P_MUL
- A PROTOCOL FOR RELIABLE MULTICAST MESSAGING IN BANDWIDTH CONSTRAINED AND DELAYED ACKNOWLEDGEMENT (EMCON) ENVIRONMENTS

ACP 142

DECEMBER 2001

FOREWORD

1. The Combined Communications-Electronics Board (CCEB) is comprised of the five member nations, Australia, Canada, New Zealand, United Kingdom and United States and is the Sponsoring Authority for all Allied Communications Publications (ACPs). ACPs are raised and issued under common agreement between the member nations.

2. ACP142, P_MUL - A PROTOCOL FOR RELIABLE MULTICAST MESSAGING IN BANDWIDTH CONSTRAINED AND DELAYED ACKNOWLEDGEMENT (EMCON) ENVIRONMENTS, is an UNCLASSIFIED CCEB publication.

3. This publication contains Allied military information for official purposes only.

4. It is permitted to copy or make extracts from this publication.

5. This ACP is to be maintained and amended in accordance with the provisions of the current version of ACP198.

THE COMBINED COMMUNICATION-ELECTRONICS BOARD
LETTER OF PROMULGATION
FOR ACP142

1. The purpose of this Combined Communication Electronics Board (CCEB) Letter of Promulgation is to implement ACP142 within the Armed Forces of the CCEB Nations. ACP142, P_MUL - A PROTOCOL FOR RELIABLE MULTICAST MESSAGING IN BANDWIDTH CONSTRAINED AND DELAYED ACKNOWLEDGEMENT (EMCON) ENVIRONMENTS, is an UNCLASSIFIED publication developed for Allied use and, under the direction of the CCEB Principals. It is promulgated for guidance, information, and use by the Armed Forces and other users of military communications facilities.

2. ACP142 is effective on receipt for CCEB Nations and when by the NATO Military Committee (NAMILCOM) for NATO nations and Strategic Commands.

EFFECTIVE STATUS

Publication	Effective for	Date	Authority
ACP142	CCEB	On Receipt	LOP

3. All proposed amendments to the publication are to be forwarded to the national co-ordinating authorities of the CCEB or NAMILCOM.

For the CCEB Principals

N. CRAM
Squadron Leader
Permanent Secretary to CCEB

RECORD OF MESSAGE CORRECTIONS

Identification of Change or Correction and Date, time group		Date Entered	By Whom Entered (Signature; rank, grade or rate; name of command)
Change	Correction		

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

101 Background

a. This Allied Communication Publication (ACP) was developed and co-ordinated by the Tactical Communication Task Force (TCTF) of the Allied Information Services (AIS) International Subject Matter Experts (ISME) working group under the auspices of the Combined Communications Electronics Board (CCEB). The initial mandate of this Task Force was to facilitate the interoperability of allied multicast information transfers in bandwidth-constrained and delayed-acknowledgement environments.

b. The function of this document, ACP 142, is to provide a protocol definition for reliable multicast information transfer in bandwidth-constrained and delayed-acknowledgement environments to support efficient allied information transfer. This document also provides examples illustrating various functional aspects of the protocol as well as some guidance on its use and implementation.

c. The protocol described in this document, P_MUL, is the result of a development effort under the multi-national Communications System Networks Interoperability (CSNI) project and enhancements to this protocol under various Joint Warfighter Interoperability Demonstration (JWID) experiments. Every effort has been made to make this protocol simple; transparent to the application it supports, as easy as possible to implement and platform-independent.

102 Evolution

a. Version 1 of the ACP provides the basic specification of the P_MUL protocol. The current specification of this protocol provides a new allied capability, namely reliable multicast information transfer in delayed-acknowledgement environments, in addition to providing a mechanism for more efficient use of bandwidth.

b It is intended that a subsequent version will specify significant improvements to this protocol in areas such as flow control, compression, acknowledgement and error correction, to further improve the operational gains associated with the use of P_MUL. It is also intended that this next release of the ACP will provide additional guidance regarding the implementation and use of P_MUL.

c. Efforts to standardise this protocol in other forums such as NATO and the Internet Engineering Task Force (IETF) is encouraged and supported to the practical extent possible.

103 Scope

a. The objective of ACP 142 is to specify and standardise the P_MUL protocol. This document defines the various states of the P_MUL transmitters and receivers when utilising the P_MUL protocol. Within the P_MUL application there are many configurable parameters that can be used to optimise P_MUL for an operational environment. This document does not mandate a particular setting for any parameter nor dictate policy for any nation for conducting a multicast service. Operational guidance is provided within this document to provide examples of how some particular parameters affect the performance of the protocol.

b. P_MUL, as a reliable multicast protocol, requires an underlying connectionless network infrastructure with multicast routing functionality. Current P_MUL implementations, as an example depicted in Figure 1-1, use the User Diagram Protocol (UDP) [UDP80] and Internet Protocol (IP). However its use in not limited to this protocol stack.

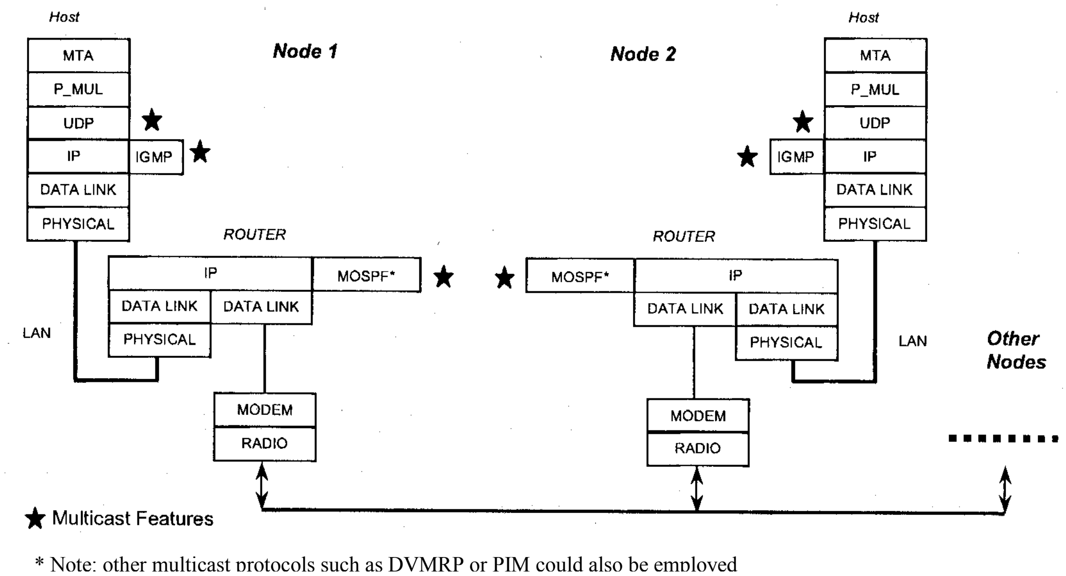


Fig 1-1: An implementation of P_MUL on a UDP/IP stack

c. This document is the standard that ensures allied interoperability. Vendors are allowed to develop products in accordance with this standard. Note the implementation of dynamic multicasting group formation (Chapter 4) is an optional capability in this Release of ACP 142.

104 Overview

a. Within this ACP defining the P_MUL protocol the word "message" is to be understood as a data element such as a P772 military message, an email, a X.400 P1 information structure, or data file.

b. This protocol takes advantage of a multicast communication service to transfer messages between different nodes on a single multicast network under both normal (which means duplex, half or full, oriented communication conditions) and under Emission Control (EMCON) conditions. EMCON or "Radio Silence" condition means that a receiving node is able to receive messages but cannot acknowledge received messages for a relatively long time (eg hours or days).

c. Figure 1-2 illustrates a simple multicast scenario, where the same message is transmitted from transmitting node, Node 0, to receiving nodes, Node 1, Node 2 and Node 3. Using a multicast communication service instead of a unicast communication service, only one message is transmitted from Node 0 to the Router, instead of three as required by unicast. This saves transmission of two messages and consequently conserves network bandwidth. Depending on the transmission rate (in some radio networks this is less than 9.6 Kb/s), this saving can be significant. Clearly, the conservation in bandwidth increases with increasing numbers of receiving nodes.

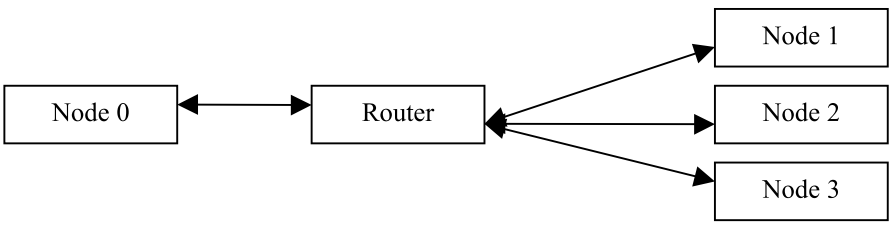


Figure 1-2: Typical messaging or file distribution configuration

d. The P_MUL protocol may be understood as an application layer protocol, in the same way as the P1 protocol, defined in the X.400 Recommendations [CCITT88] or the SMTP as specified in RFC 821 [SMTP82-1]. As in the case of these protocols, P_MUL utilises lower layer protocols to transmit its PDUs (Protocol Data Units) over a multicast network.

e. Considering the fact that nodes under EMCON are not allowed to acknowledge messages, they are unable to use a reliable transport protocol like XTP [XTP95] for the transmission of messages. Therefore, P_MUL is based on the use of a connectionless transport protocol, such as UDP.

f. Although P_MUL is based on a connectionless transport protocol, it provides users with reliable connection-oriented multicast services. It enables the receivers to receive messages while being under EMCON restrictions. It ensures that the transmitter is informed about the timely completeness of the transmission of the messages after the receivers leave the EMCON status and, if required, enables the re-transmission of any messages that were not properly received.

g. The P_MUL protocol has been applied to the communication between X.400 MTAs and between RFC 822 [SMTP82-2] mail servers, but it is not constrained to these applications and has also been used to support reliable multicast transmission of other kinds of data, eg file distribution.

h. For X.400, P_MUL is implemented within an MTA. A sample implementation has been done using the X.400 Message Handling System package available from ISODE Ltd. In this case two channel programs "Multicast_OUT" and "Multicast_IN" have been implemented and integrated. "Multicast_OUT" handles the sending of messages to the multicast-connected neighbour MTAs. "Multicast_IN" handles the incoming messages from the multicast-connected neighbour MTAs. These channels play the role of the "Higher Management Functions" mentioned in following chapters.

i. For the RFC 822 mail application P_MUL has to be implemented on behalf of a special SMTP mailer. A prototype implementation has been developed for the Sendmail system (Version 8.8.x).

j. A further application of P_MUL has been as a file transfer protocol "Multicast File Distribution". In this case the application allows a user to simultaneously distribute a single file to a set of receiving nodes, eg distribution of a software or graphics file. Because a file distribution application of this kind did not exist, a relatively simple user interface was applied to the P_MUL implementation.

k. The multicast service provided by P_MUL is to some extent "self-contained". It makes use of standard features of multicast routing protocols such as DVMRP, MOSPF and PIM, and can be used with any of them. P_MUL requires that the router be able to support static multicast routing. It also requires that the router or the subnetwork interface device be able to support a simplex link to operate in EMCON.

l. It is envisaged that P_MUL will be deployed in a network environment consisting of a few nodes to hundreds of nodes.

m. P_MUL works for fixed multicast groups and can also work for dynamically formed ad hoc multicast groups. For fixed multicast groups each node has knowledge about the group memberships of one or more multicast groups. Dynamically formed ad hoc multicast groups are used for transmitting single messages.

105 Structure of Document

Chapter 1 contains an overview of the P_MUL protocol, Definitions and Abbreviations. Chapter 2 describes the structure of the P_MUL Protocol Data Units. Chapter 3 describes the P_MUL messaging processes. Chapter 4 describes the procedures to dynamically form multicast groups. Chapter 5 contains references. Annex A contains examples illustrating the operation of the protocol. Annex B contains a table of predefined protocol parameters, recommended multicast addresses, UDP port numbers, and the checksum algorithm. Annex C provides a set of state diagrams depicting the messaging procedures. Annex D gives an operational guidance. Annex E lists a set of definitions of commonly used glossary.

106. Definitions

The definitions of a set of related and commonly used terminology are provided in Annex E.

107 Abbreviations

ACP	Allied Communication Publication
DVMRP	Distance Vector Multicast Routing Protocol
EMCON	Emission Control
GG	Global Group
IGMP	Internet Group Management Protocol
IP	Internet Protocol
ISO	International Organisation for Standardisation
MADCAP	Multicast Address Dynamic Client Allocation Protocol
MAP	More Address_PDUs
MOSPF	Multicast Extension of OSPF

MTA	Message Transfer Agent
OSPF	Open Shortest Path First an Interior Gateway Protocol
P1	Message Transfer Protocol in X.400
P_MUL	A protocol for reliable multicast messaging in bandwidth constrained and delayed acknowledgement (EMCON) environments
P772	A Military Messaging Protocol
PDU	Protocol Data Unit
PIM	Protocol Independent Multicast
RFC	Request for Comment
SMTP	Simple Mail Transfer Protocol
UDP	User Datagram Protocol
X.400	CCITT series of Recommendations on Message Handling Systems
XTP	Xpress Transport Protocol

108 Conventions

Fields and variable names are made from a string of self-explanatory words connected by underscore(s). Each non-connection word is printed in lower case with the first letter in upper case. The connection words and prepositions are printed in lower cases while abbreviations are printed in upper cases. Examples are Announce_PDU, Message_ID and Expiry_Time. Constants and pre-defined values are made from self-explanatory words all in upper cases, such as ACK_PDU_TIME.

CHAPTER 2

P_MUL PROTOCOL DATA UNITS

201 General

a. This chapter describes the protocol data units employed in P_MUL. All integers are transmitted in big endian format.

b. There are two groups of protocol data units (PDUs). The first group consists of PDUs required for the data transfer while the second group consists of PDUs used only for dynamic configuration of multicast groups. If dynamic allocation of multicast groups is not required, then only the first group of PDUs is needed (para 202 to 206).

c. As the P_MUL implementing processes have to track the states of the message transfer (connections) between the transmitting node and all receiving nodes, all PDUs are transmitted using the UDP.

d. Normally, all PDUs are delivered by the transmitting node using the multicast communication service provided in the lower layer. Only in the case where there is only one receiving node, not in EMCON, would the transmitting node switch into unicast addressing mode. All PDUs sent from receiving nodes back to the transmitting node are transmitted in unicast mode.

