INTERNATIONAL STANDARD

ISO/IEC 14476-5

First edition
2008-04-15

Information technology — Enhanced communications transport protocol: Specification of N-plex multicast transport

Technologies de l'information — Protocole de transport de communications amélioré: Spécification pour le transport N-plex en multidiffusion

Reference number
ISO/IEC 14476-5:2008(E)

© ISO/IEC 2008

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS

Not for Resale

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2008

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS

© ISO/IEC 2008 – All rights reserved

Not for Resale

CONTENTS

		Page
1	Scope	1
2	References	1
	2.1 Normative references	1
	2.2 Informative references	1
3	Definitions	1
4	Abbreviations	2
5	Conventions	3
6	Overview	3
7	Considerations	5
	7.1 Participants	5
	7.2 Data channel and addressing	6
	7.3 Control channel and tree	6
	7.4 Tokens	8
	7.5 Logical tree adaptation	8
8	Packets	10
	8.1 Base header	10
	8.2 Extension elements	12
	8.3 Packet format	17
9	Procedures	35
	9.1 Connection management	35
	9.2 Logical tree management	37
	9.3 Multicast data transport	42
	9.4 Token control	45
	9.5 RTT measurement	47
10	System parameters	47
Annex A – Application programming interfaces		49
	A.1 Overview	49
	A.2 ECTP-5 API functions	49
Annex B – State transition diagrams		54
Annex C – An example of system parameters values in ECTP-5		56

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 14476-5 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 6, *Telecommunications and information exchange between systems* in collaboration with ITU-T. The identical text is published as ITU-T Rec. X.608 (02/2007).

ISO/IEC 14476 consists of the following parts, under the general title *Information technology — Enhanced communications transport protocol*:

— *Part 1: Specification of simplex multicast transport*

— *Part 2: Specification of QoS management for simplex multicast transport*

— *Part 3: Specification of duplex multicast transport*

— *Part 5: Specification of N-plex multicast transport*

© ISO/IEC 2008 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Introduction

ECTP is designed to support tightly controlled multicast connections in simplex, duplex and N-plex applications. This part of ECTP (Part 5: ITU-T Rec. X.608 | ISO/IEC 14476-5) specifies the protocol mechanisms for the N-plex multicast data transport.

In the N-plex multicast connection, the participants include one TC-Owner and many TS-users. TC-Owner will be designated among the TS-users before the connection begins. TC-Owner is at the heart of multicast group communications. It is responsible for overall connection management by governing the connection creation and termination, multicast data transport, and the late join and leave operations.

In the N-plex multicast connection, the multicast data transmissions are allowed by TS-users as well as TC-Owner. Each TS-user is allowed to send multicast data to the group only if it gets a token from the TC-Owner. That is, the multicast data transmissions of TS-users are controlled by TC-Owner.

The N-plex multicast connection specified in this Recommendation | International Standard targets the many-to-many multicast applications in which many participants (TS-users) may want to transmit the multicast data to all the other TS-users. Typical examples of such applications include 'teleconferencing' and 'multi-users on-line game', etc. In the teleconferencing application, TC-Owner may act as a 'conferencing server', and all the other participants (TS-users) may send multicast data, such as voice, text and image, to the other participants.

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

--`,,```,,,,````-`-`,,`,,`,`,,`---

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS

Not for Resale

INTERNATIONAL STANDARD
ITU-T RECOMMENDATION

Information technology – Enhanced communications transport protocol: Specification of N-plex multicast transport

1 Scope

This Recommendation | International Standard specifies the N-plex multicast transport connection in which all participants are TS-users and one of them is TC-Owner. The N-plex multicast transport connection allows TS-users to send the multicast data to all the group members. It is noted that a TS-user is allowed to send the multicast data to the group, only if it gets a token from TC-Owner.

This Specification describes the protocol for supporting the N-plex multicast transport, which includes the connection management (establishment, termination, user join and leave) and the reliability control mechanisms for the multicast data transport.

2 References

2.1 Normative references

The following Recommendations and International Standards contain provisions which, through reference in this text, constitute provisions of this Recommendation | International Standard. At the time of publication, the editions indicated were valid. All Recommendations and Standards are subject to revision, and parties to agreements based on this Recommendation | International Standard are encouraged to investigate the possibility of applying the most recent edition of the Recommendations and Standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards. The Telecommunication Standardization Bureau of the ITU maintains a list of currently valid ITU-T Recommendations.

