable when delivery occurs directly to a Teletex terminal. The Output action is only applicable in the case of document storage.

### 3.1.1 *Send action element*

The components of the control information of the *Send* action element are as follows:

**Control information reference**

| Component | Type | Notes |
|---|---|---|
| Action Element Identifier | M | 1 |
| Number of Associated Documents | C | 2 |

*Note 1* – The Action Element Identifier identifies the Send Action element.

*Note 2* – The Number of Associated Documents indicates the number of associated normal documents, sent in sequence with this first one, as a single Action element. This parameter is mandatory when the TTX terminal cannot handle the control type of document (control document).

**Message transfer parameters**

| Component | Type | Reference |
|---|---|---|
| Originator O/R Name | M | [X.411, §§ 3.4.1-3.4.2] |
| UA Content ID | C | [X.411, § 3.4.1] |
| Deferred Delivery Time | C | [X.411, § 4.3.2] |
| Priority | C | [X.411, § 3.4.2] |
| Per Message Flag | C | [X.411, § 3.4.2] |
| Disclose Recipients | C | |
| Conversion Prohibited | C | |
| Alternate Recipient Allowed | C | |
| Content Return Request | C | |
| Recipient Info (repeated for each recipient) | M | [X.411, § 3.4.2] |
| Recipient O/R Name | M | |
| Per Recipient Flag | M | |
| – Users Report Request | M | |
| Explicit Conversion | C | |

**Interpersonal Messaging parameters**

| Component | Type | Reference |
|---|---|---|
| IP Message ID | M | [X.420, § 3.1.1] |
| Originator | C | [X.420, § 3.2.1] |
| Authorizing Users | C | [X.420, § 3.2.1] |
| Primary Recipients | C | [X.420, § 3.2.1] |
| Recipient (repeated for each primary recipient) | | |
| – O/R Descriptor | | |
| – Report Request | | |
| a) Return IP-Message<br>b) Non-receipt Notification<br>c) Receipt Notification | | |
| – Reply Request | | |
| Copy Recipients | C | [X.420, § 3.2] |
| Recipient. It is identical to the parameter Primary Recipients, above (repeated for each recipient) | | |
| Blind Copy Recipients | C | [X.420, § 3.2.1] |
| Recipient. It is identical to the parameter Primary Recipients, above (repeated for each recipient) | | |
| In Reply To | C | [X.420, § 3.2.1] |
| Obsoletes | C | [X.420, § 3.2.1] |
| Cross-references | C | [X.420, § 3.2.1] |
| Subject | C | [X.420, § 3.2.1] |
| Expiry Date | C | [X.420, § 3.2.1] |
| Reply By Time | C | [X.420, § 3.2.1] |
| Reply To Users | C | [X.420, § 3.2.1] |
| O/R Descriptor (repeated for each user) | | |
| Importance | C | [X.420, § 3.2.1 |
| Sensitivity | C | [X.420, § 3.2.1] |

3.2 *Probe action*

One action element is used in the Probe action. The Probe action element is used by the Teletex terminal to request the transfer of a probe.

3.2.1 *Probe action element*

The components of the control information of the *Probe* action element are as follows:

**Control information reference**

| Component | Type | Reference | Notes |
|---|---|---|---|
| Action Element Identifier | M | | 1 |
| Number of Associated Documents | C | [X.430, § 3.1.1] | |

*Note 1* - The Action Element Identifier identifies the Probe Action element.

**Probe Request parameters**

These components are identical to the Message Transfer parameters in the Send action element (see § 3.1.1), except for the following:

| Component | Type | Reference |
|---|---|---|
| Content Length | C | [X.411, § 3.4.4] |

*Note 2* – The following Send parameters do not occur in the Probe action element: Deferred Delivery Time, Priority, Content Return Request. There are no Interpersonal Messaging parameters in the action element.

3.3 *Message Delivery action*

One action element is used in the Message Delivery action. The Message Delivery action element is used by the TTXAU to deliver a message to the Teletex terminal.

3.3.1 *Message Delivery action element*

The components of the control information of the *Message Delivery* action element are as follows:

**Control information reference**

| Component | Type | Notes |
|---|---|---|
| Action Element Identifier | M | 1 |
| Number of Associated Documents | C | |

*Note 1* – The Action Element Identifier identifies the Message Delivery action element.

**Message Transfer parameters**

| Component | Type | Reference |
|---|---|---|
| Deliver Event ID | M | [X.411, § 4.3.3] |
| Delivery Time | M | [X.411, § 3.4.3] |
| Originator O/R Name | M | |
| Original Encoded Information Types | M | [X.411, § 3.4.4] |
| Priority | M | |
| Conversion Prohibited (Yes/No) | C | [X.411, § 3.4.2] |
| Other Recipients | C | [X.411, § 4.3.3] |
| O/R Names (repeated for each recipient) | | |
| This Recipient | M | [X.411, § 4.3.3] |
| O/R Name | | |
| Intended Recipient | C | [X.411, § 4.3.3] |
| O/R Name | | |
| Converted Encoded Information Types | C | [X.411, § 3.4] |
| Submission Time | M | [X.411, § 4.3.3] |

**Interpersonal Messaging heading**

| Component | Type | Reference |
|---|---|---|
| Auto Forwarded (Yes/No) | C | [X.420, § 3.2.1] |
| Body Part | C | [X.420, § 3.2.2] |
| Forwarded Message | | |
| – Delivery Time<br>– Delivery Information | | |

Interpersonal Messaging Parameters from the Send action (see § 3.1.1) are also included, if they are present in the message.

3.4 *Notification action*

Two action elements are used in the Notification action. The Delivery Status Notification action element is used by the TTXAU to report the status of a previously submitted message. The Receipt Status Notification action element is used by the TTXAU to report the status of a previously submitted message for which receipt notification was requested.

3.4.1 *Delivery Status Notification action element*

The components of the control information of the *Delivery Status Notification* action element are as follows:

**Control Information Reference**

| Component | Type | Notes |
|---|---|---|
| Action Element Identifier | M | 1 |
| Number of Associated Documents | C | |
| Correlation Information (of the corresponding Send action element) | M | 2 |
| Submitted TID<br>TTXAU TID<br>Date and Time<br>Additional session Reference Number<br>CD no | | |

*Note 1* – The Action Element Identifier identifies the Delivery Status Notification Action Element.

*Note 2* – The Correlation Information is composed of information extracted from the S.62 connection identification, upon which the Send Action Element has been sent (Table 2/X.430).

**Message Transfer parameters**
(repeated for each Recipient)

| Component | Type | Reference |
|---|---|---|
| Recipient O/R name | M | |
| Delivered Info [a] | C | [X.411, § 3.4.2] |
| Delivery Time | M | |
| Type of UA (Public/Private) | M | |
| Non-delivered Info | C | [X.411, § 3.4.3] |
| Reason | M | |
| Diagnostic Code | C | |
| Converted Encoded Information Types | C | |
| Intended Recipient O/R Name | C | |
| Supplementary Information | C | [X.411, § 3.4.3] |
| Delivered Document Storage | C | |
| UA Content ID (of the corresponding Send Action element) | C | |

[a] One and only one of these components must be present.

3.4.2 *Receipt Status Notification action element*

The components of the control information of the *Receipt Status Notification* action element are as follows:

**Control information reference**

| Component | Type | Notes |
|---|---|---|
| Action Element Identifier | M | 1 |
| Number of Associated Documents | C | |

*Note 1* – The Action Element Identifier identifies the Receipt Status Notification action element.

**Message Transfer Parameters**

Same as for the Message Delivery action element (see § 3.3.1).

**Interpersonal Messaging parameters**

| Component | Type | Reference | Notes |
|---|---|---|---|
| Report Type (Receipt/Non-receipt) | M | | 1 |
| IP Message Id (of received message) | M | | |
| Intended Recipient (only if not actual recipient) | C | | |
| Converted Encoded Information Types | C | | |
| Non-receipt Information | C | [X.420, § 3.3] | |
| Reason | M | | |
| Non-receipt Qualifier | C | | |
| Comments | C | | |
| Returned IP message | C | | |
| Receipt Information | C | [X.420, § 3.3] | |
| Time | M | | |
| Type of Receipt | C | | |
| This Recipient | M | | |

*Note 1* – The report type indicates whether it is a receipt or non-receipt notification.

3.5 *Document Storage Management action*

The Document Storage Management action consists of three action elements. The DS Query action element is used by the Teletex terminal to ask what output messages (if any) are stored in the DS and available for output. The DS Report action element is sent by the TTXAU in response to DS Query to indicate what output messages are in the DS. It may also be sent spontaneously from the TTXAU to provide an "alert" function. The times at which such a spontaneous DS Report will be sent are not defined in this Recommendation. The Output Request action element is used by the Teletex terminal to request the output of specific messages from the DS. (These messages are then output by means of the Output action.)

*Note* – Delivery, non-delivery and receipt notifications are held in the DS, and output to the Teletex terminal in the same way as IP-messages.

3.5.1 *DS Query action element*

The components of the control information of the *DS Query* action element are as follows:

**Control information reference**

| Component | Type | Notes |
|---|---|---|
| Action Element Identifier | M | 1 |
| Number of Associated Documents | C | |

*Note 1* – The Action Element Identifier identifies the DS Query Action Element.

3.5.2 *DS Report action element*

The components of the control information of the *DS Report* action element are as follows:

**Control information reference**

| Component | Type | Notes |
|---|---|---|
| Action Element Identifier | M | 1 |
| Number of Associated Documents | C | |

*Note 1* – The Action Element Identifier identifies the DS Report Action Element.