202 PDUs for Data Transfer

The number of PDUs used by P_MUL for data transfer is determined mainly by the requirement that the protocol be able to support communication under EMCON or "Radio Silence" conditions, ie simplex communication. Under these conditions, one or more nodes are able to receive messages but are not permitted to provide acknowledgements. Such a restriction can exist for long periods of time, eg hours or days. P_MUL uses four different PDUs for the data transfer as follows:

- Transmitter identifies message addresses (Address_PDU)
- Transmission of message fragments (Data_PDU)
- Receiver acknowledges message reception (Ack_PDU)
- Transmitter terminates the transmission of a specific message (Discard_Message_PDU)

203 Address_PDU

a. The P_MUL transmitter generates the Address_PDU to announce intended recipients of a message and provides the Message_ID. This PDU and the Ack_PDU (para 206) effect flow control of P_MUL packets. As P_MUL has to observe a maximum PDU size and as the number of Destination_Entries has no maximum value, it is essential that the total address information be able to be split into more than one Address_PDU. To distinguish between the first, middle and last Address_PDU the MAP field is used. The structure of Address_PDU is depicted in Figure 2-1.

Bits 0–15	Bits 16–23	Bits 24–27	Bits 28–31
Length_of_PDU	Priority	MAP	PDU_Type
Total_Number_of_PDUs	Checksum		
Source_ID			
Message_ID (MSID)			
Expiry_Time			
Count_of_Destination_Entries	Length_of_Reserved_Field		
List of Destination_Entries (variable length)			

One Destination_Entry

Bits 0–31
Destination_ID
Message_Sequence_Number
Reserved_Field (variable length)

Figure 2-1: Structure of an Address_PDU

b. Description of fields:

Length_of_PDU:

The field (2 octets) indicates the total number of octets within this PDU.

Priority: This 1-octet field contains the priority of the message. Priority is defined by the application. A value of 0 denotes the highest priority.

MAP: This 2-bit field specifies whether this Address_PDU is first, middle or last.

The high order bit is set to
'0: This is the first one of a set of Address_PDUs.
'1: This is NOT the first one of a set of Address_PDUs.

The low order bit is set to
'0: This is the last one of a set of Address_PDUs.
'1: This is NOT the last one of a set of Address_PDUs.

When both bits are set to '0, it means there is only on Address_PDU.

PDU_Type: This 6-bit field specifies the type of the actual PDU. PDU_Type x '02 denotes an Address_PDU.

Total_Number_of_PDUs:

These 2 octets hold the total number of Data_PDUs of the message.

Checksum: The checksum is calculated over the entire PDU except the checksum field using the Fletcher algorithm as in Annex B04.

Source_ID: This field holds the ID of the sender of this Address_PDU. This ID must be unique for the actual multicast network (eg the Internet address of the specific multicast interface of the transmitting node).

Message_ID: MSID is a unique identifier created within the scope of Source_ID by the transmitter (eg it may be derived from the seconds since 00:00:00 1.1.1970 – Unix Time).

Expiry_Time: This is the expiry time of the message. This value is set as the number of seconds since 00:00:00 1.1.1970 – Unix Time. This time is requested by the application (eg in the case of X.400 from the Latest_Delivery_Time of the X.400 message). This entry allows transmitters and receivers of a message to decide when a message entry is outdated and can be removed from a processing list.

Count_of_Destination_Entries:

These 2 octets specify the length of the Reserved_Field, which is 0 if the field is not used.

List of Destination_Entries:

This field is an array of destination entries. Its dimension is specified by count_of_Destination_Entries.

Destination_ID: This field holds a unique identifier identifying a receiving node on the actual multicast network (eg the Internet address of the receiving node).

Message_Sequence_Number:

This entry holds a message sequence number, which is unique for the sender/receiver pair denoted by Source_ID and Destination_ID. This sequence number is generated by the transmitter consecutively with no omissions and is used by receivers to detect message loss.

Reserved Field: This field is provided for future expansion.

204 Data_PDU

a. The P_MUL transmitter generates the Data_PDU to pass each of the message fragments to the intended recipients. This PDU holds the unique identifier of the message, the position of this Data_PDU within the ordered set of all Data_PDUs and a part of the total message. The structure of the Data_PDU is depicted in Figure 2-2.

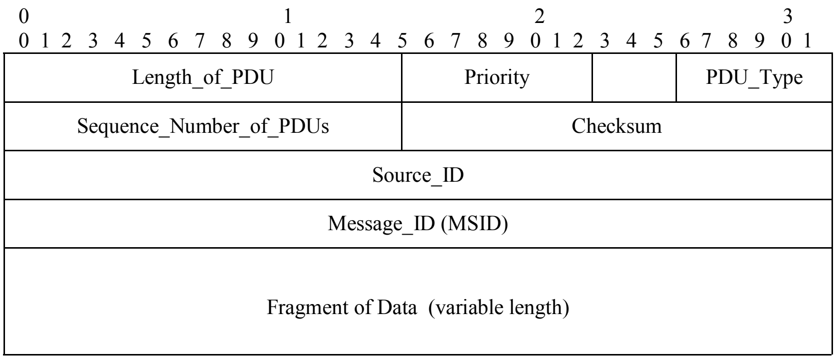


Figure 2-2: Structure of a Data_PDU

b. Description of fields:

Length_of_PDU:

The field (2 octets) indicates the total number of octets within this PDU.

Priority: This 1-octet field contains the priority of the message. Priority is defined by the application. A value of 0 denotes the highest priority.

PDU_Type: This 6-bit field specifies the type of the actual PDU. PDU_Type x '00 denotes a Data_PDU.

Sequence_Number_of_PDU:

this value specifies the order of the message fragment within the original message, starting from 1.

Checksum: The checksum is calculated over the entire PDU except the checksum field using the Fletcher algorithm as in Annex B04.

Source_ID: This field holds the ID of the sender of this Data_PDU and is equivalent to the Source_ID within the corresponding Address_PDU.

This ID must be unique for the actual multicast network (eg the Internet address of the specific multicast interface of the transmitting node).

Message_ID: MSID is a unique identifier created within the scope of Source_ID by the transmitter (eg it may be derived from the seconds since 00:00:00 1.1.1970 – Unix Time).

Fragment of Data:

This field of the Data_PDU holds the message or a fragment of the message.

205 Discard_Message_PDU

a. The Discard_PDU generated by a P_MUL transmitter is used to inform the receiving nodes that the transfer of a specific message has been terminated and no further PDUs of this message will be transmitted. Such situations can arise in the event of hardware error or message obsolescence. PDUs already received are to be discarded by the receiving node. Under these circumstances the transmitting node shall generate a Non-Delivery Report. The structure of the Discard_Message_PDU is depicted in Figure 2-3.

Bits 0–15	Bits 16–23	Bits 24–26	Bits 27–31
Length_of_PDU	Priority		PDU_Type
Unused	Checksum		
Source_ID			
Message_ID (MSID)			

Figure 2-3: Structure of a Discard_Message_PDU

b. Description of fields:

Length_of_PDU:

The field (2 octets) indicates the total number of octets within this PDU.

Priority:	This 1-octet field contains the priority of the message. Priority is defined by the application. A value of 0 denotes the highest priority.
PDU_Type:	This 6-bit field specifies the type of the actual PDU. PDU_Type x '03 denotes a Discard_Message_PDU.
Checksum:	The checksum is calculated over the entire PDU except the checksum field using the Fletcher algorithm as in Annex B04.
Source_ID:	This field holds the ID of the sender of the Discard_Message_PDU and is equivalent to the Source_ID within the corresponding Address_PDU. This ID must be unique for the actual multicast network (eg the Internet address of the specific multicast interface of the transmitting node).
Message_ID:	MSID is a unique identifier created within the scope of Source_ID by the transmitter (eg it may be derived from the seconds since 00:00:00 1.1.1970 – Unix Time).

206 Ack_PDU

a. This PDU is generated by a receiving node identified by the Source_ID_of_Ack_Sender and is used to inform the transmitting node of the status of one or more messages received. This information is composed as one or more entries of the list of Ack_Info_Entries. Each of these entries holds a message identifier (Source_ID and Message_ID) and a list of Missing_Data_PDU_Seq_Numbers, which may contain a list of those Data_PDUs not yet received. If this list is empty, the message identified by Source ID and Message ID has been correctly received. Figure 2-4 depicts the structure of an Ack_PDU.

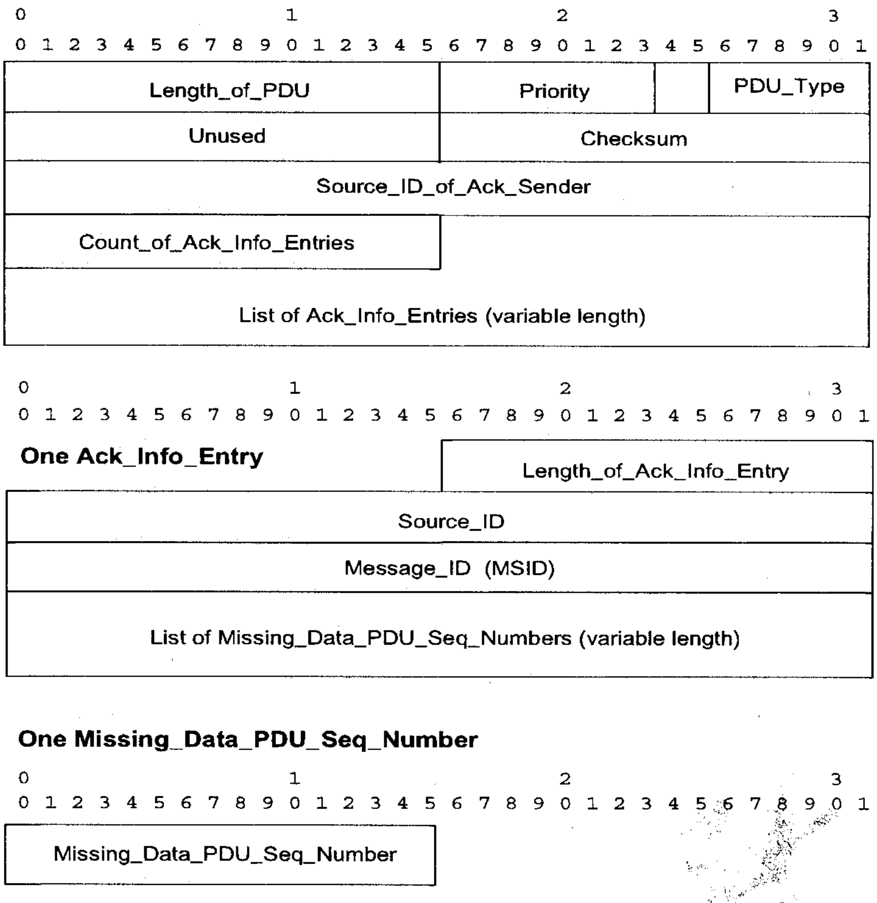


Figure 2-4: Structure of an Ack_PDU

b. Description of fields:

Length_of_PDU:

The field (2 octets) indicates the total number of octets within this PDU.

Priority: This 1-octet field contains the priority of the message. Priority is defined by the application. A value of 0 denotes the highest priority.

PDU_Type: This 6-bit field specifies the type of the actual PDU. PDU_Type x '01 denotes an Ack_PDU.

Checksum: The checksum is calculated over the entire PDU except the checksum field using the Fletcher algorithm as in Annex B04. Source_ID_of_Ack_Sender:

This field holds the ID of the sender of the Ack_PDU and is equivalent to one of the Destination_IDs described for the Address_PDU. This ID must be unique for the actual multicast network (eg the Internet address of the specific multicast interface of the Ack_PDU transmitting node).

Count_of_Ack_Info_Entries:

This field contains the number of Ack_Info_Entries. If there is only one active sender, there will be only one Ack_Info_Entry, although there may be more than one transmitting node on a given multicast network.

List of Ack_Info_Entries:

This entry is an array of Ack_Info_Entries for one or more transmitting nodes of the multicast network. Its dimension is specified by count_of_Ack_Info_Entries.

Ack_Info_Entry: This field contains the Source_ID of the transmitting node, the Message_ID and a variable length list of the sequence numbers of the Data_PDUs that have not yet been received.

Length_of_Ack_Info_Entry:

This field holds the octet length of each Ack_Info_Entry. It equals 2N+10 where N is the actual number of entries in the list of Missing_Data_PDU_Seq_Numbers of an Ack_Info_Entry. When it equals to 10, the Ack_Info_Entry acknowledges a complete message. Its maximum value is 2MM+10 for an intermediate-list, and 2MM+12 for an end-list (see \list of Missing_Data_PDU_Seq_Numbers for definitions), where MM is a receiver-configurable parameter (see para 322a).

Source_ID: This field holds the Identifier of the transmitting node. Its value is equivalent to the value "Source_ID" of the corresponding Address_PDU.

Message_ID: MSID is a unique identifier created by the sender (eg it may be derived from the seconds since 00:00:00 1.1.1970 – Unix Time). Its value is equivalent to the value of Message_ID of the corresponding Address_PDU.

List of Missing_Data_PDU_Seq_Numbers:

This field contains the list of sequence numbers of those Data_PDUs expected by the receiving node but found to be lost during the transmission.

it can be either an intermediate-list or an end-list in terms of a message transmission or re-transmission. An intermediate-list consists of MM entries and may be used only when a message has more than MM missing Data_PDUs. An intermediate-list is transmitted as soon as a receiving node realises that MM Data_PDUs are lost in a message, before reaching the end of the message transmission or re-transmission. An end-list is transmitted after receiving the last Data_PDU in a message transmission, or the highest numbered Data_PDU expected by the receiver in a message re-transmission. An end-list contains a variable number of entries, up to MM+1. A special form of an end-list is the empty list, known as the message completion acknowledgement, to report that all Data_PDUs in the message has been received correctly.

In an intermediate-list, the sequence numbers, except zeros (see below), are listed in a monotonically increasing order. In an end-list, the sequence numbers are also listed in a monotonically increasing order, until the highest number is reached, which is then followed by the (repeating) lowest missing Data_PDU number to mark the end. In this case if the list has less than MM entries, and if repeated acknowledgement is required, other previously listed (repeating numbers should also be concatenated, from either end. If repeated acknowledgement is required, then across the lists the numbers are listed in piecewise monotonically increasing order, except at the "boundaries" where the highest number meets the lowest one. The transmitting node interprets that any Data|_PDU(s) between two adjacent ones in the lists are received correctly by this receiving node. modulo the total Data_PDU number in a message, the list contains the lost sequence numbers twice, since it is both the highest and the lowest number. For efficiency, a zero may be used to represent a block of consecutive numbers between two listed numbers if no confusion is caused (eg 2, 3, 4, 5 may be represented by 2, 0, 5).

c. Normally one Ack_PDU reports the receiving status of one MSID from one Source_ID only. However, one Ack_PDU may contain the receiving status of more than one MSID from more than one Source_ID, especially at the end of an EMCON period.