- ITU-T Recommendation X.601 (2000), *Multi-peer communications framework*.
- ITU-T Recommendation X.602 (2004) | ISO/IEC 16513:2005, *Information technology – Group management protocol*.
- ITU-T Recommendation X.605 (1998) | ISO/IEC 13252:1999, *Information technology – Enhanced communications transport service definition*.
- ITU-T Recommendation X.606 (2001) | ISO/IEC 14476-1:2002, *Information technology – Enhanced communications transport protocol: Specification of simplex multicast transport*.
- ITU-T Recommendation X.606.1 (2003) | ISO/IEC 14476-2:2003, *Information technology – Enhanced communications transport protocol: Specification of QoS management for simplex multicast transport*.
- ITU-T Recommendation X.607 (2007) | ISO/IEC 14476-3:2008, *Information technology – Enhanced communications transport protocol: Specification of duplex multicast transport*.

2.2 Informative references

- ITU-T Recommendation X.607.1 (draft) | ISO/IEC 14476-4, *Information technology – Enhanced communications transport protocol: Specification of QoS management for duplex multicast transport*.

3 Definitions

This Recommendation | International Standard is based on the following definitions, which were specified in Enhanced Communications Transport Service (ITU-T Rec. X.605 | ISO/IEC 13252).

a) Transport connection: Simplex, Duplex and N-plex.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

This Recommendation | International Standard redefines the following definitions specified in Enhanced Communications Transport Service (ITU-T Rec. X.605 | ISO/IEC 13252).

a) **TC-owner (TCN)**: TCN manages overall operations of an N-plex multicast connection.

b) **transport service user (TS-user)**: TS-users can send and receive multicast data in the N-plex multicast connection.

c) **sending TS-user (SU)**: A TS-user who gets a token from TCN. Only the SU is allowed to send multicast data to the group. In other words, before sending multicast data, each TS-user must request a token to TCN.

This Recommendation | International Standard redefines the following terminologies specified in Enhanced Communications Transport Protocol: part 1 (ITU-T Rec. X.606 | ISO/IEC 14476-1) to accommodate to N-plex multicast connection.

a) **local group**: A set of nodes in vicinity which has network-layer correlation in terms of packet loss and delay.

b) **local owner (LO)**: LO is a representative node of a local group and designated statically. It is responsible for maintaining an intra-group tree of the group and control trees for all SUs in its local group. Each LO is also connected to the other LOs along inter-group trees. It also generates test traffic periodically for logical tree adaptation.

c) **multicast data channel**: TCN or SU can send multicast data to all the other group members over IP multicast address.

This Recommendation | International Standard newly defines the following terminologies:

a) **logical tree**: A tree that spans all TS-users and one or more control trees are derived from it.

b) **inter-group tree**: A per-source logical tree of the LOs.

c) **intra-group tree**: A shared logical tree of each local group.

d) **control tree**: A tree along which control packets for error control are traversed.

e) **token**: It represents the right for a TS-user to transmit multicast data. The TS-user who has a token is called SU. The tokens are managed by TCN.

4 Abbreviations

The following acronyms for ECTP protocols are used in this Recommendation | International Standard:

ECTP-1	ECTP part 1 (ITU-T Rec. X.606 | ISO/IEC 14476-1)
ECTP-2	ECTP part 2 (ITU-T Rec. X.606.1 | ISO/IEC 14476-2)
ECTP-3	ECTP part 3 (ITU-T Rec. X.607 | ISO/IEC 14476-3)
ECTP-4	ECTP part 4 (ITU-T Rec. X.607.1 | ISO/IEC 14476-4)
ECTP-5	ECTP part 5 (ITU-T Rec. X.608 | ISO/IEC 14476-5)
ECTP-6	ECTP part 6 (ITU-T Rec. X.608.1 | ISO/IEC 14476-6)

The following acronyms for ECTP-5 packets are used in this Recommendation | International Standard:

ACK	Acknowledgment
CC	Connection Creation Confirm
CCC	Control Tree Change Confirm
CCR	Control Tree Change Request
CR	Connection Creation Request
CT	Connection Termination Request
DT	Data
JC	Late Join Confirm
JR	Late Join Request
LR	User Leave Request
NACK	Negative Acknowledgement
PB	Probe

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

PBACK	Probe Acknowledgment
RD	Retransmission Data
TC	Tree Join Confirm
TCC	Tree Change Confirm
TCR	Tree Change Request
TDC	Tree Delegation Confirm
TDR	Tree Delegation Request
TGC	Token Get Confirm
TGR	Token Get Request
TJ	Tree Join Request
TLC	Tree Leave Confirm
TLR	Tree Leave Request
TNC	Tree Change Notification Confirm
TNR	Tree Change Notification Request
TRC	Token Return Confirm
TRR	Token Return Request
TSR	Token Status Report
TSRR	Token Status Report Request

5 Conventions

In this Recommendation | International Standard, packets of ECTP-5 are represented as words with all capital characters (e.g., CR for Connection Creation Request packet), and system parameters are represented as words with all italic capital characters (e.g., *TD_PACKET_INT* for test data packet interval).