**Message Parameters**
(repeated for each reported message)

| Component | Type | Notes |
|---|---|---|
| Message Type | M | 1 |
| Priority (Urgent/Normal/Non-urgent) | M | |
| Message Length (in octets) | M | 2 |
| Originator O/R Name | C | |
| Retrieval Identifier | C | 3 |

*Note 1* – The Message Type indicates the type of the subject message.

*Note 2* – The Message Length indicates the length of the subject message.

*Note 3* – The Retrieval Identifier is a numeric identifier, which has local significance between the Teletex terminal and the DS, for retrieval purposes.

3.5.3 *Ouput Request action element*

The components of the control information of the *Output Request* action element are as follows:

**Control information reference**

| Component | Type | Notes |
|---|---|---|
| Action Element Identifier | M | 1 |
| Number of Associated Documents | C | |

*Note 1* – The Action Element Identifier identifies the Output Request Action Element.

**Message parameters**

| Component | Type | Notes |
|---|---|---|
| Message Selector | M | 1 |

*Note 1* – The Message Selector is a list of Retrieval Identifier of the messages that have to be retrieved.

3.6 *Output action*

The Output action applies only when the user has subscribed to the DS, and consists of one action element. The Output Message action element is used by the TTXAU to send a message from the DS to the recipient Teletex terminal. Output Message is triggered by one of the following three events:

1) Some rule (not defined in this Recommendation) which causes the TTXAU to establish a connection to the Teletex terminal and to send Output Message at a specific time (for example, the Teletex terminal has registered its times of availability with the TTXAU).
2) The Teletex terminal establishes a connection to the TTXAU and initiates a CSCC, which is taken as an implicit request for output by the TTXAU.
3) Receipt of an Output Request action element (see § 3.5.3).

3.6.1 *Output Message action element*

The components of the control information of the *Output Message* action element are as follows:

**Control information reference**

| Component | Type | Notes |
|---|---|---|
| Action Element Identifier | M | 1 |
| Numbered of Associated Documents | C | |
| Retrieval Identifier (see § 3.5.2) | C | 2 |

*Note 1* – The Action Element Identifier identifies the Output Message Action Element.

*Note 2* – This parameter is only present if the Output Message is in response to an Output Request Action Element.

**Message parameters**

| Component | Type |
|---|---|
| Message Type | M |

**Message Transfer and Interpersonal Messaging parameters**

As per §§ 3.3.1, 3.4.1 or 3.4.2 as appropriate.

3.7 *Registration action*

One action element is used in the Registration action. The Registration action element is used by the Teletex terminal to define or modify its registration profile.

3.7.1 *Registration action element*

The components of the control information of the *Registration* action element are as follows:

**Control information reference**

| Component | Type | Notes |
|---|---|---|
| Action Element Identifier | M | 1 |
| Number of Associated Documents | C | |

*Note 1* – The Action Element Identifier identifies the Registration Action Element.

**Registration parameters**

| Component | Type | Notes |
|---|---|---|
| Document Storage Mode | C | 1 |
| Teletex Terminal Profile Registration | C | 5 |
| Error Recovery Mode | C | 2 |
| Message Length | C | 3 |
| Receipt Notification Mode | C | 4 |

*Note 1* – This indicates the following output mode.

a) **Retrieval:** In this mode the DS holds messages for the Teletex terminal until they are retrieved by the Teletex terminal.

b) **Auto-output:** In this mode the DS tries to output messages under user subscribed conditions after they are delivered to the DS.

*Note 2* – This allows the selection of one of the error recovery modes described in § 5.

*Note 3* – This indicates the maximum size in octets of the message which the Teletex terminal is willing to accept.

*Note 4* – This indicates whether or not the TTXAU shall perform auto-receipt notification.

*Note 5* – This component is only used for registration of the profile of a Teletex terminal which subscribes to DS. This is intended to allow the DS to emulate the called Teletex terminal.

3.8 *Receipt Acknowledgment action*

One action element is used in the Receipt Acknowledgment action. The Explicit Receipt Acknowledgment action element is generated by the recipient Teletex user and is used to request the TTXAU to send an explicit receipt notification to the originator.

3.8.1 *Explicit Receipt Acknowledgment action element*

The components of the control information of the *Explicit Receipt Acknowledgment* action element are as follows:

**Control information reference**

| Component | Type | Notes |
|---|---|---|
| Action Element Identifier | M | 1 |
| Number of Associated Documents | C | |

*Note 1* – The Action Element Identifier identifies the Explicit Receipt Acknowledgment Action Element.

**Interpersonal Messaging parameters**

| Component | Type |
|---|---|
| IP Message ID | M |
| Intended Recipient | C |
| Converted Encoded Information Type | C |

*Note* – The TTXAU extracts the receipt time from the CSS of the session in which this action element was transferred. This time may differ from the time of actual receipt of the IP-message.

3.9 *Exception Reporting action*

One action element is used in the Exception Reporting action. The Exception Reporting action element is used by the TTXAU to inform the Teletex user that an action element is invalid or that a partially received message was discarded.

3.9.1 *Exception Reporting action element*

The components of the control information of the *Exception Reporting* action element are as follows:

**Control information reference**

| Component | Type | Notes |
|---|---|---|
| Action Element Identifier | M | 1 |
| Number of Associated Documents | C | |
| Correlation Information (of the concerned action element) (defined in § 3.4.1.1) | M | |
| Reason Code | M | 2 |
| Invalid action element<br>Partially received message discarded | | |

*Note 1* – The Action Element Identifier identifies the Exception Reporting Action Element.

*Note 2* – The Reason Code indicates the reason for reporting a protocol error or that a partially received message has been discarded.

## 4 Control procedures

Within a single call, more than one action may be initiated and more than one action element transferred.

### 4.1 *Session control procedures*

The session control procedures shall be in accordance with Recommendation T.62. However, the qualifications listed in the following subsections shall also apply.

#### 4.1.1 *Session elements of procedure*

The values of mandatory parameters used during session establishment shall be as shown in Table 2/X.430.

The format of the TTXAU terminal identifier shall be that specified in Recommendation F.200. Information relating to the remote user terminal is contained in the text of the control information.

TABLE 2/X.430

**Mandatory parameter values during session establishment**

| T.62 Parameters | Call originator | |
|---|---|---|
| | Teletex terminal | TTXAU |
| Terminal identifier of calling terminal (in CSS) | TTX TID | TTXAU's TID |
| Terminal identifier of called terminal (in RSSP) | TTXAU's TID | TTX TID |
| Date and time | Date and time the Teletex terminal originated call | Date and time the TTXAU originated call |
| Service identifier | Teletex | Teletex |

*Note* – TID means *Terminal identification.*

#### 4.1.2 *Session rules*

Change Control (CSCC/RSCCP exchange) shall only occur outside action element boundaries.

### 4.2 *Document control procedures*

The document control procedures shall be in accordance with Recommendation T.62.

Control information is conveyed in either a control document or a normal document. The body of the IP-message (see Recommendation X.400) is conveyed in a sequence (at least one) of normal documents.

In a multiple-action-element session, the action elements are delimited by at least one of the following methods:

1) Implicitly, that is, by the first document of the action element being a control document.
2) Explicitly, that is, by the first document of the action element containing a parameter indicating the number of subsequent documents in the action element.

*Note* – The second method is mandatory for multiple-action-element session, where the first document of the action element is a normal document.

When a single action element is sent on a session connection, the action element is delimited only by the end of the session.

## 5 Error recovery

This section specifies the error recovery mechanisms for errors which may occur during the communication between a Teletex terminal and a TTXAU. The type of recovery will depend upon whether the error occurred during the transmission of control information or a normal document. The recovery mechanism of Recommendation T.62 shall apply. In addition, the qualifications given in the following paragraphs shall also apply.

For both directions of transfer, if an error is detected during the transmission of a document containing control information, the transmitted part shall be discarded and the complete action element retransmitted.

If an error is detected during the transmission of a document containing normal text, the recovery mechanism depends upon the direction of transfer. In the TTXAU-to-Teletex-terminal direction, the TTXAU shall resume the transmission of the interrupted action element. The recovery rules of Recommendation T.62 (using CDC) will apply. In the Teletex-terminal-to-TTXAU direction, three possible recovery mechanisms are identified, the choice of which is a national matter:

1) No recovery is provided. The complete action element (control information and normal documents) must be retransmitted.

2) The last acknowledged normal document is considered the end of the action element. The TTXAU will submit this part to the MTA as a complete message. If the Teletex terminal wishes to resume transmission of the interrupted action element, it must first resend the control information. There is no correlation between the original messages and the continued message at a recipient UA.

3) The TTXAU shall retain the received part up to the last acknowledged checkpoint. The originating Teletex terminal is required to resume the sending of the remainder of the action element by using the CDC protocol element as described in Recommendation T.62. The TTXAU shall automatically link the retained part with the resumed part prior to submission to the MTS. If the originating Teletex terminal does not resume the sending within a predetermined time (for example, twenty-four hours) or if the TTXAU receives a Send action element within that time, the TTXAU shall discard the received part of the action element.

*Note 1* – For failures occurring between document boundaries, the last acknowledged document shall be regarded as the end of the current action element.

*Note 2* – An Action Element, which may be composed of more than one document, cannot be considered complete before at least two documents were received.