207 PDUs to Dynamically Create Multicast Groups

a. When a set of nodes intends to deploy multicast message transfers, all nodes have to be connected to a multicast-supporting network. This set consists of a subject "T_Nodes" of potentially transmitting and a subset "R_Nodes" of potentially receiving nodes. the two subsets normally have a non-empty intersection.

b. The communication for this group management scheme is based on a globally known multicast address "GG", which all members of T_Nodes and R_Nodes join. GG is a multicast address used for group management purposes.

c. To allow different communication channels for T_Nodes and R_Nodes it is assumed that there are two well-known ports "TPORT" and "RPORT". all transmitting nodes within T_Nodes read PDUs addressed to GG and TPORT, while all receiving nodes within R Nodes read PDUs addressed to GG and RPORT. (Note: The data transmission itself is carried via a dynamically installed multicast group and two additional ports "DPORT" (Data_Port) and "APORT" (Acknowledgement_Port)).

d. For dynamic installation of a multicast group the following 4 PDUs are defined:

- Requesting a Multicast Group (Request_PDU)
- Rejecting a Multicast Group (Request_PDU)
- Releasing a Multicast Group (Release_PDU)
- Announcing a Multicast Group (Announce_PDU)

e. The first three PDU types are used only for communications between the transmitter within a transmitting node and all other members of T_Nodes being addressed by the global group GG with TPORT. The fourth PDU type is issued by a node of T_Nodes, to install a new multicast group and to inform all the affected receiving nodes to join the new group. This PDU is sent to all nodes within the set R_Nodes, being addressed by the global group GG with port RPORT. This PDU is used only after it has been agreed upon among the members of

T_Nodes, and the transmitting node is deemed to own that particular multicast group. After the completion of message transfer, the owner of the multicast group must release the multicast group.

208 Request_PDU, Reject_PDU and Release_PDU

a. These three PDU types (Request_PDU, Reject_PDU, Release_PDU) have the same simple structure. They contain only a few information fields and differ only with respect to a field "PDU_Type". The general structure of these PDUs is depicted in Figure 2-5.

Bits 0–15	Bits 16–31		
Length_of_PDU	Unused		PDU_Type
Unused	Checksum		
Source_ID			
Message_ID (MSID)			
Multicast_Group_Address			

Figure 2-5: General Structure of Request_PDU, Reject_PDU and Release_PDU

b. Description of fields

Length_of_PDU: The field (2 octets) indicates the total number of octets within the PDU.

PDU_Type: This 6-bit field specifies the type of the actual PDU. PDU_Type x '05 for a Request_PDU, or x '06 for a Reject_PDU or x '07 for a Release_PDU.

Checksum: The checksum is calculated over the entire PDU except the checksum field using the Fletcher algorithm as in Annex B04.

Source_ID: This field holds the ID of the sender of the PDU. This ID must be unique for the actual multicast network (eg the Internet address of the specific multicast interface of the sending MTA node).

Message_ID: MSID is a unique identifier created by the transmitter within the scope of Source_ID (eg it may be derived from the seconds since 00:00:00 1.1.1970 – Unix Time).

Multicast_Group_Address:

This field holds the address of a multicast group, to be requested, rejected, or released.

c. Request_PDU: The Request_PDU is distributed by a node wishing to initiate a new multicast group within T_Nodes, using group GG and port TPORT.

d. Reject_PDU: The Reject_PDU is used by a member of T_Nodes which already "owns" the multicast group. The Reject_PDU is sent to the requesting node in unicast mode.

e. Release_PDU: The Release_PDU is used in the following two situations:

(1) after the sender has received a Reject_PDU,
(2) after a transmission has finished.

This PDU type is used to inform those members of T_Nodes, which senders have relinquished particular multicast addresses.

209 Announcement of a Multicast Group using the Announce_PDU

a. The Announce_PDU sends a list of nodes, which are to receive a specific message to the nodes of R_Nodes. As P_MUL has to observe a maximum PDU size and as the number of Destination_IDs has no maximum limit, it is essential that the total list of Destination_IDs be able to be split more than one Announce_PDU. To distinguish between the first, middle, or last Announce_PDU, the MAP field ("More Address_PDUs") is used. The structure of the Announce_PDU is depicted in Figure 2-6.

0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

Length_of_PDU	Unused	MAP	PDU_Type
Count_of_Destination_IDs	Checksum		
Source_ID			
Message_ID (MSID)			

Expiry_Time
Multicast_Group_Address
List of Destination_IDs (variable length)