6 Overview

The Enhanced Communications Transport Protocol (ECTP) is a transport protocol designed to support Internet multicast applications. ECTP operates over IPv4/IPv6 networks that have the IP multicast forwarding capability with the help of IGMP and IP multicast routing protocols, as shown in Figure 1. ECTP could possibly be provisioned over UDP.

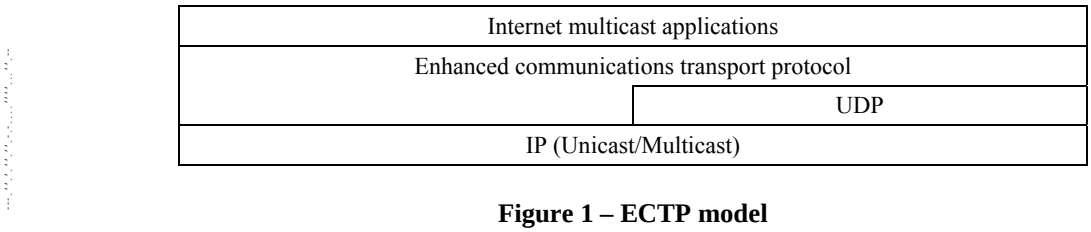

Figure 1 – ECTP model

This Recommendation | International Standard describes the protocol specification of the ECTP part 5 (ECTP-5) for the N-plex multicast connection. The N-plex multicast connection is used for supporting multicast data transport between the participants (TS-users). In the N-plex multicast connection, TS-users can send multicast data packets to the group over multicast data channel. A TS-user who is sending multicast data in the N-plex multicast connection is called SU (Sending TS-user). Any SU must have a token for multicast data transmission. In other words, the TS-user who gets a token from TCN is called an SU.

Figure 2 illustrates the multicast data transport channel in the N-plex multicast connection. As shown in the figure, TCN and SU can transmit multicast data to the other session members over IP multicast (group) address.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

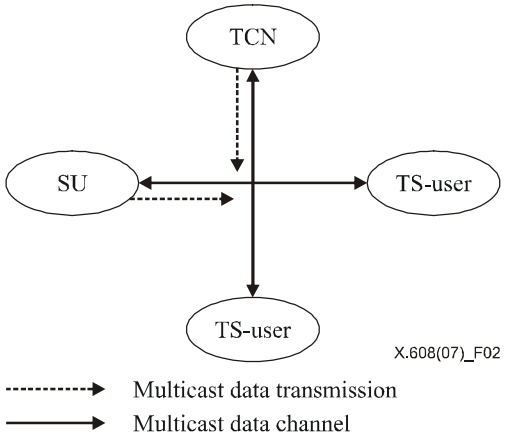

Figure 2 – Multicast data transport in N-plex multicast connection

To establish an N-plex multicast connection, TCN should be activated to manage session information and tokens.

If the TCN is informed of a participant list by out-of-band signalling prior to starting the connection, it should start a connection creation phase by transmitting a CR packet to the group. The CR packet contains the connection information including general characteristics of the connection. Each TS-user who is contained in the participant list should respond to the TCN with a CC packet. The connection creation operation will be completed when the TCN receives CC packets from all of the TS-users in the participant list, and the data transmission phase starts. If there is no predetermined participant prior to starting the session, TCN starts the data transmission phase without connection creation operations.

In the middle of data transmission phase, the prospective TS-users should join the connection as late-joiners. The late-joining TS-user participates in the connection by sending a Late Join Request (JR) message to TCN. In response to the JR message, TCN sends a Late Join Confirm (JC) message to the TS-user. After a TS-user is confirmed to join the session, it should join a logical tree by exchanging the Tree Join Request (TJ) and Tree Join Confirm (TC) messages with an appropriate LO. The logical tree is used for error control.

An N-plex multicast connection builds a two-layer logical tree, which consists of intra-group shared trees and inter-group per-source trees, as shown in Figure 3.

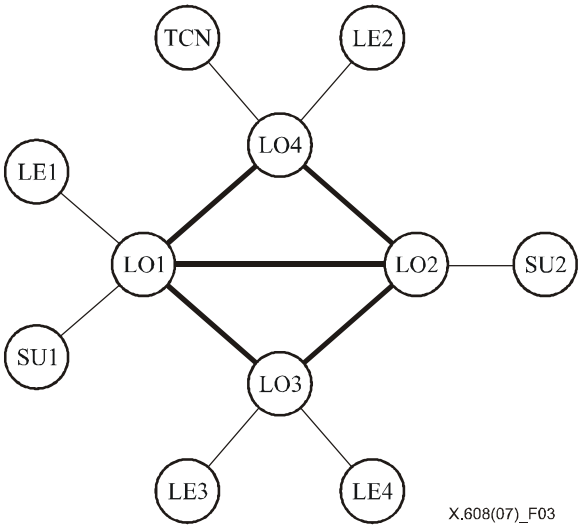

Figure 3 – A two-layer tree for N-plex multicast connection

At the bottom layer, each LE in a local group joins an LO-rooted shared logical tree (*intra-group tree*). At the top layer, LOs constitute logical trees for each SU (*inter-group tree*). It is noted that the control tree for each SU is derived by grafting these inter-group and intra-group trees.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