*Note 3* – If there is a relationship between the sequence of normal documents (for example a multi-part body message) being transferred the recovery mechanism 2 should not be used.

The TTXAU shall return an Exception Reporting action element to the Teletex terminal in the following cases:

1) The action element is unrecognized.

2) The partially retained action element is discarded as described in the third recovery method and optionally, the first method.

## 6 Encoding and formatting of control information

### 6.1 *Cross-reference between action element components and their coding*

This section provides a table which maps the coding used for TTX-TTXAU interaction and the corresponding action element component. The cross-reference Table 3/X.430 comprises four columns:

1) The first column contains the element number and a possible element name, as defined in Annex B.

2) The second column contains the element values associated with a specific element, as defined in Annex B.

3) The third column contains the action element components used in the specification of the action elements in § 3.

4) The fourth column contains the references to Recommendation in which the precise definition of the components are specified. The Recommendation number is followed by the section number in which the parameter definition can be found.

TABLE 3/X.430

**Specifications and coding cross references**

| Element number: Element name | Element value | Action element component | Reference |
|---|---|---|---|
| 1: CORRELATION INFORMATION | Submitted TID<br>TTXAU TID<br>Date and Time<br>Additional Session Ref. No.<br>CD No. | Correlation Information | X.430, § 3.4.1 |
| 3: DELIVER INFORMATION | O/R Name<br><br>Received Terminal Identification | Reported Recipient Info<br>– Recipient O/R Name | X.411, § 3.4.3<br><br>For further study |
| 4: TIME OF DELIVERY | Date and Time | Delivery Time | X.411, § 3.4.3 |
| 5: DELIVERY TIME DURATION | Delivery Transmission Duration | | For further study |
| 6: NOTE | Note | | For further study |
| 7: RECEIVED RECORDED MESSAGE | Received Recorded Message | | For further study |
| 9: FAILURE CAUSE | | Reason Code | X.430, § 3.9.1 |
| 13: PRIORITY | Priority | Priority | X.411, § 3.4.2 |
| 14: ORIGINATOR | O/R Name<br>Disclosure of Recipients<br><br>Alternate Recipient Allowed<br><br>Date and Time | Originator O/R Name<br>Per Message Flag<br>– Disclose Recipients<br>Per Message Flag<br>– Alternate Recipient Allowed<br>Deferred Delivery Time | X.411, § 3.4.1<br>X.411, § 3.4.2<br><br>X.411, § 3.4.2<br><br>X.411, § 4.3.2 |
| 15: RECIPIENTS | O/R Name<br><br>User Report Request<br><br><br>Explicit Conversion | Recipient Info<br>– Recipient O/R Name<br>Recipient Info<br>– Per Recipient Flag<br>– Users Report Request<br>Recipient Info<br>– Explicit Conversion | X.411, § 3.4.2<br><br>X.411, § 3.4.2<br><br><br>X.411, § 3.4.2 |
| 16: CONVERSION | Conversion Prohibition | Per Message Flag<br>– Conversion Prohibited | X.411, § 3.4.2 |
| 17: CONTENT INFO | UA Content ID<br>Content Return | UA Content ID<br>Per Message Flag<br>– Content Return Request | X.411, § 3.4.1<br>X.411, § 3.4.2 |

TABLE 3/X.430 *(cont.)*

| Element number: Element name | Element value | Action element component | Reference |
|---|---|---|---|
| 18: CONTENT INDICATORS | Importance Indicator | Importance | X.420, § 3.2.1 |
| | Sensitivity Indicator | Sensitivity | X.420, § 3.2.1 |
| | Date and Time | Expiry Date | X.420, § 3.2.1 |
| | Auto-forwarded Indicator | Auto-forwarded | X.420, § 3.2.1 |
| 20: FROM | Originator/Recipient Descriptor | Originator | X.420, § 3.2.1 |
| 21: AUTHORIZED | Originator/Recipient Descriptor | Authorizing Users | X.420, § 3.2.1 |
| 22: TO | Originator/Recipient Descriptor | Primary Recipients<br>– Recipient<br>– O/R Descriptor | X.420, § 3.2.1 |
| | Receipt Notification Indicator | Primary Recipients<br>– Recipient<br>– Report Request<br>– Receipt Notification | X.420, § 3.2.1 |
| | Non-Receipt Notification Indicator | Primary Recipients<br>– Recipient<br>– Report Request<br>– Non-Receipt Notification | X.420, § 3.2.1 |
| | Return Message Indicator | Primary Recipients<br>– Recipient<br>– Report Request<br>– Return IP-Message | X.420, § 3.2.1 |
| | Reply Request Indicator | Primary Recipients<br>– Recipient<br>– Reply Request | X.420, § 3.2.1 |
| 23: CC | Originator/Recipient Descriptor | Copy Recipients<br>– Recipient | X.420, § 3.2.1 |
| | Receipt Notification Indicator | Copy Recipients<br>– Recipient (see 22:) | X.420, § 3.2.1 |
| | Non-Receipt Notification Indicator | Copy Recipients<br>– Recipient (see 22:) | X.420, § 3.2.1 |
| | Return Message Indicator | Copy Recipients<br>– Recipient (see 22:) | X.420, § 3.2.1 |
| | Reply Request Indicator | Copy Recipients<br>– Recipient (see 22:) | X.420, § 3.2.1 |
| 24: BCC | Originator/Recipient Descriptor | Blind Copy Recipient<br>– Recipient (see 22:) | X.420, § 3.2.1 |
| | Receipt Notification Indicator | Blind Copy Recipient<br>– Recipient (see 22:) | X.420, § 3.2.1 |
| | Non-Receipt Notification Indicator | Blind Copy Recipient<br>– Recipient (see 22:) | X.420, § 3.2.1 |
| | Return Message Indicator | Blind Copy Recipient<br>– Recipient (see 22:) | X.420, § 3.2.1 |
| | Reply Request Indicator | Blind Copy Recipient<br>– Recipient (see 22:) | X.420, § 3.2.1 |

TABLE 3/X.430 *(cont.)*

| Element number: Element name | Element value | Action element component | Reference |
|---|---|---|---|
| 25: REPLY | Date and Time<br>Originator/Recipient Descriptor | Reply By Time<br>Reply To Users<br>– O/R Descriptor | X.420, § 3.2.1<br>X.420, § 3.2.1 |
| 26: SUBJECT | Subject Identifier | Subject | X.420, § 3.2.1 |
| 27: MESSAGE ID | Message Identifier | IP Message Id | X.420, § 3.1.1 |
| 28: CROSS REF | Message Identifier | Cross-References | X.420, § 3.2.1 |
| 29: OBSOLETES | Message Identifier | Obsoletes | X.420, § 3.2.1 |
| 30: IN REPLY TO | Message Identifier | In Reply To | X.420, § 3.2.1 |
| 31: BODY TYPE | Body Part | Body Part | X.420, § 3.2.2 |
| 32: FORWARD INFO | Date and Time<br>Delivery Envelope | Body Part<br>– Forwarded Message<br>– Delivery Time<br>Body Part<br>– Forwarded Message<br>– Delivery Information | X.420, § 3.2.2<br>X.420, § 3.2.2 |
| 33: SUBMISSION TIME | Date and Time | Submission Time | X.411, § 4.3.3 |
| 35: EVENT ID | Event ID | Delivery Event ID | X.411, § 4.3.3 |
| 36: UA CONTENT ID | UA Content ID | UA Content ID | X.411, § 3.4.1 |
| 37: MESSAGE LENGTH | Message Length | Message Length | X.430, § 3.5.1 |
| 38: RETRIEVAL IDENTIFIER | Retrieval Identifier | Retrieval Identifier | X.430, § 3.5.2 |
| 39: THIS RECIPIENT | O/R Name | This Recipient | X.411, § 4.3.3 |
| 40: INTENDED RECIPIENT | O/R Name | Intended Recipient | X.411, § 4.3.3 |
| 41: OTHER RECIPIENTS | O/R | Other Recipients | X.411, § 4.3.3 |
| 42: CONVERSION INDICATION | Encoded Information Type<br>Conversion Prohibition | Original Encoded Information Type<br>Conversion Prohibited | X.411, § 3.4.2<br>X.411, § 3.4.2 |
| 43: NOTIFICATION TYPE | Report Type | Report Type | X.430, § 3.4.2 |
| 44: CONVERTED INFORMATION TYPE | Encoded Information Type | Converted Encoded Information Types | X.411, § 3.4.3 |
| 45: TYPE OF UA | Type of UA | Delivered Info<br>– Type of UA | X.411, § 3.4.3 |