One Destination_Entry

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
```

Destination_ID

Figure 2-6: Structure of an Announce_PDU

b. Description of fields:

Length_of_PDU: The field (2 octets) indicates the total number of octets within this PDU.

MAP: This 2-bit field specifies whether this Announce_PDU is first, middle or last.

The high order bit is set to
'0: this is the first one of a set of Announce_PDUs.
'1: This is NOT the first one of a set of Announce_PDUs.

The low order bit is set to
'0: this is the last one of a set of Announce_PDUs.
'1: This is NOT the last one of a set of Announce_PDUs.

When both bits are set to '0, it means there is only one Announce_PDU.

PDU_Type: This 6-bit field specifies the type of the actual PDU. PDU_Type x '04 denotes an Announce_PDU.

Count_of_Destination_IDs:

These 2 octets hold the total number of Destination_IDs within the list of Destination IDs.

Checksum: The checksum is calculated over the entire PDU except the checksum field using the Fletcher algorithm as in Annex B04.

Source_ID: This field holds the ID of the sender of this Announce_PDU This ID must be unique for the actual multicast network (eg the Internet address of the specific multicast interface of the sending MTA node).

Message_ID: MSID is a unique identifier created within the scope of Source_ID by the transmitter (eg it may be derived from the seconds since 00:00:00 1.1.1970 – Unix Time).

Expiry_Time: This is the expiry time of the message. This value is set as the number of seconds since 00:00:00 1.1.1970 – Unix Time.

This time is requested by the application (eg in the case of X.400 from the Latest_Delivery_Time of the X.400 message). This entry allows transmitters and receivers of a message to decide when a message entry is outdated and can be removed from a processing list.

Multicast_Group_Address:

This 4-octet field holds the address of the multicast group announced for a message transfer denoted by Source_ID and MSID.

List of Destination_IDs:

This variable length field contains a list of Destination_IDs of the intended receiving nodes for the message denoted by source_ID and MSID. A Destination_ID may hold the Internet address of the receiving node.

CHAPTER 3

MESSAGING PROCEDURES

301 General

This chapter describes the messaging procedures employed between a transmitting node and one or more receiving nodes either in the non-EMCON or EMCON mode of operation, using the P_MUL multicast protocol. These procedures are implemented for every message transmitted. The procedures are divided into three main areas, namely:

- Transmission and re-transmission (see paras 302-309)
- Reception (see paras 310-316)
- Acknowledgement (see paras 317-327)

A set of state diagrams depicting these procedures is provided for information at Annex C.

302 Transmission and Re-transmission of a Message

a. A node shall initiate the transmission of a message by transmitting an Address_PDU containing a list of nodes that are to receive the message. The transmitting node shall be notified by a "Higher Management Function" about the operation mode of each receiving node, that is, which receiving nodes are in the EMCON mode, and which are in the non-EMCON mode. Based on this information, the transmitting node shall create a list of non-EMCON receving nodes, from which it will expect to receive Ack_PDUs.

b. After transmitting the Address_PDU, the transmitting node will transmit the Data_PDU(s), with the Message_ID field set equal to the Message_ID field of the associated Address_PDU. After transmitting the last Data_PDU of a message, the transmitting node shall initialise a "Transmitter Expiry_Time Timer". If one or more of the receiving nodes is in the EMCON mode, the transmitting node shall initialise an "EMCON Re-transmission counter" (EMCON_RTC). The transmitting node shall then enter the non-EMCON or EMCON re-transmission mode (see paras 305 and 308) according to the higher management function.

303 Transmitter Expiry_Time Timer

a. This timer indicates the time remaining before the contents of the transmitted messages are considered invalid. It is initialised in accordance with the value transmitted in the Expiry_Time field of the Address_PDU.

b. If one or more of the receiving nodes have not acknowledge the receipt of the complete message when this timer expires the transmitting node will transmit a Discard_Message_PDU with the Message_ID set equal to the Message_ID field in the associated Address_PDU and Data_PDUs.

c. If, after transmitting a Discard_Message_PDU, the transmitting node receives an Ack_PDU indicating partial reception of the message, the transmitting node shall re-transmit the Discard-Message_PDU.

d. If, after having transmitted a Discard_Message-PDU, the transmitting node received an Ack_PDU indicating receipt of the complete message, the transmitting node shall disregard the Ack_PDU.

c. If all the receiving nodes addressed in the list of Destination_Entries of the Address_PDU acknowledge receipt of the complete message before this timer expires, the timer shall be stopped.

304 EMCON Re-transmission Counter

This counter (EMCON_RTC) indicates the maximum number of times the complete message may be re-transmitted while in the EMCON Re-transmission mode.

305 Non-EMCON Re-transmission

a. The transmitting node may enter this mode when one or more of the receiving nodes are in the non-EMCON mode. On entering this mode the transmitting node shall either:

(1) having transmitted the last Data_PDU, initialise the "Ack Re-transmission Timer", or

(2) having left the "EMCON Re-transmission" mode, re-transmit all the indicated missing Data_PDUs, and then initialise the "Ack Re-transmission Timer".

b. Every non-EMCON re-transmission session is started with a transmission of an updated Address_PDU, listing only those receiving nodes that still need to receive the consequent re-transmitted data.

306 Ack Re-transmission Timer

a. This timer (ACK_RE-TRANSMISSION_TIME) shall be initialised when one or more of the receiving nodes are in the non-EMCON mode.

b. If the transmitting node received Ack-PDUs from all of those receiving nodes in non-EMCON mode before this timer expires, the transmitting node shall either:

(1) if one or more of the Ack_PDUs indicates that some receiving nodes have missing Data_PDUs, re-transmit all the indicated missing Data_PDUs accompanied by a updated Address_PDU, and re-initialise the timer and remain in the "Non-EMCON Re-transmission" mode, or

(2) if all the Ack_PDUs indicate that the complete message has been received, stop the timer. If at this stage there are receiving nodes in the EMCON mode, the transmitting node shall enter the EMCON Re-transmission mode.

c. If the transmitting node has not received all expected Ack_PDU(s) from one or more of the non-EMCON receiving node(s) and this timer expires, the transmitting node shall update and transmit the Address_PDU, and then either:

(1) if it has not received previous Ack_PDU(s) from one or more receiving node, it shall re-transmit the complete message and reset the timer to a value greater than the initial value by a configurable BACK-OFF_FACTOR, or

(2) if it has received previous Ack_PDU(s) from some receiving nodes, I shall only re-transmit those unacknowledged Data_PDUs, and reset the timer to a value greater than the initial value bya configurable BACK-OFF_FACTOR.

d. On each subsequent occasion when the timer expires, it shall be reset to a value greater than the previous value by a configurable BACK-OFF_FACTOR.

307 Receipt of Message complete Ack_PDU

a. When a transmitting node receives an Ack_PDU from a non-EMCON or an EMCON receiving node (after leaving EMCON) indicating that it has received the complete message, the transmitting node shall acknowledge this Ack_PDU by sending a modified Address_PDU. That is, the destination address of the node, which has acknowledged complete reception of the message, will be removed from the list of Destination_Entries of any subsequent re-transmission of this particular message.

b. After all receiving nodes have forwarded acknowledgements, an Address_PDU containing an empty list of Destination_Entries is sent out. If the particular multicast group was one that was dynamically created, the transmission node shall transmit a Release_PDU to all nodes of that multicast group to release the multicast group.

308 EMCON Re-transmission

The transmitting node may enter EMCON Re-transmission mode when any receiving nodes are in the EMCON mode. Having transmitted the last Data-PDU, the transmitting node shall initialise the "EMCON Re-transmission Timer" (EMCON_RTI).

309 EMCON Re-transmission Timer

a. If this timer expires, implying that since the last EMCON re-transmission all the receiving nodes in the EMCON mode have not entered the non-EMCON mode (by sending Ack_PDUs), and the "EMCON Re-transmission Counter" has not exceeded its maximum, the transmitting node shall re-transmit the Address_PDU and all Data_PDUs not already acknowledged. The transmitting node shall re-initialise this timer and increment the "EMCON Re-transmission Counter". The transmitting node shall wait until either:

(1) all the receiving nodes in EMCON mode respond with an Ack_PDU at which point the transmitting node shall enter the "Non-EMCON Re-transmission" mode, or

(2) the "Transmitter Expiry_Time timer" expires at which point the transmitting node shall transmit a Discard-Message_PDU.

b. If all of the receiving nodes in the EMCON mode respond (entering non-EMCON mode) with an Ack-PDU indicating partial or complete reception of the message before this timer expires, the transmitting node shall stop the timer and enter the "Non-EMCON Re-transmission" mode (see para 305).

310 Reception of a Message

A receiving node shall enter the "Reception of a Message" mode when ot receives either an Address_PDU or a Data_PDU.

311 Receipt of an Address_PDU

a. On receipt of an Address_PDU the reeiving node shall first check whether the Address_PDU with the same tuple "Source_ID, MSID" has already been received.

b. If such an Address_PDU has already been received the receiving node shall check whether it has previously sent a message complete Ack_PDU for this message.

(1) If it has sent a message complete Ack_PDU, then it shall either:

(a) if its own ID is not in the list of Destination_Entries, it knows that its own Ack_PDU has been successfully received by the transmitting node. Consequently, the receiving node can release all information about this message and its membership of the dynamically created multicast group, and can then discard the Address_PDU, or

(b) if its own ID exists in the list of Destination_Entries, re-transmit the message complete Ack_PDU, and discard the Address_PDU, or

(2) if it has not sent a message complete, Ack_PDU, discard the Address_PDU and wait for remaining Data_PDUs.

c. If the Address_PDU has not been previously received the receiving node shall either:

(1) if its own ID is not in the list of Destination_Entries, the receiving node shall check whether it has previously received any Data_PDUs associated with this Address_PDU (ie same Source_ID and Message_ID) and then discard the Address_PDU. If there are Data_PDUs associated with this Address_PDU, the receiving node shall discard these Data_PDUs, or

(2) if its own ID is in the list of Destination_Entries, determine whether it has previously received any Data_PDUs associated with this Address_PDU.

(a) If there are no Data_PDUs associated with this Address_PDU, the receiving node shall create a message entry and wait transmission of associated Data-PDUs.

(b) If there are Data_PDUs associated with this Address_PDU, the receiving node shall update the status of the Data_PDU entry (see para 313) to a message entry. The receiving node shall stop the "Unidentified_Data_PDU_Validity_Timer" (see para 314), and initialise the "Receiver Expiry_Time Timer". The receiving node shall determine whether to enter the "Acknowledgement of a Message" mode (see para 315).

312 Receiver Expiry_Time Timer

a. This timer indicates the time remaining before the contents of the received message are considered invalid. It is initialised in accordance with the value received in the Expiry_Time field of the Address_PDU.

b. If this timer expires before the receiving node has received all the Data_PDUs associated with a message, the receiving node shall discard the associated Data_PDUs and Address_PDU.

313 Receipt of a Data_PDU

a. On receipt of a Data_PDU the receiving node shall first check whether the Data_PDU has already been received.

b. If the Data_PDU has already been received the receiving node shall either:

(1) if the receiving node has not received a duplicate of the associated Address_PDU (see para 311) and it has previously sent a "message complete" Ack_PDU for this message, re-transmit the "message complete" Ack_PDU and discard the Data_PDU, or

(2) otherwise, discard the Data_PDU.

c. If the Data_PDU has not been previously received, the receiving node shall check whether it has received the associated Address_PDU.

(1) If the associated Address_PDU has been received and a message entry exists, the receiving node shall update the status of the message entry and determine whether to enter the "Acknowledgement of a Message" mode (see para 315). If the associated Address_PDU has been received but no message entry exists, the receiving node shall discard the Data_PDU.

(2) If the associated Address_PDU has not yet been received the receiving node shall check whether there is a Data_PDU entry associated with the Source_ID and Message_ID contents of the received Data_PDU. If there is no Data_PDU entry associated with this Data_PDU, the receiving node shall create a Data_PDU entry and await transmission of the associated Address_PDU. In addition, the receiving node shall initialise a "Unidentified_Data_PDUValidity_Timer". If there is a Data_PDU entry associated with this Data_PDU, the receiving node shall update the status of the Data_PDU entry and await transmission of the associated Address_PDU.

314 Unidentified_Data_PDU_Validity_Timer

a. This timer indicates the time remaining before the Data_PDUs in a Data_PDU entry are no longer considered valid and can therefore be discarded.

b. If the timer expires before the receiving node receives either the Address_PDU or a Discard_Message_PDU associated with the Data_PDUs in the Data_PDU entry, the receiving node shall discard all Data_PDUs.

315 Entry to "Acknowledgement of a Message"

Having updated the status of a message entry , the receiving nodes shall either:

(1) if in non-EMCON mode, check whether the last expired Data_PDU of the message has been received or whether there are MM (see para 206) missing Data_PDUs. In either case the receiving node shall enter the "Acknowledgement of a Message" mode (see para 318), otherwise the receiving node shall remain in the "Reception of a Message" mode, or

(2) if in EMCON mode, check whether all the Data_PDUs for the message have been received. If there are missing Data_PDUs the receiving node shall remain in the "Reception of a Message" mode. If all the Data_PDUs have been received, the message shall be tagged as complete and ready for acknowledgement when the non-EMCON mode is entered. The completed message can then be passed up to the "Higher Layer Application". The receiving node shall remain in the "Reception of a Message" mode.

316 Receipt of a Discard_Message_PDU

If a receiving node receives a Discard_Message_PDU, it shall discard all the PDUs (ie data and address) associated with the message identified by the combination of the Source_ID and Message_ID fields in the Discard_Message_PDU, and stop any associated timers. If a special multicast group has been dynamically created for this specific message, the receiving node shall release this group immediately.

317 Acknowledgement of a Message

a. The "Acknowledgement of a Message" mode can only be entered by a receiving node that is in the non-EMCON mode of operation. the "Acknowledgement of a Message" mode procedures shall be dependent on whether the receiving node received the messages in the non-EMCON mode (see paras 318-322), or in the EMCON mode (see paras 323-327).

b. To avoid the problem of acknowledgement implosion at the message-transmitting site, in the event that the receiving node has several Ack_PDUs to transmit, each transmission of an Ack_PDU is delayed by a randomly determined period of time.

318 Receiving Node in Non-EMCON

A receiving node, if operating in the non-EMCON mode, shall enter the "Acknowledgement of a Message" mode:

(1) when the last Data_PDU of a message is received for the first time, or

(2) when the expected highest numbered (previously missing) Data_PDU of a message is received, in a re-transmission session, or

(3) there are MM missing Data_PDUs (see para 206 and 322).

319 Last Data_PDU Received

In an initial transmission of a message, if the last Data_PDU of a message has been received, the receiving node shall determine whether there are any missing Data_PDUs in the message. In a re-transmission session, this happens once the highest numbered missing Data_PDU has been received.

320 Missing Data_PDUs

If there are missing Data_PDUs the receiving node shall either:

(1) if no Ack_PDU associated with this message has been transmitted, transmit an Ack_PDU listing which Data_PDUs are missing, and return to the "Reception of a Message" mode (refer para 310), or

(2) if an Ack_PDU associated with this message has been transmitted, return to the "Reception of a Message" (refer para 310) mode.

321 Received Complete Message

If there are no missing Data_PDUs, the receiving node shall transmit an Ack_PDU indicating that the message is complete, and stop the "Receiver Expiry_Time Timer" associated with this message. The complete message shall be passed up to the "Higher Layer Application". On completion of a transmission of the Ack_PDU the receiving node shall return to the "Reception of a Message" (refer para 310) mode.

322 Maximum Number of Missing Data_PDUs (MM)

a. MM is a receiver-configurable parameter defined as the maximum number of missing Data_PDUs a receiving node should acknowledge (negatively) at a time. MM is also the maximum number of Missing_Data_PDU_Seq_Numbers an Ack_Info_Entry of an Ack_PDU can carry in an intermediate-list (see para 206). However, in an end-list (at the end of a message transmission or re-transmission), the maximum number of entries in one Ack_Info_Entry is MM+1.

b. A message may consist of more than MM Data_PDUs. Therefore a receiving node may miss more than MM Data_PDUs before receiving the last Data_PDU of a message. A re-transmission session may also consist of more than MM Data_PDUs and a receiving node (in non-EMCON mode) may miss more than MM Data-PDUs and a receiving node (in non-EMCON mode) may miss more than MM of them before receiving the highest numbered Data_PDU as expected from this re-transmission session. the receiving node shall construct and transmit an Ack_Info_Entry of an Ack_PDU for every MM missing Data_PDUs that it misses, until the last Data_PDU of the message, or the highest numbered Data_PDU expected is received.

c. Once the expected last Data_PDU in the message has been received, the receiving node will transmit an Ack_PDU with an end-list, listing the remaining N missing Data_PDUs, where $0 < N \leq MM$, followed by the (previously reported) lowest missing Data_PDU sequence number. If the total number in the list N+1 is less than MM+1, then the receiving node may fill in with previously reported MM-N numbers as a repeating measure for better reliability.

d. If the receiving node has sent a number of intermediate lists and since then has found no further missing Data_PDU when the last expected Data_PDU of the message is received ($N = 0$), it should mark the end of the overall list by sending another Ack_PDU. This Ack_PDU contains a list of the highest and lowest missing Data_PDU numbers as previously reported. Again, this list can be filed with MM-1 repeating missing Data_PDU sequence numbers to make up MM+1 for better reliability.

e. When all of the Data_PDUs of a message have been received correctly, the receiving node will transmit a message completion Ack_PDU, with which the Ack_Info_Entry has an empty list of Missing_Data_PDU_Seq_Numbers, and the length _of_Ack_Info_Entry equals 10.

f. On completion of transmission of every Ack_PDU, the receiving node shall return to the "Reception of a Message" mode.

323 Receiving node leaving EMCON

a. As soon as a receiving node changes its operation mode from EMCON to non-EMCON, it shall enter the "Acknowledgement of a Message" mode. Upon entering the non-EMCON mode it shall determine whether it has received the last Data_PDU of a message.

b. Note: The following procedures are relevant only for those messages (complete or partial) received while in the EMCON mode. All new messages received before the change into the EMCON mode or after the change to the non-EMCON mode of operation are acknowledged in accordance with the procedures described in para 315.

324 Last Data_PDU Received

If the last Data_PDU of a message has been received, the receiving node shall determine whether there are any Data_PDUs missing from the message.

325 Missing Data_PDUs

a. If any Data_PDUs are missing, the receiving node shall transmit an Ack_PDU listing the Data_PDUs which are missing, and initialise an associated "Ack_PDU timer" of the open message (see para 327). If there are N missing Data_PDUs and $N > MM$, the receiving node shall transmit at least $\lceil N/MM \rceil$ Ack_PDUs in order to indicate the total number of missing Data_PDUs, where $\lceil x \rceil$ means the smallest integer no less than x. No Ack_PDU shall indicate more than MM new missing Data_PDUs (see para 322). The receiving node shall initialise the associated "Ack_PDU Timer" with each Ack_PDU transmitted.

b. On completion of transmission of the Ack_PDU(s) the receiving node shall then return to the "Reception of a Message" mode.

326 Received Complete Message

If there is no missing Data_PDU, the message is tagged as complete (see para 315(2)), and the receiving node shall transmit a message completion Ack-PDU (para 322e) with an empty list of Missing_Data_PDU_Seq_Numbers and initialise the "Ack_PDU Timer". the receiving node shall, if it has not already done so, stop the "Receiver_Expiry_Time Timer" associated with this message and pass the complete message to the "Higher Layer Application". the receiving node shall return to the "Reception of a Message" mode.

327 Ack_PDU Timer

a. There is one Ack_PDU Timer for each message that is not yet fully received. The "Ack_PDU Timer" shall be initialised for every Ack_PDU transmitted by a receiving node in the non-EMCON mode, having previously been in the EMCON mode.

b. If the receiving node receives a response to a transmitted Ack_PDU from the transmitting node in the form of the requested missing Data_PDUs or an Address_PDU, it shall stop the associated "Ack_PDU Timer".

c. If the receiving node does not receive a response to the transmitted Ack_PDU(s) from the transmitting node in the form of the requested missing Data_PDUs or an Address_PDU, and an "Ack_PDU Timer" expires, the receiving node shall re-transmit the associated Ack_PDU(s) and re-initialise the timer.

CHAPTER 4

MULTICAST GROUP FORMING PROCEDURES

401 General

a. Multicast group formation procedures can be statically or dynamically assigned. Statically assigned groups require an external authority to establish and manage these groups. P_MUL supports the dynamic formation of multicast groups, allowing groups to be established to support a single-message transmission.

b. The rest if this chapter deals with procedures for dynamic multicast group formation. Note that the implementation of dynamic multicasting group is an optional capability in this release of ACP 142.

c. Dynamic instantiation of multicast groups is a useful means of reducing the overall network load within a multicast network. Especially in cases when the sender of a message or file has precise knowledge of the addresses of the intended receiving nodes.

d. Joining and leaving of a multicast group will influence the distribution tree for the multicast group. This distribution tree is heavily dependent on the dialogue between neighbouring routers. Therefore the "dynamic multicast group forming procedure" can only be employed efficiently if none of the network links between the transmitting node and the intended receiving node is under EMCON condition at the time of group formation.

402 Request for a Multicast Group

a. First the transmitting node has to determine a multicast address for the multicast group which shall be dynamically installed. This multicast address can be chosen by different methods, including:

- selection from a set of multicast addresses, which is predefined and mutually exclusive for each member of T_Nodes,
- random choice from a predefined set of multicast addresses
- on behalf of MADCAP (Multicast Address Dynamic Client Allocation Protocol) from a multicast address allocation server [MADCAP99].

b. When a member of T_Nodes node receives a Request_PDU, it checks whether this multicast group is in use. If so, it will reject this group by sending a Reject_PDU, otherwise it does not respond. This is known as the "Silent Procedure" which aims to reduce the network load.

403 Rejecting a Multicast Group

a. If a transmitting node detects that another transmitting node is requesting a multicast address, which it is already using, it should reject this request by sending a Reject_PDU.

b. If the requesting node receives such a Reject_PDU it must revoke its request by the transmission of a corresponding Release_PDU, and has to renew its request with a different multicast address.

c. If the node detecting this collision is under EMCON condition, this rejection is impossible. Therefore, the use of the requested multicast group will result in some additional network load.

d. The same result as in c above will occur, if the particular transmitting node, which already owns the requested multicast group, does not receive the Request_PDU, or if the Reject_PDU it generates is not received by the requesting node.

e. If all members of T_Nodes keep track of the ownership of requested multicast addresses, then the rejection would rarely happen.

404 Release of a Multicast Group

A transmitting node will release a once-requested or used multicast address in the following two situations:

(1) after having received a Reject_PDU, or
(2) after completion of a message transmission.

405 Announcement to join a Multicast Group

a. After the transmission of the Request_PDU the requesting node waits a certain time (WAIT_FOR_REJECT_TIME) to receive Reject_PDUs from members of T_Nodes. If no Reject_PDU has been received when this time has expired, the requesting node announces the requested multicast address to all receiving nodes. Once a multicast group has been announced, the late reception of a corresponding Reject_PDU will be ignored.

b. This information is sent by the transmitting node via the Announce_PDU to the global group GG and port RPORT.