TABLE 3/X.430 *(end)*

| Element number: Element name | Element value | Action element component | Reference |
|---|---|---|---|
| 46: FAILURE REASON | Reason Code<br><br>Diagnostic Code | Non-Delivered Info<br>– Reason<br>Non-Delivered Info<br>– Diagnostic Code | X.411, § 3.4.3<br><br>X.411, § 3.4.3 |
| 47: TIME OF RECEIPT | Date and Time | Receipt Information<br>– Receipt Time | X.420, § 3.3 |
| 48: TYPE OF RECEIPT | Type of Receipt | Receipt Information<br>– Type of Receipt | X.420, § 3.3 |
| 49: NON-RECEIPT REASON | Reason<br>Non-Receipt Qualifier | Non-Receipt Information<br>– Reason<br>– Non-Receipt Qualifier | X.420, § 3.3 |
| 50: COMMENTS | Comments | Non-Receipt Information<br>– Comments | X.420, § 3.3 |
| 51: MESSAGE RETURNED INDICATION | | Non-Receipt Information<br>– Returned IP-Message | X.420, § 3.3 |
| 52: MESSAGE TYPE | Message Type | Message Type | X.430, § 3.6.1 |
| 53: WINDOW SIZE | Window Size | Window Size | T.62, § 4.3 |
| 54: GRAPHIC CHAR SETS | Character Set | Character Set | T.62, § 5.7.4 |
| 55: CONTROL CHAR SETS | Character Set | Character Set | T.62, § 5.7.4 |
| 56: PAGE FORMATS | Paper Size | Paper Size | T.62, § 5.7.4 |
| 57: OTHER TERM CAPABILITIES | Terminal Capabilities | Terminal Capabilities | T.62, § 5.7.4 |
| 58: DOCUMENT STORAGE MODE | Document Storage Mode Indicator | Document Storage Mode | X.430, § 3.7.1 |
| 59: TTXAU OPERATION | Error Recovery Mode Indicator<br>Receipt Notification Mode | Error Recovery Mode<br>Receipt Notification Mode | X.430, § 3.7.1<br>X.430, § 3.7.1 |
| 60: AUTO OUTPUT | Frequency<br>Date and Time | Document Storage Mode | |
| 61: CONTENT LENGTH | Content Length | Content Length | X.411, § 3.4.4 |
| 62: QUANTITY OF DOCUMENTS | No. of Documents | Number of Associated Normal Documents | X.430, § 3.1.1 |

## 6.2 *Format of Teletex action elements*

In this section the formats of the control information for different action elements are shown. The principles of encoding are given in Annex B. The formats of elements as defined in Annex B are illustrated by the use of the following four different syntax elements:

1) The element number field is represented by a sequence of numeric graphic characters.
2) The element name field is represented by a text string giving the CCITT language reference name of the field. The actual value shall be a language-dependent representation of that reference name.
3) Separators are shown as they shall be represented in the actual control information.
4) Element value fields are shown in square brackets (" [ " and " ] "). The actual parameter values are described in § 6.3.

### 6.2.1 *Message Sending action*

#### 6.2.1.1 *Send action element format*

3.1: SEND:

62: QUANTITY OF DOCS: [No. of Documents]

13: PRIORITY: [Priority]

14: ORIGINATOR:
[O/R Name] "NL"
= [Disclosure of Recipients] [Alternate Recipient Allowed] [Date and Time]

15: RECIPIENTS:
[[O/R Name] "NL"
= [User Report Request] [Explicit Conversion]]

16: CONVERSION: [Conversion Prohibition]

17: CONTENT INFO: [UA Content ID] "NL"
= [Content Return]

18: CONTENT INDICATORS:
[Importance Indicator] [Sensitivity Indicator] [Date and Time]

20: FROM:
[Originator/Recipient Descriptor]

21: AUTHORIZED:
[[Originator/Recipient Descriptor]]

22: TO:
[[Originator/Recipient Descriptor] "NL"
= [Receipt Notification Indicator] [Non-receipt Notification Indicator] [Return Message Indicator] [Reply Request Indicator]]

23: CC:
[[Originator/Recipient Descriptor] "NL"
= [Receipt Notification Indicator] [Non-receipt Notification Indicator] [Return Message Indicator] [Reply Request Indicator]]

24: BCC:
[[Originator/Recipient Descriptor] "NL"
= [Receipt Notification Indicator] [Non-receipt Notification Indicator] [Return Message Indicator] [Reply Request Indicator]]

25: REPLY:
[Date and Time]
[[Originator/Recipient Descriptor]]

26: SUBJECT:
[Subject Identifier]

27: MESSAGE ID:
[Message Identifier]

28: CROSS REF:
[[Message Identifier]]

29: OBSOLETES:
[[Message Identifier]]

30: IN REPLY TO:
[[Message Identifier]]

6.2.2 *Probe action*

6.2.2.1 *Probe action element format*

3.2: PROBE:

62: QUANTITY OF DOCS: [No. of Documents]

61: CONTENT LENGTH: [Content Length]

14: ORIGINATOR:
[O/R Name] "NL"
= [Disclosure of Recipients] [Alternate Recipient Allowed]

15: RECIPIENTS:
[[O/R Name] "NL"
= [User Report Request] [Date and Time] [Explicit Conversion]]

16: CONVERSION: [Conversion Prohibition]

17: CONTENT INFO: [UA Content ID]

6.2.3 *Message Delivery action*

6.2.3.1 *Message Delivery action element format*

3.3: DELIVERY:

62: QUANTITY OF DOCS: [No. of Documents]

13: PRIORITY: [Priority]

14: ORIGINATOR:
[O/R Name]

39: THIS RECIPIENT:
[O/R Name]

40: INTENDED RECIPIENTS:
[O/R Name]

41: OTHER RECIPIENTS:
[[O/R Name] "NL"]

33: SUBMISSION TIME: [Date and Time]

4: TIME OF DELIVERY: [Date and Time]

35: EVENT ID:
[Event ID]

42: CONVERSION INDICATION: [[Encoded Information Type]] [Conversion Prohibition]

18: CONTENT INDICATORS: [Importance Indicator] [Sensitivity Indicator] [Date and Time] [Auto-forwarded Indicator]

20: FROM:
[Originator/Recipient Descriptor]

21: AUTHORIZED:
[[Originator/Recipient Descriptor]]

22: TO:
[[Originator/Recipient Descriptor] "NL"
= [Receipt Notification Indicator] [Non-receipt Notification Indicator] [Return Message Indicator] [Reply Request Indicator]]

23: CC:
[[Originator/Recipient Descriptor] "NL"
= [Receipt Notification Indicator] [Non-receipt Notification Indicator] [Return Message Indicator] [Reply Request Indicator]]

24: BCC:
[[Originator/Recipient Descriptor] "NL"
= [Receipt Notification Indicator] [Non-receipt Notification Indicator] [Return Message Indicator] [Reply Request Indicator]]

25: REPLY:
[Date and Time]
[[Originator/Recipient Descriptor]]

26: SUBJECT:
[Subject Identifier]

27: MESSAGE ID:
[Message Identifier]

28: CROSS REF:
[[Message Identifier]]

29: OBSOLETES:
[[Message Identifier]]

30: IN REPLY TO:
[[Message Identifier]]

31: BODY TYPE: [Body Part]

32: FORWARD INFO: [Date and Time] SP [Delivery Envelope]

44: CONVERTED INFORMATIONTYPE: [[Encoded Information Type]]

### 6.2.4 *Notification action*

#### 6.2.4.1 *Delivery Status Notification action element format*

3.4: DELIVERY STATUS NOTIFICATION:

62: QUANTITY OF DOCS: [No. of Documents]

1: CORRELATION INFORMATION:
[Submitted TID]/[TTX TID]/[Date and Time]/[CD No.]/
[Additional Session Reference No.]

36: UA CONTENT ID:
[UA Content ID]

43: NOTIFICATION Type: [Report Type]

40: INTENDED RECIPIENT:
[O/R Name]

44: CONVERTED INFORMATION TYPE:
[[Encoded Information Type]]

4: TIME OF DELIVERY: [Date and Time]

45: TYPE OF UA: [Type of UA]

3: DELIVERY INFORMATION: [O/R Name]

5: DELIVERY TRANSMISSION DURATION: [Delivery Transmission Duration]

7: RECEIVED RECORDED MESSAGE:
[Received Recorded Message]

6: NOTE:
[Note]

46: FAILURE REASON: [[Reason Code] "NL"
= [Diagnostic Code]]

6.2.4.2 *Receipt Status Notification action element format*

3.5: RECEIPT STATUS NOTIFICATION:

62: QUANTITY OF DOCS: [No. of Documents]

13: PRIORITY: [Priority]

14: ORIGINATOR:
[O/R Name]

39: THIS RECIPIENT:
[O/R Name]

40: INTENDED RECIPIENT:
[[O/R Name]]

41: OTHER RECIPIENT:
[[O/R Name] "NL"]

33: SUBMISSION TIME: [Date and Time]

4: TIME OF DELIVERY: [Date and Time]

35: EVENT ID:
[Event ID]

42: CONVERSION INDICATION: [[Encoded Information Type]] [Conversion Prohibition]

43: NOTIFICATION TYPE: [Report Type]

27: MESSAGE ID: [Message Identifier]

44: CONVERTED INFORMATION TYPE:
[Encoded Information Types]

47: TIME OF RECEIPT: [Date and Time]

48: TYPE OF RECEIPT: [Type of Receipt]

49: NON-RECEIPT REASON: [Reason] "NL"
= [Non-receipt Qualifier]

50: COMMENTS:
[Comments]

51: MESSAGE RETURNED INDICATION:

6.2.5 *Document Storage Management action*

6.2.5.1 *DS Query action element format*

3.7: DS QUERY:

62: QUANTITY OF DOCS: [No. of Documents]

6.2.5.2 *DS Report action element format*

3.8: DS REPORT:

62: QUANTITY OF DOCS: [No. of Documents]

52: MESSAGE TYPE: [Message Type]

13: PRIORITY: [Priority]

37: MESSAGE LENGTH: [Message Length]

14: ORIGINATOR:
[O/R Name]

38: RETRIEVAL IDENTIFIER: [Retrieval Identifier]

6.2.5.3 *Output Request action element format*

3.9: OUTPUT REQUEST:

62: QUANTITY OF DOCS: [No. of Documents]

38: RETRIEVAL IDENTIFIER: [Retrieval Identifier]

6.2.6 *Output action*

6.2.6.1 *Output Message action element format*

3.10: OUTPUT MESSAGE:

62: QUANTITY OF DOCS: [No. of Documents]

52: MESSAGE TYPE: [Message Type]

38: RETRIEVAL IDENTIFIER: [Retrieval Identifier]

The remaining elements of the Output Message action element are identical to the elements in the Message Delivery, Delivery Status Notification, or Receipt Status Notification action element. The actual elements to be used depend upon the Message Type parameter value specified in the Message Type element.