c. The Announce_PDU may be transmitted multiple times as it is essential for subsequent data transmission. the number of transmission times of Announce_PDU is specified by ANNOUNCE_CT.

d. As soon as a member of R_Nodes has received the Announce_PDU, it decides whether it is a member of the announced group, based on whether it is listed in the list of Destination_IDs. If it is not in the list if Destination_IDs it can ignore the Announce_PDU; otherwise it must join the multicast group denoted by the Announce_PDU.

e. The announcing node must wait a certain period of time (ANNOUNCE_DELAY) until all routers within the multicast tree acquire information about the group memberships of those nodes in the list of Destination_IDs. After this time has expired, data transfer may begin.

f. After this announcing phase the transmitting node assumes that the intended receiving nodes have received the Announce_PDU and therefore have joined the announced multicast group with Multicast_Group_Address. It will start the first data transmission phase by transmitting theAddress_PDU(s) and the Data_PDU(s). It will then wait for the Ack_PDU(s). Before the transmitting node restarts a re-transmission phase it has to check whether it can be sure that all intended receiving nodes had received the Announce_PDU. This decision can be made upon the fact, whether it received an Ack_PDU from each intended receiving node. If not the re-transmission of the eventually updated Announce_PDU according to the above rules concerning ANNOUNCE_DELAYand ANNOUNCE_CT followed by the next data transmission phase.

CHAPTER 5

REFERENCES

[CCITT88] CCITT, ISO: "Data Communication Networks: Message Handling Systems, Recommendations X.400-X.420", Blue Book, 1988

[FS1037] YS Federal Standard, FS1037C, "Telecommunications: Glossary of Telecommunication Terms", August 7, 1966

[ISO86] OSO: "International Standard 8602 – Information Processing Systems – OSI: Connectionless Transport Protocol Specification", December 1986

[MADCAP99] S R Hanna, B V Patel, and M Shah, IETF RFC 2730, "Multicast address dynamic client allocation protocol (MADCAP)", December, 1999

[SMTP82-1] J Postel, "Simple Mail Transfer Protocol", IEFT RFC 821, August 1982

[SMTP82-2] D Crocker, "Standard for the format of ARPA Internet text messages", RFC822, August 1982

[UDP80] J Postel: "User Datagram Protocol", IETF RFC768, August 1980

[XTP95] XTP Forum: "Xpress Transport Protocol Specification – XTP Revision 4.0", XTP Forum, 1394 Greenworth Place, Santa Barbara, CA 93108, March 1, 1995

ANNEX A

EXAMPLES

A01 Introduction

The following examples are designed to illustrate how the defined PDUs are used between the different nodes on a multicast network and to give an impression of how the protocol works. Protocol exchanges are described in a function call type notation.

A02 Example of Agreement about New Multicast Groups

a. This example describes how a new multicast group is agreed among transmitting nodes and announced to the receiving nodes.

b. It is assumed that M0 is the transmitting node, which sends one message to nodes M1, M2, M3 and M4. Only M3 and M4 are assumed to be under EMCON.

c. Firstly, M0 selects a new multicast address (eg 240.1.2.3) and transmits a corresponding Request_PDU:

Request_PDU (source_ID =M0, MSID = 9876,
Multicast_Group_Address = 240.1.2.3)

d. It is assumed that a node within the network already "owns this multicast address. this node will detect this collision and will send M0 (unicast mode) a corresponding Reject_PDU.

Reject_PDU (source_ID =M0, MSID = 9876,
Multicast_Group_Address = 240.1.2.3)

e. As soon as M0 receives this Reject_PDU, it releases its old request by transmitting a corresponding Release_PDU, and starts a new request with another multicast address (eg 240.1.2.4).

Release_PDU (source_ID =M0, MSID = 9876,
Multicast_Group_Address = 240.1.2.3)

Request_PDU (source_ID =M0, MSID = 9876,
Multicast_Group_Address = 240.1.2.3)

f. It is assumed that this multicast address is not already in use, or that the multicast address is owned by a node in EMCON. Therefore, after the timer WAIT_FOR_REJECT_TIME expires, and M0 has not received any correspondence Reject_PDUs, M0 transmits the following Announce_PDU:

Announce_PDU (source_ID =M0, MSID = 9876,
Expiry_Time = 1234567890,
Multicast_Group_Address = 240.1.2.4)
Count_of_Destination_IDs = 4
Destination_IDs = (M1, M2, M3, M4))

g. After the transmission of this Announce_PDU any further reception of a Reject_PDU will be discarded. M0 starts the timer ANNOUNCE_DELAY to enable the intended receiving nodes to join the announced multicast group and to enable the routers within the network to build the new multicast routing structure. To avoid packet loss of the Announce_PDU, the transmission of this packet will be repeated until ANNOUNCE_DELAY expires.

A03 Example of Data Transfer

a. This example illustrates how data transfer is coordinated.

b. Continuing the example in para A02, it is assumed that M0 is the transmitting node, which sends one message to the nodes M1, M2, M3 and M4 (Figure A-1). Only M3 and M4 are assumed to be under EMCON. The message length is assumed to be a little larger than the maximum PDU size, which means that the message (Data) is split into two fragments. It is also assumed that until now M0 has sent 99 messages to M1, 77 messages to M2, 10 messages to M3 and 14 messages to M4.

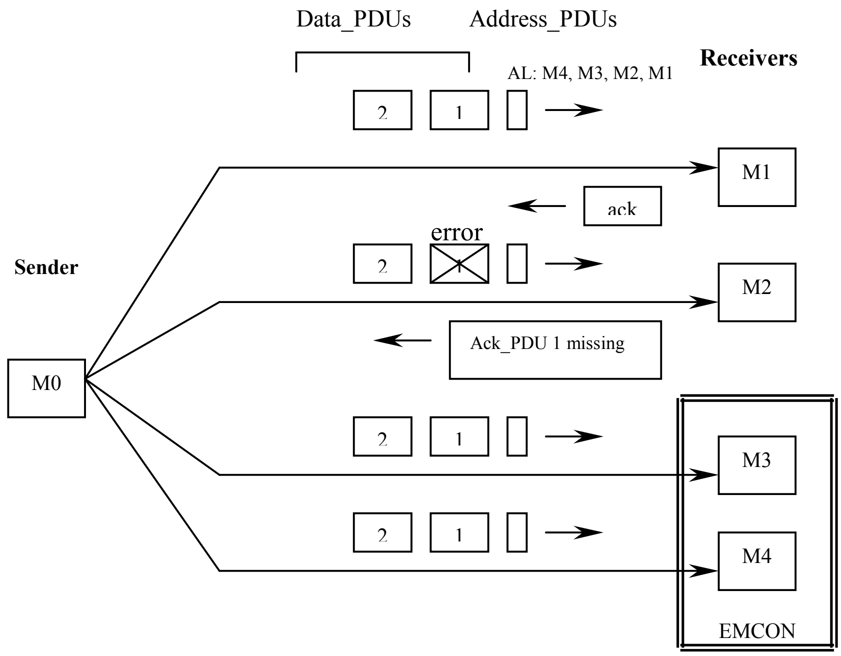


Figure A-1: M0 sends a message toM1 – M4 in multicast group, M1 and M2 acknowledge

c. M0 constructs an Address_PDU and the two Data_PDUs. It sends all three PDUs to the multicast network.

Address_PDU (source_ID =M0, MSID = 9876,
Expiry_Time = 1234567890, Total_Number_of_PDUs = 2,
Count_of_Destination_Entries = 4,
Length _of_Reserved_Field = 0,
Destination_Entries = ((M1, 100), (M2, 78), (M3, 11), (M4, 15)))

Data_PDU (Source_ID = M0, MSID = 9876,
Sequence_Number_of_PDU = 1, Data = First N octets of message)

Data_PDU (Source_ID = M0, MSID = 9876,
Sequence_Number_of_PDU = 2, Data = remaining octets of message)

d. Assuming M1, M2, M3 and M4 receive the Address_PDU, M1, M3 and M4 receive the Data_PDUs without failures, while M2 receives the first Data_PDU incorrectly and the second one correctly. As M3 and M4 are in EMCON mode, they cannot send any Ack_PDU. M1 sends a completion Ack_PDU, while M2 sends an Ack_PDU indicating that the first Data_PDU is missing, and that is all is missing (figure A-1).

Ack_PDU (Source _ID_of_Ack_Sender = M1,
Count_of_Ack_Info_Entries = 1.
Ack_Info_Entry = (Length_of_Ack_Info_Entry = 10,
Source_ID = M0, MSID = 9876))

Ack_PDU (Source _ID_of_Ack_Sender = M2,
Count_of_Ack_Info_Entries = 1.
Ack_Info_Entry = (Length_of_Ack_Info_Entry = 14,
Source_ID = M0, MSID = 9876,
Missing_Data_PDU_Seq_Number = 1,
Missing_Data_PDU_Seq_Number = 1)

e. M0 has to re-transmit the first part of the message for M2, because M2 marked this PDU as missing. As M0 has received the completion Ack_PDU of M1, M1 is deleted from the list of Destination_Entries.

Address_PDU (source_ID =M0, MSID = 9876,
Expiry_Time = 1234567890, Total_Number_of_PDUs = 2,
Count_of_Destination_Entries = 3,
Length _of_Reserved_Field = 0,
Destination_Entries = ((M2, 78), (M3, 11), (M3, 11), (M4, 15)))

Data_PDU (Source_ID = M0, MSID = 9876,
Sequence_Number_of_PDU = 1, Data = First N octets of message)

f. Assuming that M2 will receive this PDU correctly, M2 will acknowledge with the following message complete Ack_PDU (Figure A-2):

Ack_PDU (Source _ID_of_Ack_Sender = M2,
Count_of_Ack_Info_Entries = 1.
Ack_Info_Entry = (Length_of_Ack_Info_Entry = 10,
Source_ID = M0, MSID = 9876))

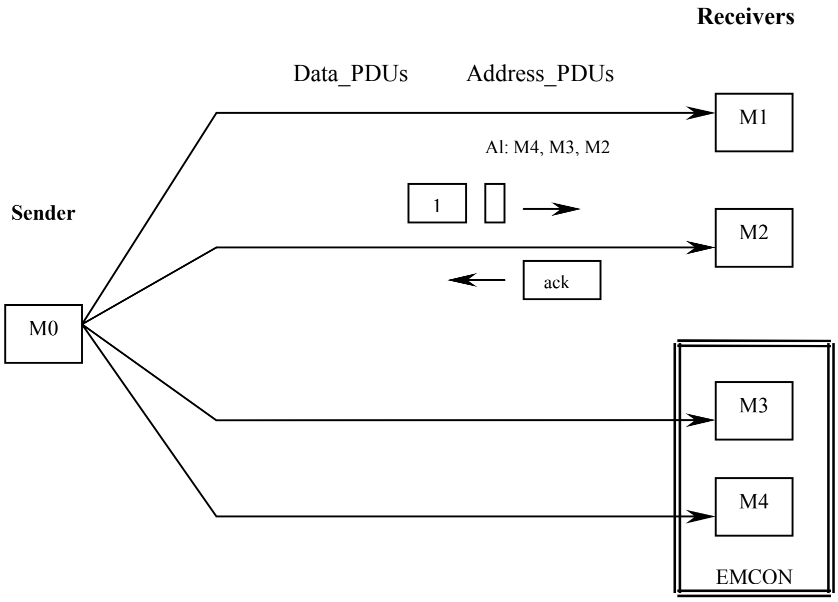


Figure A-2: Corrupted Data_PDU 1 resent to M2, M2 acknowledges receipt of message

g. As M0 has received the completion Ack_PDU of M2, M2 will be deleted from the list of Destination_Entries.

h. Supposing that M3 has left the EMCON situation (figure A-3), M3 will send its message complete Ack_PDU to M0 as:

Ack_PDU (Source _ID_of_Ack_Sender = M3,
Count_of_Ack_Info_Entries = 1.
Ack_Info_Entry = (Length_of_Ack_Info_Entry = 10,
Source_ID = M0, MSID = 9876))

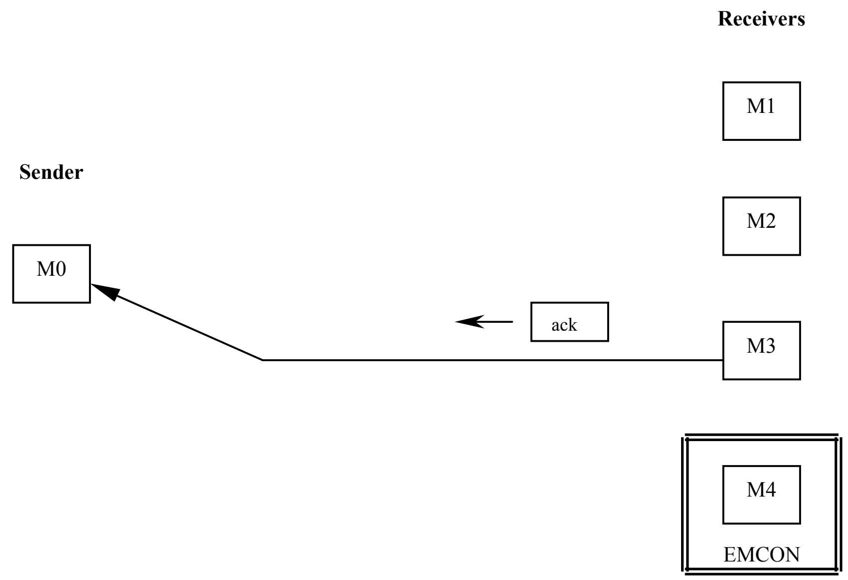


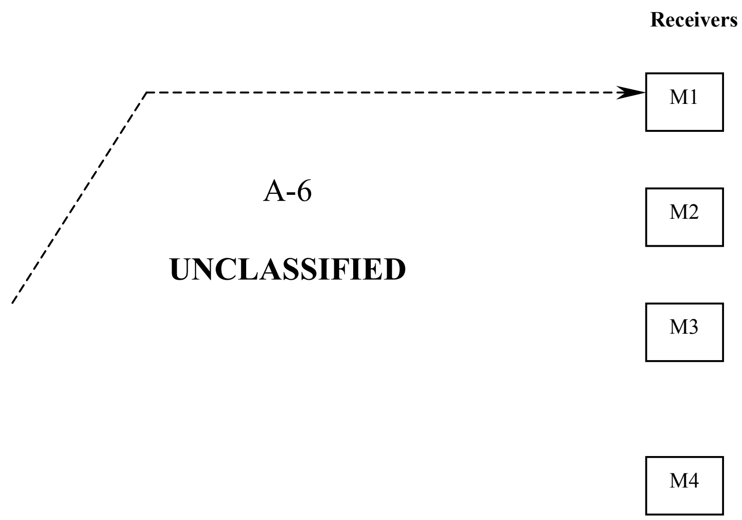
Figure A-3: M3 leaves EMCON and acknowledges correct receipt of message

i. Assuming that an EMCON Re-transmission for M4 takes place, M0 will re-transmit the total message. As M1, M2 and M3 already have completely acknowledged the message, the Address_PDU will hold only one Destination_Entry for M4 (Figure A-4).

Address_PDU (source_ID =M0, MSID = 9876,
Expiry_Time = 1234567890, Total_Number_of_PDUs = 2,
Count_of_Destination_Entries = 1,
Length _of_Reserved_Field = 0,
Destination_Entries = (M4, M15))

Data_PDU (Source_ID = M0, MSID = 9876,
Sequence_Number_of_PDU = 1, Data = First N octets of message)

Data_PDU (Source_ID = M0, MSID = 9876,
Sequence_Number_of_PDU = 2, Data = Remaining octets of message)



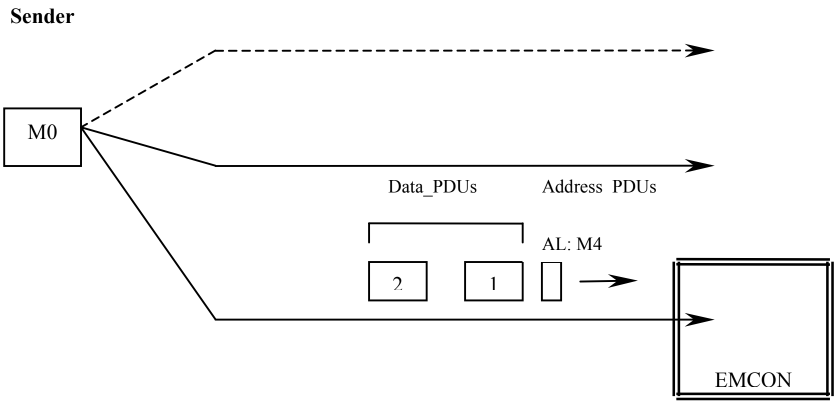
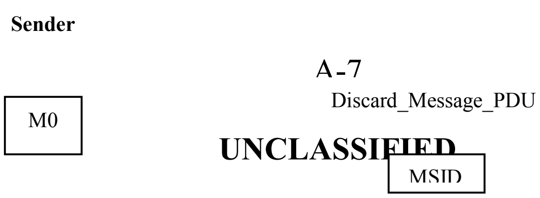


Figure A-4: Re-transmission of message to M4 occurs

j. As M4 is still under EMCON and the Expiry_Time for the message is exceeded (Figure A-5), M0 will send the following PDUs:

Discard_Message_PDU (Source_ID = M0, MSID = 9876)



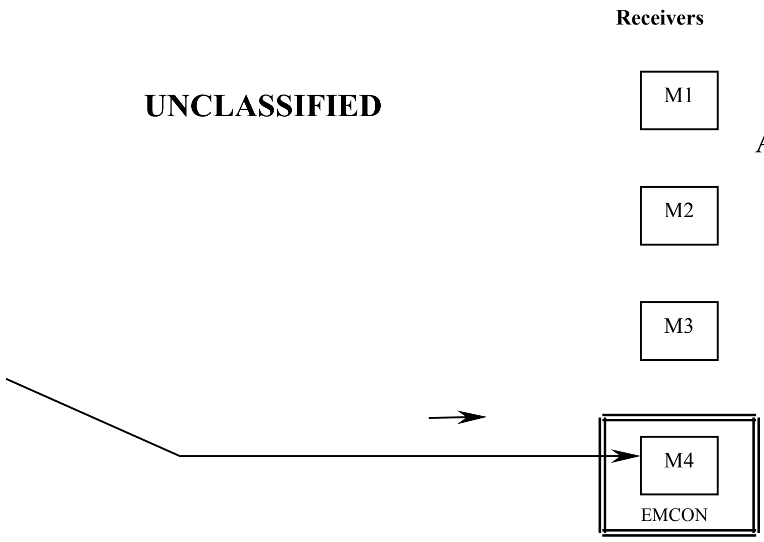
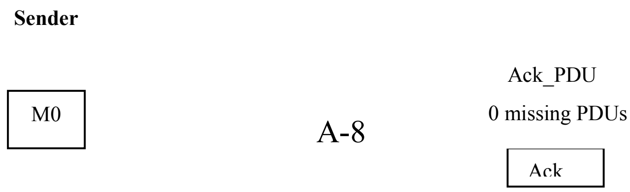


Figure A-5: M0 forwards a Discard_Message_PDU to M4

k. After a further period of time, M4 leaves EMCON and sends an Ack_PDU to M0 (Figure A-6). As M4 received the total message before the Expiry_time was reached, it will send the following Ack_PDU:

Ack_PDU (Source_ID_of_Ack_Sender = M4,
Count_of_Ack_Info_Entries = 1,
Ack_Info_Entry = (Length_of_Ack_Info_Entry = 10,
Source_ID = M0, MSID = 9876))



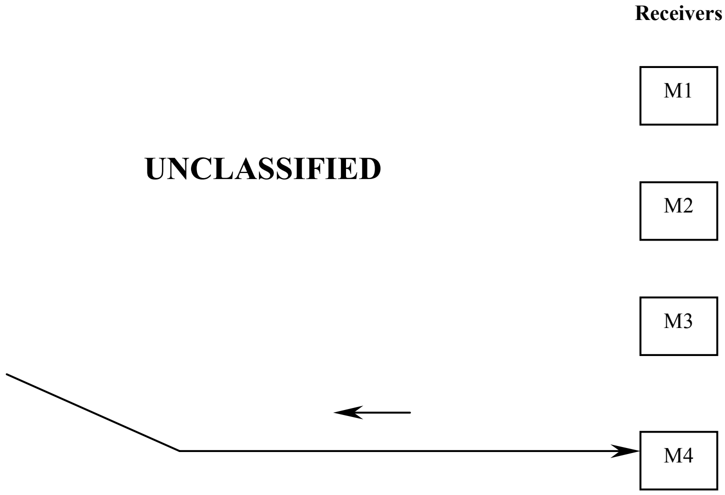


Figure A-6: leaves EMCON and acknowledges message receipt

l. At this time the Message Transmitter can hand over the message to Multicast_OUT to update the message queue entry. M0 will delete M4 from the list of Destination_Entries and will send out an Address_PDU with an empty list of Destination_Entries.

Address_PDU (source_ID =M0, MSID = 9876,
Expiry_Time = 1234567890, Total_Number_of_PDUs = 2,
Count_of_Destination_Entries = 0,
Length _of_Reserved_Field = 0, Destination_Entry = ( ))

to inform M1, M2, M3 and M4, that M0 received all Ack_PDUs from all receiving nodes (figure A-7). Consequently, all information about the message MSID = 9876 sent by M0 can be deleted.

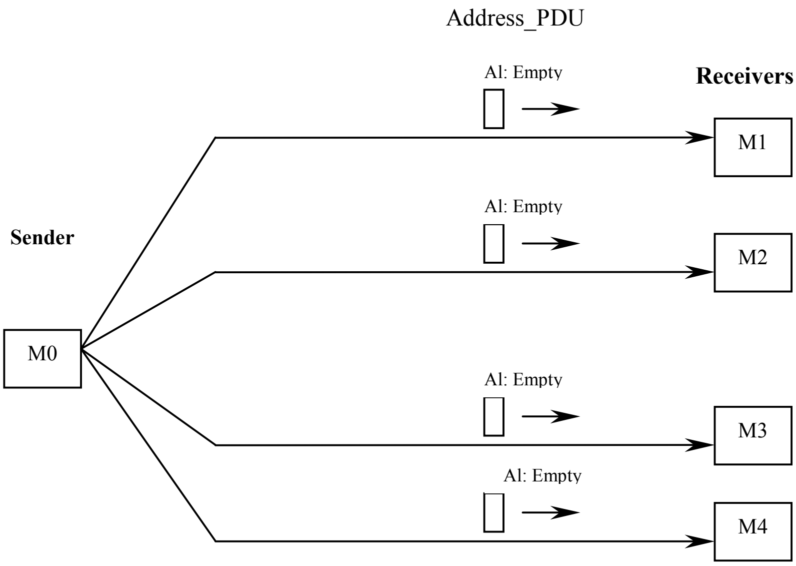


Figure A-7: M0 indicates that all nodes have received the message

m. In case the message has been sent to a multicast group that had been dynamically agreed upon, the transmitting node M0 will release this multicast group by sending a Release_PDU to all members of the multicast group (figure A-8):

Release_PDU (source_ID = ID = M0, MSID = 9876,
Multicast_Group_Address = 240.1.2.4)

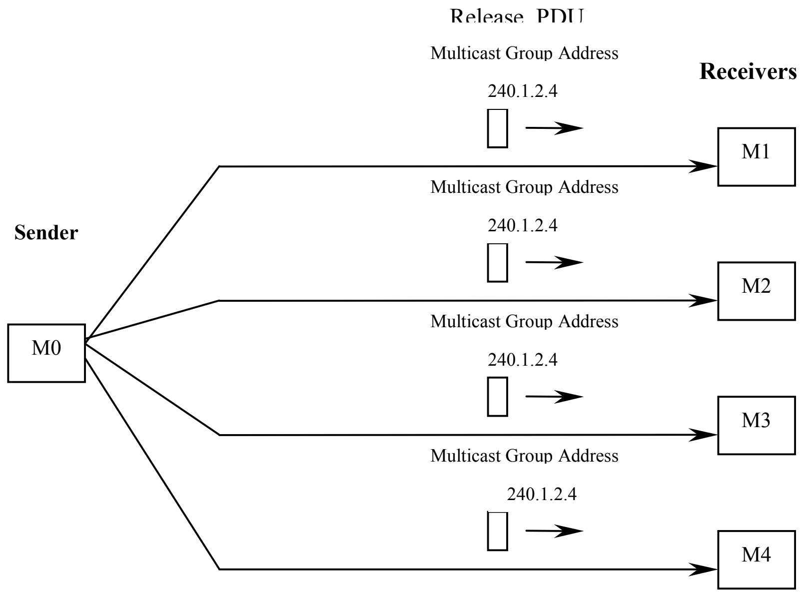


Figure A-8: Dynamically created multicast group is released by Release_PDU

ANNEX B

PARAMETERS AND ALGORITHM

B01 Predefined Protocol Parameters

WAIT_FOR_REJECT_TIME	Time between sending a Request_PDU and an affiliated Announce_PDU.
ANNOUNCE_DELAY	Time between sending an Announce_PDU and the first afiliated Address_PDU.
ANNOUNCE_CT	The number of times the Announce_PDU is transmitted,
ACK_RE-TRANSMISSION_TIME	Time a transmitter waits before re-transmitting a message to receivers not in EMCON, if no acknowledgement received.
BACK-OFF_FACTOR	Multiplying factor applied to ACK_RE-TRANSMISSION_TIME on subsequent re-transmissions, to achieve exponentially increasing delay [On n^{th} re-transmission, $delay_n$ = A_R_T_*(B_F)^n]
EMCON_RTC	Re-transmission count – Maximum number of message re-transmission for receivers in EMCON.
EMCON_RTI	Re-transmission interval – Time in seconds a transmitter waits before re-transmitting a message for receivers in EMCON.
MM	Maximum number of new entries in the list of Missing_Data_PDU_Seq_Numbers field in an Ack_Info_Entry of an Ack_PDU.
ACK_PDU_TIME	time a receiver waits before re-transmitting Ack_PDU(s) if no response is received from the transmitting node.

MULTICAST_GROUP_TABLE	For each multicast group the table lists the IP address of each member and the multicast IP address of the group.

B02 Recommended Multicast Group Address (for organisational purposes)

to allow the dynamic building of multicast groups it is necessary to have one multicast address to which all nodes have joined.

239.1.1.1 (GG)	This group address is used for group management purposes. The transmitting node is using this group management Request_PDUs, Release_PDUs and Announce_PDUs. All sending and receiving nodes have to join this multicast group.

B03 Recommended UDP Port Numbers

To allow multiplexed multicast communication between all nodes of a network it is necessary to define some UDP port numbers:

PORT 2751 (TPORT)	is used for the transmission of Request_PDUs, Reject_PDUs and Release_PDUs between the transmission programs. All transmitter processes have to listen to this port in conjunction with the multicast group GG.
PORT 2752 (RPORT)	is used by the transmitters to send the Announce_PDUs, informing those receivers involved in the concerning message transfer to join a specific multicast group. All receiver processes have to listen to this port in conjunction with the multicast group GG.
PORT 2753 (DPORT)	is used for the data traffic from the Message Transmitter of Multicast_OUT to the Message Receiver of Multicast_IN.
PORT 2754 (APORT)	is used for the traffic from the Message Receiver of Multicast_IN to the message Transmitter of Multicast_OUT.

B04 Checksum Algorithm