6.2.7 *Registration action*

6.2.7.1 *Registration action element format*

3.11: REGISTRATION:

62: QUANTITY OF DOCS: [No. of Documents]

53: WINDOW SIZE: [Window Size]

54: GRAPHIC CHAR SETS: [Character Set]

55: CONTROL CHAR SETS: [Character Set]

56: PAGE FORMATS: [Paper Size]

57: OTHER TERM CAPABILITY: [Terminal Capability]

37: MESSAGE LENGTH: [Message Length]

58: DOCUMENT STORAGE MODE: [Document Storage Mode Indicator]

59: TTXAU OPERATION: [Implicit Conversion] [Error Recovery Mode Indicator] [Receipt Notification Mode]

60: AUTO-OUTPUT: [Date and Time] [Frequency]

6.2.8 *Receipt Acknowledgment action*

6.2.8.1 *Explicit Receipt Acknowledgment action element format*

3.6: EXPLICIT RECEIPT ACKNOWLEDGMENT:

62: QUANTITY OF DOCS: [No. of Documents]

27: MESSAGE ID: [Message Indentifier]

40: INTENDED RECIPIENT:
[O/R Name]

44: CONVERTED INFORMATION TYPE:
[Encoded Information Type]

6.2.9 *Exception Reporting action*

6.2.9.1 *Exception Reporting action element format*

3.12: EXCEPTION REPORTING:

62: QUANTITY OF DOCS: [No. of Documents]

1: CORRELATION INFORMATION:
[Submitted TID]/[TTX TID]/[Date and Time]/[CD No.]/
[Additional Session Reference No.]

9: FAILURE CAUSE: [Failure Reason]

6.3 *Description of element values in action elements*

The values of the parameters shall be based upon the T.61 graphic character repertoire and are defined nationally. They are described below in alphabetical order.

6.3.1 *Pre-defined element values*

There is specific set of values for the pre-defined element values, the coding of each of which is unique. If these parameters are not present, a default value shall apply in some specific cases. They are described below in alphabetical order.

*[Alternate Recipient Allowed]*

This parameter has two values, which are used to indicate the following:

1) Allowing of alternate recipient
2) No alternate recipient allowed (the default)

*[Auto-forwarded Indicator]*

This parameter has two values, which are used to indicate the following:

1) Auto-forwarded message
2) Not an auto-forwarded message (the default)

*[Auto-receipt Notification Indicator]*

The values of this parameter are used to indicate whether or not auto-receipt notification is required. The default is no auto-receipt notification.

*[Content Return]*

This parameter has two values, which are used to indicate the following:

1) Requesting content return
2) Not requiring content return (the default)

*[Conversion Prohibition]*

This parameter has two values, which are used to indicate the following:

1) Prohibit conversion
2) No prohibition (the default)

*[Diagnostic Code]*

This parameter conveys the diagnostic code of the types in § 3.4.3.2 of Recommendation X.411.

*[Disclosure of Recipients]*

This parameter has two values, which are used to indicate the following:

1) Allow disclosure
2) No disclosure allowed (the default)

*[Document Storage Mode Indicator]*

This parameter has two values, which are used to indicate the following (there is no default):

1) Auto-delivery mode
2) Retrieval mode

*[Encoded Information Type]*

This parameter value shall indicate the encoded information of the types in § 3.4.1.4 of Recommendation X.411.

*[Error Recovery Mode Indicator]*

The parameter values indicate the error recovery mode being used, as specified in § 5.

*[Explicit Conversion]*

This parameter value shall indicate the explicit conversion values of the types in § 3.4.2.1 of Recommendation X.411.

*[Failure Reason]*

The parameter values shall indicate a set of defined failure reasons.

*[Importance Indicator]*

This parameter has three values, which are used to indicate the following levels of importance:

1) Low
2) Normal (the default)
3) High

*[Message Type]*

This parameter has three values, which are used to indicate the following:

1) IP-message
2) Delivery status notification
3) Receipt status notification

*[Non-receipt Notification Indicator]*

This parameter value shall indicate that non-receipt notification is required. The default is not requiring non-receipt notification.

*[Non-receipt Qualifier]*

This parameter value shall indicate the non-receipt qualifier of the types in § 3.3 of Recommendation X.420.

*[Priority]*

This parameter has three values, which are used to indicate the following priorities:

1) Urgent
2) Non-urgent
3) Normal (the default)

*Reason*

This parameter value shall indicate the reason parameter of the types in § 3.3 of Recommendation X.420.

*[Reason Code]*

This parameter conveys the reason code of the types in § 3.4.3.2 of Recommendation X.411.

*[Receipt Notification Indicator]*

This parameter value is used to indicate whether receipt notification is required. The default is that no receipt notification is required.

*[Reply Request Indicator]*

This parameter value is used to indicate whether reply request is required. The default is that no reply is requested.

*[Report Type]*

This parameter has four values, which are used to indicate the following:

1) Receipt
2) Non-receipt
3) Delivery
4) Non-delivery

*[Return Message Indicator]*

There are two parameter values. This is used to indicate whether return of the message is required. The default is not requiring return of the message.

*[Sensitivity Indicator]*

This parameter has four values, which are used to indicate the following sensitivity levels:

1) Personal
2) Private
3) Company confidential
4) Normal (the default)

*[Type of Receipt]*

This parameter has two values, which are used to indicate the following types of receipt:

1) Explicit
2) Automatic

*[Type of UA]*

This parameter has two values, which are used to indicate whether the UA is private or public.

*[User Report Request]*

This parameter has two values, which are used to indicate one of the following:

1) No report at all (that is prevention of non-delivery notification)
2) Report for every case (including notification in case of delivery)

The default value is to convey non-delivery notification.

6.3.2 *General element values*

General parameters are those with a wide range of values. They are classified into the following three groups:

1) General element values that are described in Recommendation T.90.
   a) Additional Session Reference Number
   b) CD no
   c) Submitted TID
   d) TTXAU TID
2) General element values that consist of a sequence of graphic characters:
   a) Comments
   b) Date and Time (as defined in Figure 5 of Recommendation F.200)
   c) Delivery Envelope
   d) Event ID
   e) Message Identifier
   f) O/R Name (one or more lines with the format described in § 3 of Recommendation X.400)
   g) Originator/Recipient Descriptor (one or more lines with the format described in § 3 of Recommendation X.400)
   h) Subject Identifier
   i) UA Content ID
   j) Retrieval Identifier (however, it may have a parameter value of alphanumeric graphic characters to indicate the retrieval of all messages)
   k) Character set
   l) Paper size
   m) Terminal capabilities
3) General element values that consist of a sequence of numeric graphic characters:
   a) Content Length (in octets)
   b) Frequency (in minutes)
   c) Message Length (in octets)
   d) No. of Documents
   e) Window size

ANNEX A

(to Recommendation X.430)

**Abbreviations**

The following abbreviations are used in this Recommendation.

| | |
|---|---|
| BCC | Blind courtesy copy |
| BS | Back space |
| C | Conditional |
| CC | Courtesy copy |
| CD | Control document |
| CDC | Command document continue |
| CDS | Command document start |
| CR | Carriage return |
| CSCC | Command session change control |
| CSS | Command session start |
| DS | Document storage |
| FF | Form feed |
| ID | Identity |
| IP | Interpersonal |
| IPM | Interpersonal messaging |
| ISO | International Organisation for Standardization |
| LF | Line feed |
| M | Mandatory |
| MHS | Message handling system |
| MT | Message transfer |
| MTA | Message transfer agent |
| MTL | Message transfer layer |
| MTS | Message transfer system |
| NL | New line |
| O/R | Originator/Recipient |
| P5 | Teletex access protocol |
| RSCCP | Response session change control positive |
| RSSP | Response session start positive |
| SP | Separator |
| TID | Terminal identification |
| TTX | Teletex |
| TTXAU | Teletex access unit |
| UA | User agent |
| UAE | User agent entity |
| UAL | User agent layer |

ANNEX B

(to Recommendation X.430)

## Coding and formatting principles

### B.1 *General formatting principles*

B.1.1 The Control information is subdivided in *elements* each consisting of a number of *fields.* For Teletex terminals, the control information shall be coded using explicit, human-readable graphic characters of Recommendation T.61 coding scheme.

B.1.2 An element is uniquely identifiable by an *Element Number* field, which is language independent, and an *Element Name* field, which is language dependent. These two fields are always closed by the character "Colon" (:). In TTX-TTXAU control information, at least one of the above fields must be present.

B.1.3 Parameters to the elements shall be coded in separate *Element Value* fields. The parameters will be illustrated in this Recommendation by enclosing them between square brackets ([ ]).

B.1.4 In the text each element is contained in one or more lines. When more than one line is used, only the element Number field and the Element Name field shall be present in the first line. See Figure B-1/X.430.

B.1.5 The first element present in the control information shall be the "Action Element Identifier". This Element is mandatory and used to identify the function of the control information, e.g. submission of a message, acknowledgement of a previous message.

B.1.6 The element denoted by the name "Note" will be allowed for extending the format to accommodate national requirements.

B.1.7 An Element Name shall be represented by a *text string,* that is a sequence of graphic characters. Some parameters may also consist of such text strings. All text strings are language dependent.

B.1.8 Each element is identified by the following sequence of Characters:

CR LF BS +

B.1.9 If some parameters are absent but required, then their default values shall apply.

B.1.10 Parameters are shown in square brackets [ ]. Parameters or groups of parameters and elements which may be repeated are shown in another set of square brackets i.e.:

a) to repeat parameter use [[ ]]

b) to repeat a group of parameters use [[ ] [ ] [ ]]

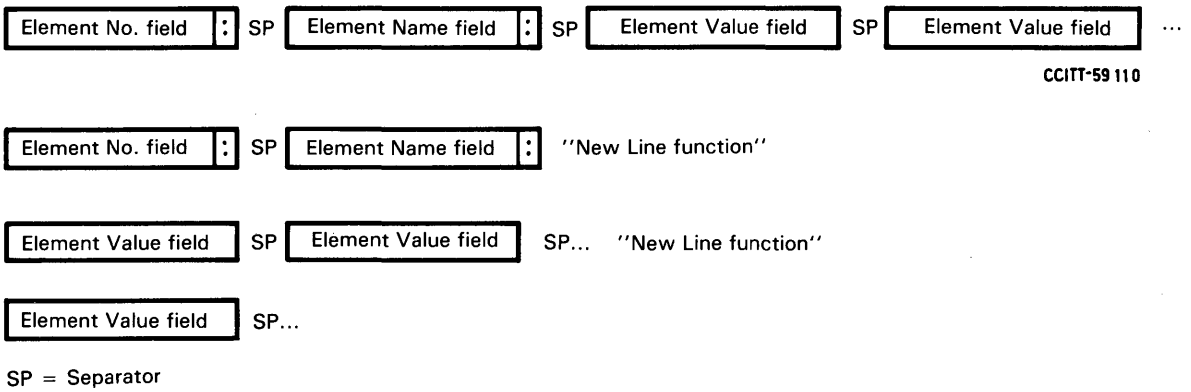



FIGURE B-1/X.430
**Formats of the elements**

B.2 *General coding principles*

B.2.1 When encoding the control information the following rules shall apply:

a) Each page (as defined in Recommendation T.61) containing control information shall always begin with the character sequence FF CR or CR FF preceded by applicable presentation control functions (see Recommendation T.61, § 3.3.1.4).

b) Each element following the "Action Element Identifier" may be preceded by one or more empty lines.

c) Each field in an element shall be separated by the character "space" if no other special separator is defined for the element.

B.2.2 When decoding the control information the following rules shall apply:

a) The first element in the text shall be recognized as the "Action Element Identifier". Leading New Line functions (CR LF or LF CR), LF and leading spaces shall be ignored.

b) Contiguous New Line functions (CR LF or LF CR) or LF shall be considered as *one* New Line function.

c) Contiguous embedded spaces are considered as one space. Leading spaces in a line shall be ignored.

B.3 *Principles for assignment of elements and parameters*

B.3.1 Element names and numbers should be easy to understand and decode without error by humans. They should preferably be short in length and simple to use.

B.3.2 It shall be possible to input the elements of a control document in any order. There shall be no restriction in the order of occurrence of the elements in a control document.

B.3.3 Elements should be automatically and unambiguously processable by machines.

B.3.4 Elements shall be independent of one another and not nested within one another.

B.3.5 Parameters shall have no identifier name or number. Some parameters shall be positionally determined and delimited by separators within an element. Other parameters shall not be positionally determined and are identified by unique parameter indicator within an element.

B.3.6 Parameters are categorized as follows:

a) Pre-defined parameters, i.e. parameters with a specific, enumerable set of known, unique values.

b) General parameters, i.e. parameters with a wide range of values which are not pre-defined.

B.3.7 When both categories of parameters are contained in an element:

a) General parameters should be separated by a unique symbol.

b) Pre-defined parameters are grouped together on a single line and preceded by a "=" sign. They shall be able to be present in any order. If absent, a default value shall apply.

B.3.8 The symbol "NL" is used in the formatting to show "New Line function".

B.4 *Assignment of Element Numbers*

B.4.1 The element number assigned to the Action Element Identifier shall consist of two parts separated by a period ".". Each part is a sequentially assigned number. The first part identifies the application which uses the control document. The second part identifies the control document.

B.4.2 Each distinct element name, other than Action Element Identifiers, shall be sequentially assigned a different number.

B.4.3 There shall be no restriction on the number of digits for element numbers. Any leading zeros in an element are ignored.

B.4.4 Where national requirements dictate the use of non-standardized element numbers, Administrations may choose any value in the range 1000-1999 for the first part of non-standardized control document identifiers and for non-standardized elements.

ANNEX C

(to Recommendation X.430)

**Formats of action element components**

TABLE C-1/X.430

**Formats of action element components**

| Action element component (§ 3) | Element format (§ 6) | |
|---|---|---|
| | Number: Name | Value |
| Action element identifier | 3.1: SEND<br>3.2: PROBE<br>3.3: DELIVERY<br>3.4: DELIVERY STATUS NOTIFICATION<br>3.5: RECEIPT STATUS NOTIFICATION<br>3.6: EXPLICIT RECEIPT ACKNOWLEDGEMENT<br>3.7: DS QUERY<br>3.8: DS REPORT<br>3.9: OUTPUT REQUEST<br>3.10: OUTPUT MESSAGE<br>3.11: REGISTRATION<br>3.12: EXCEPTION REPORTING | |
| Authorizing Users | 21: AUTHORIZED | Originator/Recipient descriptor |
| Auto forwarded | 18: CONTENT INDICATORS | auto forwarded indicator |
| Body Part<br>– Forwarded Message<br>– Delivery Time<br>– Delivery Information | 32: FORWARD INFO | <br><br>– date and time<br>– delivery envelope |
| Blind Copy Recipients<br>– Recipient<br>– O/R description<br>– Report request<br>– Receipt notification<br>– Non-Receipt notification<br>– Return IP Message<br>– Reply request | 24: BCC | <br><br>originator/recipient description<br><br>Receipt notification<br>non-Receipt notification<br>return message<br>Reply Request Indicator |
| Content Length | 61: CONTENT LENGTH | Content length |
| Converted Encoded Information Types | 44: CONVERTED INFORMATION TYPE | encoded information type |
| Conversion Prohibited | 42: CONVERSION INDICATION | conversion prohibited |
| Copy Recipients<br>– Recipient<br>– O/R description<br>– Report Request<br>– Receipt notification<br>– Non-receipt notification<br>– Return ID Message<br>– Reply Request | 23: CC | <br><br>O/R descriptor<br><br>Receipt notification<br>non-Receipt notification<br>return message<br>Reply Request Indicator |

TABLE C-1/X.430 *(cont.)*

| Action element component (§ 3) | Element format (§ 6) | |
|---|---|---|
| | Number: Name | Value |
| Correlation Information | 1: CORRELATION INFORMATION | Submitted TID<br>TTXAU TID<br>Date and Time<br>Additional session reference No.<br>CD No. |
| Cross References | 28: CROSS REFERENCE | message identifier |
| Deferred Delivery Time | 14: ORIGINATOR | date and time |
| Delivered Information<br>– Type of UA<br>– Delivery Time | <br>45: TYPE of UA<br>4: TIME OF DELIVERY | <br>type of UA<br>date and time |
| Delivery Event Id | 35: EVENT ID | event id |
| Document Storage Mode<br>– date/time<br>– Frequency | 58: DOCUMENT STORAGE MODE<br>60: AUTO-OUTPUT<br>60: AUTO-OUTPUT | document storage mode indicator<br>date and time<br>frequency |
| Error Recovery Mode | 59: TTXAU OPERATION | error recovery mode indicator |
| Expiry Date | 18: CONTENT INDICATORS | date and time |
| In Reply To | 30: IN REPLY TO | message identifier |
| Intended Recipient | 40: INTENDED RECIPIENT | O/R name |
| Importance | 18: CONTENT INDICATORS | Importance |
| IP-message Id | 27: MESSAGE ID | message identifier |
| Message Length | 37: MESSAGE LENGTH | message length |
| Message Selector | 38: RETRIEVAL IDENTIFIER | Retrieval Identifier |
| Message Type | 52: MESSAGE TYPE | message type |
| Non Delivered Info<br>– Reason<br>– Diagnostic Code | 46: FAILURE REASON | <br>reason code<br>diagnostic code |
| Non-Receipt Information<br>– Reason<br>– Non-Receipt Qualifier<br>– Comments<br>– Returned IP message | 49: NON RECEIPT REASON<br><br><br>50: COMMENTS<br>51: MESSAGE RETURN INDICATION | <br>reason<br>non-receipt qualifier<br>comments |