```
# include "includes.h"
```

```
/* function checksum
 * This function computes and sets the checksum (FLETCHER algorithm).
 * If "offset" is NULL, then the checksum will be calculated but the
 * checksum bytes will not be set.
 * If "offset" is on non-NULL, the checksum bytes at "offset" will be set
 * such that the checksum accumulations, when recalculated, will both
 * be zero.
 * return : checksum accumulations
 */
int checksum (buffer, len, offset)
  octet *buffer;
  int len, offset;
(
  unsigned int c0 c1;
  int cs;
  long ctmp, cl0, cl1;
  octet *hpp, *pls;

  if (offset) {
  buffer [offset] = 0;
  ctmp = len – offset – 1;
}

pls = buffer + len;
c0 = c1 = 0;
hpp = buffer;

while (hpp < pls) {
  if ((c0 + = *hpp++) > 254 { c0 - = 255; }
  if ((c1 + = c0) > 254) { c1 - = 255; }
} /* while */

if (offset) {
  cl0 = c0;
  cl1 = c1;
  if ((cs = (( ctmp * cl0) – cl1) % 255L) < 0) { cs + = 255; }
  buffer [offset] = cs;
  if ((cs = (cl1 = ((ctmp + 1L) * cl0)) % 255L) < 0) { cs + = 255;}
  buffer [offset + 1] = cs;
} /* if (offset) */
```