TABLE C-1/X.430 *(cont.)*

| Action element component (§ 3) | Element format (§ 6) | |
|---|---|---|
| | Number: Name | Value |
| Number of Associated Normal Documents | 62: QUANTITY OF DOCUMENTS | No. of Documents |
| Obsoletes | 29: OBSOLETES | message identifier |
| Original Encoded Information Types | 42: CONVERSION INDICATION | encoded Information type |
| Originator | 20: FROM | originator/recipient descriptor |
| Originator O/R Name | 14: ORIGINATOR | O/R Name |
| Other Recipients | 41: OTHER RECIPIENTS | O/R Name |
| Per Message Flag<br>– Disclose Recipients<br>– Conversion Prohibited<br>– Alternate Recipient Allowed<br>– Content Return Request | <br>14: ORIGINATOR<br>16: CONVERSION<br>14: ORIGINATOR<br>17: CONTENT INFORMATION | <br>disclosure of Recipients<br>conversion prohibition<br>alternate recipient allowed<br>content return |
| Primary Recipients<br>– Recipient<br>– O/R Descriptor<br>– Report Request<br>– Receipt notification<br>– Non-Receipt Notification<br>– Return IP message<br>– Reply Request | 22: TO | <br><br>O/R descriptor<br><br>Receipt notification<br>non-Receipt notification<br>Return message<br>Reply Request Indicator |
| Priority | 13: PRIORITY | Priority |
| Reason Code | 9: FAILURE CAUSE | |
| Receipt Notification Mode | 59: TTXAU OPERATIONS | Receipt notification mode |
| Receipt Information<br>– time<br>– type of receipt | <br>47: TIME OF RECEIPT<br>48: TYPE OF RECEIPT | <br>date and time<br>type of receipt |
| Recipient Information<br>– Recipient O/R name<br>– Per Recipient Flag<br>– Users Report Request<br>– UA Content Id<br>– Explicit Conversion | 15: RECIPIENTS<br><br><br><br>17: CONTENT INFORMATION<br>15: RECIPIENTS | <br>O/R name<br><br>users report request<br>UA content ID<br>explicit conversion |
| Reply by Time | 25: REPLY | date and time |
| Reply to Users<br>– O/R Descriptor | 25: REPLY | O/R descriptor |

TABLE C-1/X.430 *(end)*

| Action element component (§ 3) | Element format (§ 6) | |
|---|---|---|
| | Number: Name | Value |
| Report Type | 43: NOTIFICATION TYPE | report type |
| Reported Recipient Information<br>– Recipient O/R Name<br>– Supplementary Information | 3: DELIVERY INFORMATION | O/R Name |
| Retrieval Identifier | 38: RETRIEVAL IDENTIFIER | retrieval identifier |
| Sensitivity | 18: CONTENT INDICATORS | sensitivity indicator |
| Subject | 26: SUBJECT | subject |
| Submission Time | 33: SUBMISSION TIME | date and time |
| This Recipient | 39: THIS RECIPIENT | O/R name |
| UA Content Id | 17: CONTENT INFORMATION | UA Content Id |

APPENDIX I

(to Recommendation X.430)

**List of Terms**

The following terms are contained in this Recommendation. References to definitions, implicit or explicit, are included where appropriate and where available.

## A

**Action Element**
cont. in X.430. impl. def. in X.430.

**Action Element Identifier**
cont. in X.430; impl. def. in X.430

**Actions**
cont. in X.430; impl. def. in X.430

**Additional Session Reference No.**
cont. in X.430; impl. def. in X.430

**Address**
cont. in X.430; impl. def. in X.430

**Alternate Recipient Allowed**
cont. in X.430; impl. def. in X.430

**Authorizing User**
cont. in X.430; impl. def. in X.430

**Auto-forwarded**
cont. in X.430; impl. def. in X.430

**Auto-forwarded Indicator**
cont. in X.430; impl. def. in X.430

**Auto-output**
cont. in X.430; impl. def. in X.430

**Auto-receipt Notification**
cont. in X.430; impl. def. in X.430

## B

**Basic Teletex Service**
cont. in X.430; impl. def. in F.200

**BCC (Blind Courtesy Copy)**
cont. in X.430; impl. def. in X.430

**Blind Copy Recipient**
cont. in X.430; impl. def. in X.430

**Body**
cont. in X.430; impl. def. in X.430

**Body Part**
cont. in X.430; impl. def. in X.430

## C

**Call Originator**
cont. in X.430; impl. def. in X.430

**CC (Courtesy Copy)**
cont. in X.430; impl. def. in X.430

**Character Set**
cont. in X.430; impl. def. in X.430

**Comments**
cont. in X.430; impl. def. in X.430

**Component**
cont. in X.430

**Conditional**
cont. in X.430

**Content**
cont. in X.430; impl. def. in X.430

**Content Indicator**
cont. in X.430; impl. def. in X.430

**Content Length**
cont. in X.430; impl. def. in X.430

**Content Return**
cont. in X.430; impl. def. in X.430

**Content Return Request**
cont. in X.430; impl. def. in X.430

**Control Document**
cont. in X.430; impl. def. in X.430

**Control Information**
cont. in X.430; impl. def. in X.430

**Control Information Reference**
cont. in X.430; impl. def. in X.430

**Control Procedure**
cont. in X.430.

**Conversion**
cont. in X.430; impl. def. in X.408

**Conversion Indicator**
cont. in X.430; impl. def. in X.430

**Conversion Prohibited**
cont. in X.430; impl. def. in X.430

**Conversion Prohibition**
cont. in X.430; impl. def. in X.430

**Converted Encoded Information Types**
cont. in X.430; impl. def. in X.430

**Converted Information Type**
cont. in X.430; impl. def. in X.430

**Copy Recipients**
cont. in X.430; impl. def. in X.430

**Correlation Recipients**
cont. in X.430; impl. def. in X.430

**Cross-references**
cont. in X.430; impl. def. in X.430

# D

**Date and Time**
cont. in X.430; impl. def. in X.430

**Deferred Delivery Time**
cont. in X.430; impl. def. in X.430

**Deliver**
cont. in X.430; impl. def. in X.430

**Delivered Document Storage**
cont. in X.430; impl. def. in X.430

**Delivered Info**
cont. in X.430; impl. def. in X.430

**Delivery**
cont. in X.430; impl. def. in X.430

**Delivery Envelope**
cont. in X.430; impl. def. in X.430

**Delivery Event ID**
cont. in X.430; impl. def. in X.430

**Delivery Information**
cont. in X.430; impl. def. in X.430

**Delivery/Status Notification**
cont. in X.430; impl. def. in X.430

**Delivery Time**
cont. in X.430; impl. def. in X.430

**Delivery Transmission Duration**
cont. in X.430; impl. def. in X.430

**Diagnostic Code**
cont. in X.430; impl. def. in X.430

**Disclose Recipients**
cont. in X.430; impl. def. in X.430

**Disclosure of Recipients**
cont. in X.430; impl. def. in X.430

**Document**
cont. in X.430

**Document Control Procedure**
cont. in X.430

**Document Storage**
cont. in X.430; impl. def. in X.430

**Document Storage Management**
cont. in X.430; impl. def. in X.430

**Document Storage Management Action**
cont. in X.430; impl. def. in X.430

**Document Storage Mode**
cont. in X.430; impl. def. in X.430

**Document Storage Mode Indication**
cont. in X.430; impl. def. in X.430

**Document Storage Mode Indicator**
cont. in X.430; impl. def. in X.430

**DS Query**
cont. in X.430; impl. def. in X.430

**DS Query Action Element**
cont. in X.430; impl. def. in X.430

**DS Report**
cont. in X.430; impl. def. in X.430

**DS Report Action Element**
cont. in X.430; impl. def. in X.430

# E

**Element**
cont. in X.430; impl. def. in X.430

**Element Name**
cont. in X.430; impl. def. in X.430

**Element Number**
cont. in X.430; impl. def. in X.430

**Element Value**
cont. in X.430; impl. def. in X.430

**Encoded Information Type**
cont. in X.430; impl. def. in X.400

**Error Recovery**
cont. in X.430; impl. def. in X.430

**Error Recovery Mode Indication**
cont. in X.430; impl. def. in X.430

**Error Recovery Mode Indicator**
cont. in X.430; impl. def. in X.430

**Event ID**
cont. in X.430; impl. def. in X.430

**Exception Reporting**
cont. in X.430; impl. def. in X.430

**Exception Reporting Action**
cont. in X.430; impl. def. in X.430

**Exception Reporting Action Element**
cont. in X.430; impl. def. in X.430

**Expiry**
cont. in X.430; impl. def. in X.430

**Expiry Date**
cont. in X.430; impl. def. in X.430

**Explicit Conversion**
cont. in X.430; impl. def. in X.430

**Explicit Receipt Acknowledgement**
cont. in X.430; impl. def. in X.430

**Explicit Receipt Request Indicator**
cont. in X.430; impl. def. in X.430

# F

**Failure Cause**
cont. in X.430; impl. def. in X.430

**Failure Reason**
cont. in X.430; impl. def. in X.430

**Field**
cont. in X.430; impl. def. in X.430

**Forwarded Message**
cont. in X.430; impl. def. in X.430

**Forwarded Info.**
cont. in X.430; impl. def. in X.430

**Frequency (of Auto-output)**
cont. in X.430; impl. def. in X.430

**From**
cont. in X.430; impl. def. in X.430

# I

**Implicit Conversion**
cont. in X.430; impl. def. in X.430

**Implicit Conversion Enable/Disable**
cont. in X.430; impl. def. in X.430

**Importance**
cont. in X.430; impl. def. in X.430

**Importance Indicator**
cont. in X.430; impl. def. in X.430

**In Reply To**
cont. in X.430; impl. def. in X.430

**Intended Recipient**
cont. in X.430; impl. def. in X.430

**Intended Recipient O/R Name**
cont. in X.430; impl. def. in X.430

**Interpersonal Messaging**
cont. in X.430; impl. def. in X.400

**Interpersonal Messaging Heading**
cont. in X.430; impl. def. in X.430

**Interpersonal Messaging Parameters**
cont. in X.430; impl. def. in X.430

**Interpersonal Messaging Service**
cont. in X.430; impl. def. in X.400

**IP-message**
cont. in X.430; impl. def. in X.400

**IP-message ID**
cont. in X.430; impl. def. in X.430

**Invalid Action Element**
cont. in X.430; impl. def. in X.430

# M

**Mandatory**
cont. in X.430

**Message**
cont. in X.430; impl. def. in X.400

**Message Delivery**
cont. in X.430; impl. def. in X.430

**Message Delivery Action**
cont. in X.430; impl. def. in X.430

**Message Identifier**
cont. in X.430; impl. def. in X.430

**Message Indicator**
cont. in X.430; impl. def. in X.430

**Message Handling System**
cont. in X.430; impl. def. in X.400

**Message Length**
cont. in X.430; impl. def. in X.430

**Message Output**
cont. in X.430; impl. def. in X.430

**Message Parameters**
cont. in X.430; impl. def. in X.430

**Message priority**
cont. in X.430; impl. def. in X.430

**Message Returned**
cont. in X.430; impl. def. in X.430

**Message Selector**
cont. in X.430; impl. def. in X.430

**Message sending**
cont. in X.430; impl. def. in X.430

**Message Sending Action**
cont. in X.430; impl. def. in X.430

**Message Transfer**
cont. in X.430; impl. def. in X.430

**Message Transfer Agent**
cont. in X.430; impl. def. in X.400

**Message Transfer Layer**
cont. in X.430; impl. def. in X.400

**Message Transfer Parameters**
cont. in X.430; impl. def. in X.430

**Message Transfer Service**
cont. in X.430; impl. def. in X.400

**Message Transfer System**
cont. in X.430; impl. def. in X.400

**Message Type**
cont. in X.430; impl. def. in X.430

# N

**No. of Documents**
cont. in X.430; impl. def. in X.430

**Non-delivered Info**
cont. in X.430; impl. def. in X.430

**Non-delivery Info**
cont. in X.430; impl. def. in X.430

**Non-receipt Information**
cont. in X.430; impl. def. in X.430

**Non-receipt Notification**
cont. in X.430; impl. def. in X.400

**Non-receipt Notification Indicator**
cont. in X.430; impl. def. in X.430

**Non-receipt Qualifier**
cont. in X.430; impl. def. in X.430

**Non-receipt Reason**
cont. in X.430; impl. def. in X.430

**Normal Document**
cont. in X.430; impl. def. in X.430

**Note**
cont. in X.430; impl. def. in X.430

**Notification**
cont. in X.430; impl. def. in X.430

**Notification Action**
cont. in X.430; impl. def. in X.430

**Notification Type**
cont. in X.430; impl. def. in X.430

**Notify**
cont. in X.430; impl. def. in X.430

# O



# P



# Q



# R

**Report Type**
cont. in X.430; impl. def. in X.430

**Reported Recipient Info**
cont. in X.430; impl. def. in X.430

**Retrieval Identifier**
cont. in X.430; impl. def. in X.430

**Return IP-message**
cont. in X.430; impl. def. in X.430

**Returned IP-message**
cont. in X.430; impl. def. in X.430

## S

**Send**
cont. in X.430; impl. def. in X.430

**Send action Element**
cont. in X.430; impl. def. in X.430

**Sensitivity**
cont. in X.430; impl. def. in X.430

**Sensitivity Indicator**
cont. in X.430; impl. def. in X.430

**Service Element**
cont. in X.430; impl. def. in X.400

**Session Rules**
cont. in X.430; impl. def. in X.430

**Subject**
cont. in X.430; impl. def. in X.430

**Subject Identifier**
cont. in X.430; impl. def. in X.430

**Submission**
cont. in X.430; impl. def. in X.430

**Submission Time**
cont. in X.430; impl. def. in X.430

**Submit**
cont. in X.430; impl. def. in X.430

**Submitted TiD**
cont. in X.430; impl. def. in X.430

**Supplementary Info**
cont. in X.430; impl. def. in X.430

**Subscription**
cont. in X.430; impl. def. in X.430

## T

**Teletex**
cont. in X.430; impl. def. in X.430

**Teletex Access Model**
cont. in X.430; impl. def. in X.430

**Teletex Access Protocol**
cont. in X.430; impl. def. in X.430

**Teletex Action Element**
cont. in X.430; impl. def. in X.430

**Teletex Document**
cont. in X.430; impl. def. in X.430

**Teletex Service**
cont. in X.430; impl. def. in X.200

**Teletex Terminal**
cont. in X.430; impl. def. in X.430

**Teletex Terminal Profile Registration**
cont. in X.430; impl. def. in X.430

**This Recipient**
cont. in X.430; impl. def. in X.430

**Terminal Identification**
cont. in X.430; impl. def. in X.430

**Terminal Identity**
cont. in X.430; impl. def. in X.430

**Terminal Profile**
cont. in X.430; impl. def. in X.430

**Time**
cont. in X.430; impl. def. in X.430

**Time of Delivery**
cont. in X.430; impl. def. in X.430

**Time of Receipt**
cont. in X.430; impl. def. in X.430

**To**
cont. in X.430; impl. def. in X.430

**TTXAU Operation**
cont. in X.430; impl. def. in X.430

**TTX TiD**
cont. in X.430; impl. def. in X.430

**Type of Receipt**
cont. in X.430; impl. def. in X.430

**Type of UA**
cont. in X.430; impl. def. in X.430

## U

**UA Content ID**
cont. in X.430; impl. def. in X.430

**User Agent**
cont. in X.430; impl. def. in X.400

**User Agent Layer**
cont. in X.430; impl. def. in X.400

**User-ID**
cont. in X.430; impl. def. in X.430

**Users Report Request**
cont. in X.430; impl. def. in X.430

## W

**Window Size**
cont. in X.430; impl. def. in X.430

APPENDIX II

(to Recommendation X.430)

**Further considerations on document storage**

## II.1 *General*

This Appendix addresses some additional functional requirements for Document Storage (DS) which have been identified during the study of the DS.

The DS, being an additional facility of the TTXAU, caters for intermediate (additional) storage provisions between the Message Transfer System (MTS) and recipient Teletex terminals.

*Note* – For the time being, only basic Teletex terminal capabilities are assumed to be supported by the TTXAU. However, support for optional capabilities is a matter for urgent further study.

The integrity of messages stored in the DS in the TTXAU's responsibility until they are successfully delivered to the terminal.

The following rationale explains the value-added functionality of the DS for Teletex subscribers:

1) Messages submitted by Teletex or non-Teletex originators can be delivered to the DS for retrieval by the intended Teletex terminal, thus overcoming all kinds of terminal unavailability problems (for example, terminals may be temporarily unavailable due to being occupied, out of order, or short of memory).
2) Terminal users (recipients) may be mobile and have the need to retrieve messages from different locations and using various terminal types, at a time and place convenient to them (for further study).
3) Terminals are often used by more than one user (recipient). In order to offer privacy and security, particular instances of the DS can be devoted to O/R names and/or equipped with specific retrieval keys and/or passwords (for further study).

## II.2 *Details of operation and use*

### II.2.1 *Introduction*

The DS reserves a specific amount of storage for each DS subscriber, either on a predetermined or statistical basis.

DS subscriber recipients are known to the DS by their O/R names and/or Teletex TIDs and are registered as such at subscription time. An individual instance of the DS is allocated to each DS subscriber.

### II.2.2 *User of Document Storage as an enhancement of the Teletex service*

The following items were identified to meet further requirements for network storage facilities with respect to enhanced Teletex terminals:

1) The need to directly address the individual DS of the intended recipient [see Item 1) of § II.2.3].
2) The facility of automatic redirection of Teletex documents to a recipient's DS [see Item 2) of § II.2.3].
3) Indication of redirection to the Teletex originator [see Item 2) of § II.2.3].
4) Conditions under which output occurs and additional output facilities (see § II.2.4).
5) Support of optional control facilities for enhanced Teletex or Telematic terminals. This includes non-basic Teletex terminal capabilities in order to emulate the Teletex subscriber to the DS with respect to network storage of documents (see § II.2.4).

II.2.3 *Document Storage input*

Messages destined for a specific DS subscriber can be stored in the individual document storage in the following ways:

1) On request of the originator by addressing the DS of the recipient. The addressing mechanism is for further study.

2) By redirection of messages directly addressing DS subscribers. In this case the DS acts on behalf of the recipient.

   A Teletex originator who sends a normal Teletex document to the addressed recipient is made aware of the fact that the message was redirected to the DS (and not to the designated Teletex recipient) by the coding of Part 3 of the recipient TID is "+ + +" preceded by "−", to indicate DS intervention.

Delivery notification, if requested, is qualified "to DS", while the message resides under the responsibility of the TTXAU.

II.2.4 *Document Storage output*

DS output is based upon the use of the Output action and (conditionally) the Document Storage Management actions as described in the body of this Recommendation.

*Note* – The following more sophisticated output facilites are for further study:

1) Selective output retrieval on specific keys.

2) Envelope forwarding by the TTXAU in the DS Report action element.

3) Lifetime in storage under recipient control.

4) The support of enhanced Telematic terminals (such as facsimile group 4 or mixed mode) which generalizes the TTXAU to become a Telematic Access Unit (TAU).

Printed in Switzerland — ISBN 92-61-02361-4