```
return (c0) | c1);
}
```

ANNEX C

STATE TRANSITION DIAGRAMS

C01 Introduction

The following state transition diagrams depict the messaging procedures. They are provided here for information and clarification. the states (bubbles) on each of the state transition diagrams are numbered to aid commenting on the diagrams.

C02 Notes on Multicast OUT State Transition Diagram

a. The Multicast OUT STD does not mention the Outbound Handler, Message Sent Store or Message Transmitter. the STD describes the events of the Multicast OUT channel as a single entity.

b. The "New message" event from bubbles 1 to 22 only occurs once for a message, this is when a message is submitted for transmission to a receving MTA.

c. Determination of which new message to process first is dependent on the Priority of the message.

d. The "Ack_PDU received" event from bubbles 1 to 2 indicates that an MTA not in EMCON has acknowledged the receipt of Data_PDUs. It can also indicate that an MTA previously in EMCON has now come out of EMCON and hence has acknowledge any Data_PDUs received during EMCON.

e. If an acknowledgement has not been received from all the recipients of a message and the re-transmission timer of the message expires, the complete message is re-sent. this is shown in the "None of message acknowledged" event from bubbles 4 to 3.

f. If the complete message has been acknowledged by a recipient, the recipient's address is removed from the Address_PDU. This is shown in the "update Address_PDU recipient list" event from bubbles 26 to 1.

g. The "EMCON" Re-transmission Count" is determined from the priority of the message (bubble 24).

h. The re-transmission timer of a message is determined from the messages original priority. this allows re-transmission of high priority messages before lower priority messages.

i. When the Expiry_Time of a message is reached the event "Message Expiry timer expired" from bubbles 1 to 5 occurs causing a Discard_Message_PDU to be transmitted to all recipients of the message. If the receiving MTA has received the entire message the Discard_Message_PDU is ignored, otherwise the elements of the expired message, which have been received, are deleted.

j. When the EMCON re-transmission count of a message is reached the event "EMCON Re-transmission Count of Message reached" from bubbles 1 to 21 occurs causing the re-transmission timer of the message to be cancelled.

k. When an Ack_PDU is received from an MTA, which was in EMCON, Multicast OUT determines if the message associated with the Ack_PDU was completely received. If the message was completely received the PP Queue Status of the recipient is updated to indicate successful delivery. this is shown in the "All Data_PDUs received" and "Updated message status" events from bubbles 14 to 15 and bubbles 15 to 26 respectively.

l. If an Ack_PDU is received from an MTA, which was in EMCON , and the message associated with the Ack_PDU has expired, A Discard_Message_PDU is sent to the MTA. The PP Queue Status of the recipient is updated to indicate that the message has timed out. this is shown in the events "message has expired". "Transmitted Discard_Message_PDU" and "Updated message status" from bubbles 16 to 20, bubbles 20 to 17 and bubbles 17 to 1 respectively.

m. If an Ack_PDU has not been completely received and has not expired, re-transmission of the missing parts of the message commences.

C03 Notes on Multicast IN State Transition Diagram

a. The Multicast IN STD does not mention the Inbound Handler, Message Received Store or Message Receiver. The STD describes the events of the Multicast IN channel as a single entity.

b. The Multicast IN STD is split into two parts:

(1) The "Multicast IN – Receiving Messages when EMCON is not active" STD shows the STD of Multicast IN when EMCON is not active. It includes the processing necessary when a receiving MTA comes out of EMCON.

(2) The "Multicast IN – Receiving Messages when EMCON is active" STD shows the STD of Multicast IN when receiving MTA is in EMCON.

c. The "EMCON activated at this MTA" event from bubbles 1 to 26 occurs when Multicast IN detects that the receiving MTA has changed from a Normal (Non-EMCON condition) to an EMCON condition.

d. The "EMCON deactivated at this MTA" event from bubbles 26 to 25 occurs when Multicast IN detects that the receiving MTA has changed from an EMCON condition to a Normal (Non-EMCON condition).

e. If a Discard_Message_PDU is received for a message, which has been completely received and has been introduced into the input PP Queue Area, the Discard_Message_PDU is ignored. This is shown in the "All Aata PDUss received" event from bubbles 29 to 1.

f. If a duplicate Address_PDU is received for a message the duplicate Address_PDU is ignored. this is shown in the "Address_PDU recognized" event from bubbles 28 to 30.

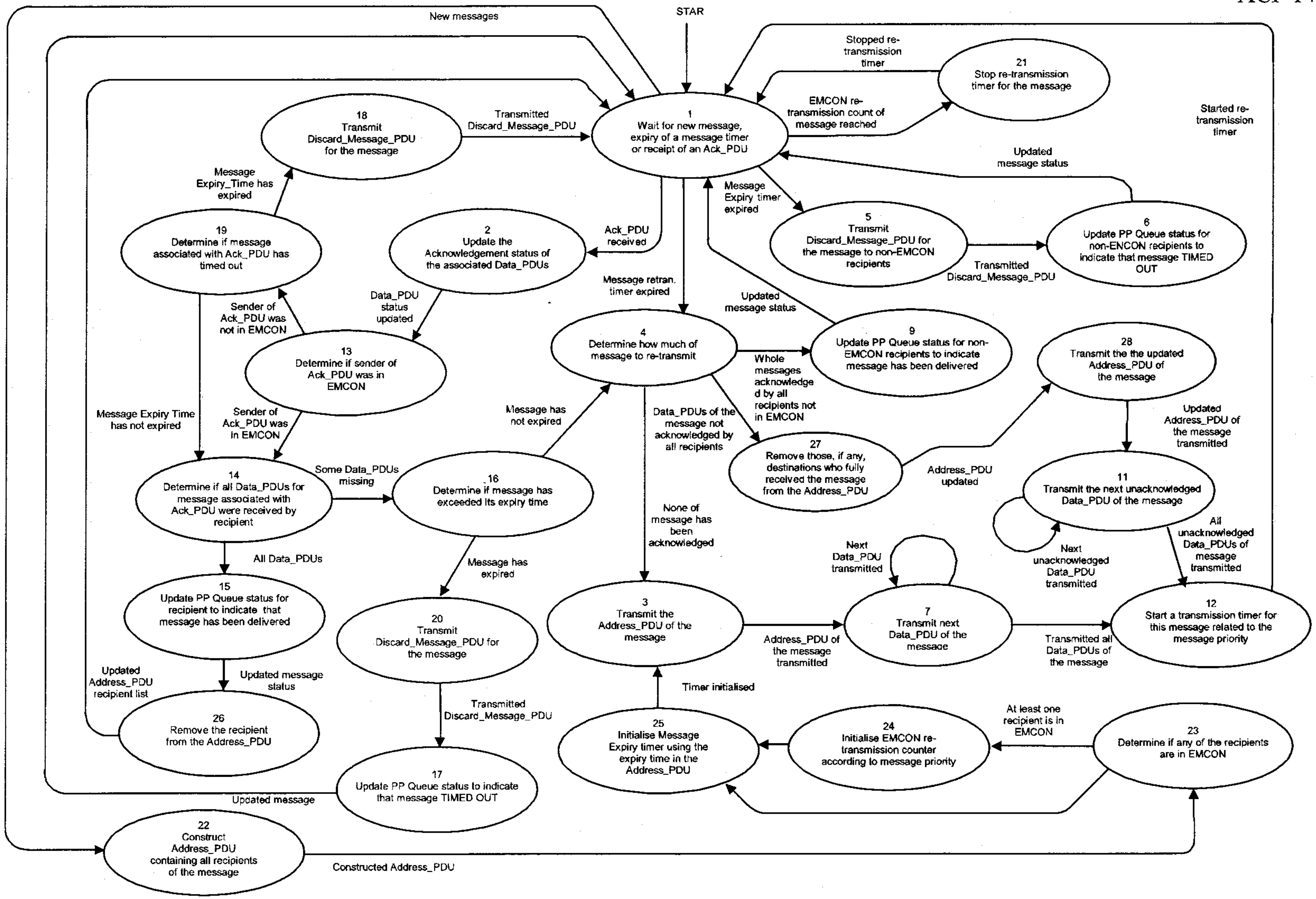


Figure C-1: Multicast Out (Transmitter)

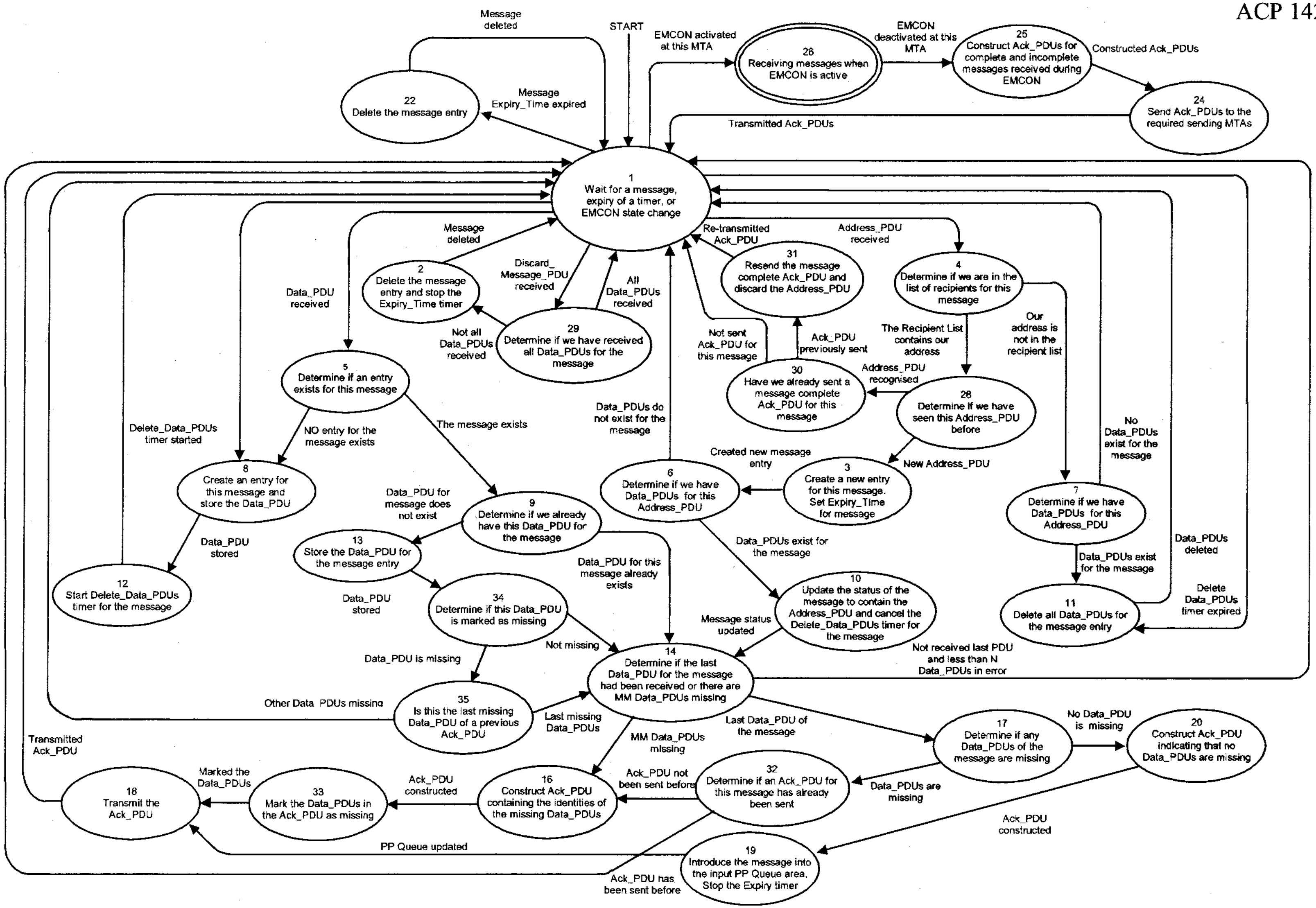


Figure C-2: Multicast messages when EMCON is not active (Receiver)

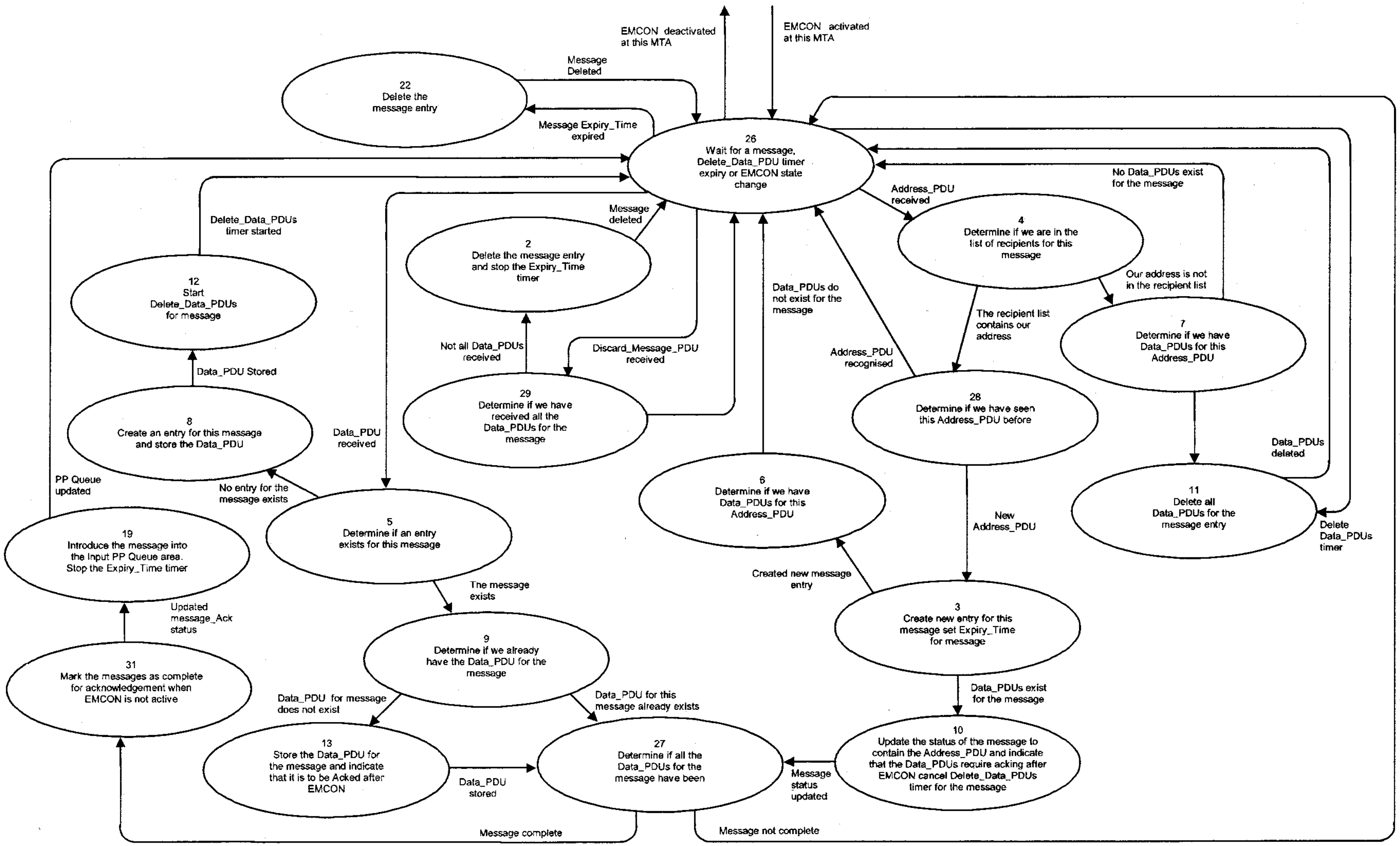


Figure C-3: Multicast messages when EMCON is active (Receiver)

ANNEX D

OPERATIONAL GUIDANCE

D01 Purpose

This annex addresses operational considerations to how this protocol is to be used in the tactical messaging environment and the impact of parameter settings.

D02 Introduction

P-MUL provides connection-oriented reliable multicast messaging for a pre-established or dynamically established multicast group. In addition, it provides an enhanced connectionless delivery mechanism, which transmits a message multiple times and accepts late acknowledgements when receivers leave EMCON. It can be optimised for low bandwidth and high delay environments but its use is not limited to those environments.

D03 Considerations:

a. There are two focus areas with respect to the operation of P-MUL.

(1) Multicast Addressing Scheme
(2) Configurable variables and parameter settings impact on P_MUL performance

b. These focus areas affect the performance and determine the behaviour of the protocol. All factors which influence the behaviour of the protocol will not be discussed nor will a recommendation be made as to how to configure this protocol. This section will simply discuss the protocol's behaviour as a result of how these factors affect it.

D04 Multicast Addressing Scheme

a. The class D addressing scheme can be done according to:

(1) Predefined Grouping Destination MTA
(2) Predefined Grouping by Network reachability
(3) Dynamic Group Allocation by Destination MTA
(4) One multicast group for all potential recipients.

b. Regardless of the method chosen when implementing this protocol within an operational environment the range of multicast addresses used must be defined and managed. All P_MUL transmitters must know the range of multicast IP addressed.

D05 Predefined Grouping by Destination MTA

If the multicast groups are defined according to destination MTA, then class D addresses must be reserved to permit all of the possible combinations of the destination MTAs. The number of Class D addresses required grows exponentially to the number of MTAs as examples shown in Figure D-1. This growth is characterised by the equation

Number of class D addresses required = n!/((k!)*(n-k)!)
n = Total number of MTAs in the group.
k = Number of MTAs within the group addressed in message

Groupings (k)	Number of Combinations		
	n = 6 ships or MTAs in group	n = 10 ships or MTAs in group	n = 15 ships or MTAs in group
1	6	10	15
2	15	45	105
3	20	120	455
4	15	210	1365
5	6	252	3003
6	1	210	5005
7	-	120	6435
8	-	45	6435
9	-	10	5005
10	-	1	3003
11	-	-	1365
12	-	-	455
13	-	-	105
14	-	-	15
15	-	-	1
Total	63	1023	32767

Requires too many class D addresses (→ 32767)

Figure D-1: Multicast addressing scheme according to destination MTA

D06 Predefined Groupings by Network Reachability

a. If the multicast groups are defined according to network reachability, then consideration must be given to how best to group the class D. Embedded within this approach is the assumption that there is an awareness of the network topology and that any changes in the network topology is immediately reflected in the multicast group assignments. An example is plotted in Figure D-2.

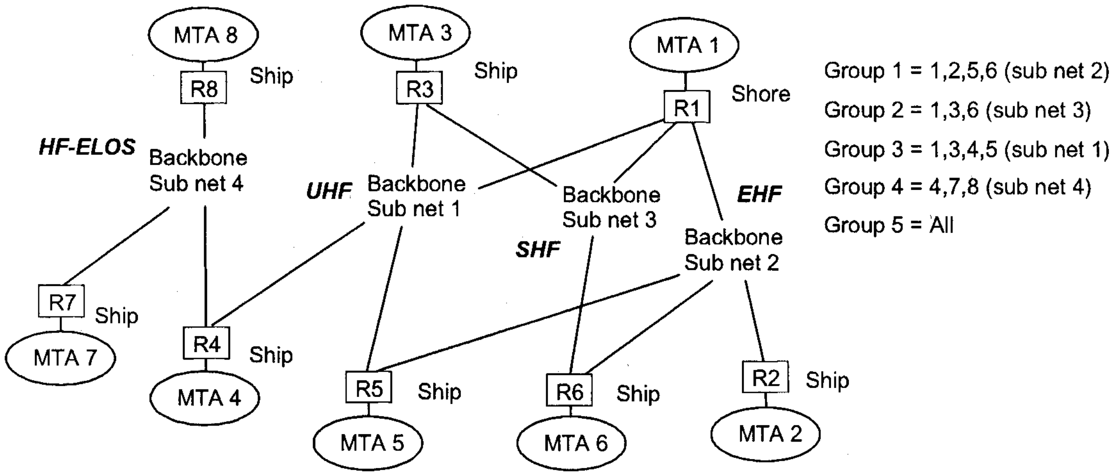


Figure D-2: Multicast address grouping according to network reachability

b. The advantage of this approach is that each network would at worst case only receive 1 message per independent of the number of MTAs on the message.

D07 Dynamic Group Allocation by Destination MTA

With this approach a multicast group is only formed when required. This approach is similar to predefined grouping by destination MTA except that the groups are not predetermined and no Class D addresses reserved for the possible combinations. Therefore there is no exponential growth in Class D addresses required.

D08 One Multicast Group

This approach simply defines all of the potential MTAs within one multicast group. The disadvantage of this approach is that it can be extremely inefficient if all of the potential recipients are not on a shared network. Unnecessary traffic would be introduced on networks which for a given message did not have recipient MTAs in the message.

D09 Configurable Variables and Their Impact on P_MUL

a. The assignment of values to the configurable parameters in P_MUL determines the behaviour and performance of the protocol. As the operators of this protocol set these values, the following information can be used as a guide to understand how the protocol will behave based on the decisions of the operator. This document will not discuss all of the configurable variables but a select few that have more significance.

b. **EMCON Re-Transmission counter (EMCON_RTC):** This parameter is used to determine the maximum number of times the transmitter will re-transmit the information to receiving nodes that are in EMCON. This variable can be used to increase the probability of reception for users in EMCON by setting a value other than 0. The operator should consider factors such as the average size of messages, bit error rate of the bearer, and whether or not there is a desire for P_MUL to manage EMCON re-transmissions. If there is a higher level mechanism managing EMCON re-transmissions, then the EMCON Re-transmission Counter value can be set to zero and bandwidth conserved during EMCON with the higher level mechanism administrating EMCON re-transmissions. Typical values range from 0-5.

c. **Acknowledgement PDU Timer (ACK_PDU_TIME):** This timer determines how long the receiver waits before sending acknowledgements to the transmitter when it has not received a response from the transmitter node. This timer can be used to prevent acknowledgement implosion. If this timer is too low then the receiver will tend to send unnecessary acknowledgement not giving the transmitter sufficient time to respond to the receiver's acknowledgements. When this variable is tuned to the network and the message load, it helps to minimise the number of unnecessary re-transmissions by the transmitter.

d. **Acknowledgement Re-transmission Timer (ACK_RE-TRANSMISSION_TIME):** This timer determines how long a transmitter waits for acknowledgements from the receivers not in EMCON before re-transmitting a message. This variable can be used to make the transmitter responsive to network delays and round trip times. If this value is too low with respect to network delays and round trip times then there will be unnecessary re-transmission potentially crippling the networks. the higher the value the fewer unnecessary re-transmissions.

e. **Relationship of Minimum Scan Time (MIN_SCAN_TIME) to PDU_DELAY. T**he minimum scan time determines how often the receiver scans its directory to see if PDUs have been received, sending acknowledgements when it believes the transmitter should have sent the next PDU. The PDU_DELAY determines how long the transmitter waits before sending successive PDUs. As the PDU_DELAY exceeds the MIN_SCAN_TIME unnecessary acknowledgements increase significantly.

D10 Conclusion

P_MUL can be extremely efficient when careful consideration is given to the multicast addressing scheme used and the assignment values to configurable values. Because P_MUL provides for reliable multicast delivery of information, the transmitter and the receivers must be responsive to each other and their communication environment. Inefficiencies are realised when overcompensation and under-compensating occur. When P_MUL parameters are tuned properly to the network and to each other between transmitting and receiving nodes, the protocol is extremely efficient and quiet, in terms of network traffic.

ANNEX E

DEFINITIONS

Broadcast operation: The transmission of signals that may be simultaneously received by stations that usually make no acknowledgement.

Datagram: In packet switching, a self-contained packet, independent of other packets, that contains information sufficient for routing from the originating data terminal equipment (DTE) to the destination DTE without relying on prior exchanges between the equipment and the network. *Note:* Unlike virtual call service, when datagrams are sent there are no call establishment or clearing procedures. Thus, the network may not be able to provide protection against loss, duplication, or misdelivery.

Emission control (EMCON): The selective and controlled use of electromagnetic, acoustic, or other emitters to optimise command and control capabilities while minimising, for operations security (OPSEC): (a) detection by enemy sensors; (b) to minimise mutual interference among friendly systems; and/or (c) to execute a military deception plan.

Full-duplex (FDX) circuit: A circuit that permits simultaneous transmission in both directions.

Half-duplex (HDX) operation: Operation in which communication between two terminals occurs in either direction, but in only one direction at a time. *Note:* Half-duplex operation may occur on a half-duplex circuit or on a duplex circuit, but it may not occur on a simplex circuit. *Synonyms* **one-way reversible operation, Two-way alternate operation**.

Multicast: 1. In a network, a technique that allows data, including packet form, to be simultaneously transmitted to a selected set of destinations. *Note:* Some networks, such as Ethernet, support multicast by allowing a network interface to belong to one or more multicast groups. **2.** To transmit identical data simultaneously to a selected set of destinations in a network, usually without obtaining acknowledgement of receipt of the transmission.

Multicast address: A routing address that (a) is used to address simultaneously all the computers in a group and (b) usually identifies a group of computers that share a common protocol, as opposed to a group of computers that share a common network. *Note:* Multicast address also applies to radio communications. *Synonym (in Internet protocol)* **class d address**.

Open Systems Interconnection (OSI)-Protocol Specification: The lowest level of abstraction within the OSI standards scheme. *Note:* Each OSI-Protocol Specification operates at a single layer. Each defines the primitive operations and permissible responses required to exchange information between peer processes in communicating systems to carry out all or a subset of the services defined within the OSI-Service Definitions for that layer.

Open Systems Interconnection–Reference Model (OSI–RM): An abstract description of the digital communications between application processes running in distinct systems. The model employs a hierarchical structure of seven layers. Each layer performs value-added service at the request of the adjacent higher layer and, in turn, requests more basic services from the adjacent lower layer.

- **Physical Layer:** Layer 1, the lowest of seven hierarchical layers. The Physical layer performs services requested by the Data Link Layer. The major functions and services performed by the physical layer are (a) establishment and termination of a connection to a communications medium; (b) participation in the process whereby the communication resources are effectively shared among multiple users, *eg*, contention resolutions and flow control; and (c) conversion between the representation of digital data in user equipment and the corresponding signals transmitted over a communications channel.

- **Data Link Layer:** Layer 2. this layer responds to service requests from the Network Layer and issues service requests to the Physical Layer. The Data Link Layer provides the functional and procedural means to transfer data between network entitles and to detect and possibly correct errors that may occur in the Physical Layer. *Note:* Examples of data link protocols are HDLC and ADCCP for point-to-point or packet-switched networks and LLC for local area networks.

- **Network Layer:** Layer-3. This layer responds to service requests from the Transport Layer and issues service requests to the Data Link Layer. The Network Layer provides the functional and procedural means of transferring variable length data sequences from a source to a destination via one or more networks while maintaining the quality of service requested by the Transport Layer. The Network Layer performs routing, flow control, segmentation/desegmentation,and error control function.

- **Transport Layer:** Layer 4. This layer responds to service requests from the Session Layer and issues service requests to the Network Layer. The purpose of the Transport Layer is to provide transparent transfer of data between end users, thus relieving the upper layers from any concern with providing reliable and cost-effective data transfer.

- **Session Layer:** Layer 5. This layer responds to service requests from the Presentation Layer and issues service requests to the Transport Layer. the Session Layer provides the mechanism for managing the dialogue between end-user application processes. It provides for either duplex or half-duplex operation and establishes checkpointing, adjournment, termination, and restart procedures.

- **Presentation Layer:** Layer 6. This layer responds to service requests from the Application Layer and issues service requests to the Session Layer. The Presentation Layer relieves the Application Layer of concern regarding syntactical differences in data representation within the end-user systems. *Note:* An example of a presentation service would be the conversion of an EBCDIC-coded text file to an ASCII-coded file.

- **Application Layer:** Layer 7, the highest layer. this layer interfaces directly to and performs common application services for the application processes; it also issues requests to the Presentation Layer. The common application services provide semantic conversion between associated application processes. *Note:* Examples of common application services of general interest include the virtual file, virtual terminal and job transfer and manipulation protocols.

Protocol data unit (PDU): 1. Information that is delivered as a unit among peer entities of a network and that may contain control information, address information, or data. 2. In layered systems, a unit of data that is specified in a protocol of a given layer and that consists of protocol-control information of the given layer and possible user data of that layer.

Simple Mail Transfer Protocol (SMTP): The Transmission Control Protocol (TCP/IP) standard protocol that facilitates transfer of electronic-mail messages, specifies how two systems are to interact, and specifies the format of messages used to control the transfer of electronic mail.

Simple Network Management Protocol (SNMP): The Transmission Control Protocol/Internet Protocol (TCP/IP) standard protocol that (a) is used to manage and control IP gateways and the networks to which they are attached, (b) uses IP directly, bypassing the masking effects of TCP error correction, (c) has direct access to IP diagrams on a network that may be operating abnormally, thus requiring management, (d) defines a set of variables that the gateway must store, and (e) specifies that all control operations on the gateway are a side-effect of fetching or storing those data variables *ie*, operations that are analogous to writing commands and reading status.

Simple operation: **1.** Operation in which transmission occurs in one and only one pre-assigned direction. *Synonym* **one-way operation**. *Note:* Duplex operation may be achieved by simplex operation of two or more simplex circuits or channels. **2.** Operating method in which transmission is made possible alternately in each direction of a telecommunication channel, for example by means of manual control. *Note:* In general, duplex operation and semi-duplex operation require two frequencies in radio communications; simplex operation may use either one or two. *Note 2:* These two definitions are contradictory, however, both are in common use. The first one is used in telephony and the second one is in radio. The user is cautioned to verify the nature of the service specified by this term.

Source quench: A congestion-control technique in which a computer experiencing data traffic congestion sends a message back to the source of the messages or packets causing the congestion, requesting that the source stop transmitting.

Transmit flow control: In data communications systems, control of the rate at which data are transmitted from a terminal so that data can be received by another terminal. *Note 1:* Transmit flow control may occur between data terminal equipment (DTE) and a switching centre, via data circuit-terminating equipment (DCE), or between two DTEs. The transmission rate may be controlled because of network or DTE requirements. *Note 2:* Transmit flow control can occur independently in the two directions of data transfer, thus permitting the transfer rates in one direction to be different from the transfer rates in the other direction.

UDP: *Abbreviation for* **user datagram protocol**. An Internet protocol for datagram service.

Unidirectional operation: Operation in which data are transmitted from a transmitter to a receiver in only one direction.

X.400: A set of protocols defining the structure, delivery and processing of electronic mail messages.

X.500: A set of protocols defining a distributed directory structure, accesses, replication and related directory functions.

LIST OF EFFECTIVE PAGES

Subject Matters	Page Numbers
Title Page	I
Foreword	II
Letter of Promulgation	III
Record of Message Corrections	IV
Table of Contents	V to VII
List of Figures	VIII
Chapter 1	1-1 to 1-6
Chapter 2	2-1 to 2-16
Chapter 3	3-1 to 3-11
Chapter 4	4-1 to 4-3
Chapter 5	5-1
Annex A	A-1 to A-10
Annex B	B-1 to B-4
Annex C	C-1 to C-6
Annex D	D-1 to D-5
Annex E	E-1 to E-4
List of Effective Pages	LEP-1