In N-plex multicast connection, intra-group trees can be organized differently depending on the roles of LEs, which is determined by TCO (Tree Configuration Option). One option of TCO is that all LEs are direct children of LOs and do not participate in the repair process and ACK aggregations. Thus, intra-group trees will be an LO-rooted one-level tree in this option. The other option is that LEs can be children of other LEs. In this option, LEs are responsible for reliability support of its children LEs and intra-group trees can be multi-level trees. For the sake of efficiency of reliability support, the multi-level intra-group trees should be as close as possible to underlying multicast routing tree. The N-plex multicast connection adopts logical tree adaptation mechanism for this option. For this, error bitmaps of TS-users are used. An error bitmap represents packet delivery status, which indicates the loss pattern of multicast packets. Each TS-user sends its error bitmap to its parent with respect to multicast data from the root node of its intra-group tree with periodic ACK messages. By comparing error bitmaps of itself and those of its children, a node decides whether each child is likely to be its actual child in the underlying multicast routing tree or not. If the child node is determined not to be its actual child, a node changes the logical tree by delegating the child node to its parent or one of its other children. After recursive tree changes, the intra-group tree will converge to a multi-level tree that is closer to underlying multicast routing tree.

The error control will be performed based on the ECTP control tree which is constructed as described above. If packet loss is detected by a gap of the packet sequence numbers, a child node sends a NACK packet to its parent immediately via unicast. The parent LO or SU, that receives the NACK packet, will retransmit data packet (RD) to the requestor via unicast. Each child generates an ACK packet every *ACK_GENERATION_NUM* (*AGN*) data packets.

In the multicast data transmission, TCN and SUs can begin the multicast data transmission to the group by using the IP multicast address and group port number. The TS-users will deliver the received DT packets to the upper-layer application in the order transmitted by SU or TCN.

For the multicast data transport, a TS-user in the connection may get a token from TCN by sending a TGR message. The TCN will then respond to the TS-user with the TGC message that contains a *Token ID*. Accordingly, the total number of tokens in the connection is controlled by TCN. The TS-user who has a token is called Sending TS-user (SU). In a certain case, the TCN can first request a TS-user to be an SU by sending the TGR message to the promising SU, which is called Token Give.

After completing the multicast data transmission, the SU will return the token to the TCN by sending a TRR message. TCN will respond to the SU with a TRC message. In a certain case, the TCN first requests the SU to return the token, which is called Token Withdrawal. TCN announces the overall status of the Token IDs valid in the connection to the group by sending the TSR packets.

TCN manages the connection for user leave. In the User Leave operation, a participating TS-user may leave the connection by sending an LR message to the parent. In a certain case, the TCN may enforce a specific TS-user to leave the connection by sending the LR message, which is called the troublemaker ejection.

TCN may terminate the N-plex connection by sending a CT message to the group.

7 Considerations

7.1 Participants

All participants to an N-plex multicast connection are TS-users and one of them is TCN (TC-Owner).

TCN (TC-Owner):

An N-plex multicast connection has a single TCN. The TCN is responsible for connection management including connection creation/termination, late join, connection maintenance, and token management.

For example, in the teleconferencing applications, the TCN may act as the 'conference server', which may be used for control of the conferencing without sending multicast data. In the example of 'multi-users on-line game' application, the TCN may act as the 'game-control server'.

TS-user (Transport Service User):

An N-plex multicast connection has one or more TS-users. Each of them sends and receives multicast data in the connection.

A TS-user can become LO or LE depending on its role.

LO (Local Owner):

LO is a representative node of a local group and designated statically. It is responsible for maintaining an intra-group tree of the group and control trees for all SUs in its local group. Each LO is also connected to the other LOs with inter-group trees. It also generates test traffic periodically for logical tree adaptation.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

LE (Leaf Entity):

LE is a member of a local group whose representative is an LO. It should join an intra-group tree of the group and is responsible for exchanging control packets with its parent or child LEs along the control tree.

A TS-user can become SU when it obtains a token from TCN.

SU (Sending TS-user):

An SU is a TS-user who can send multicast data to the group. In an N-plex multicast connection, a TS-user becomes an SU when it has a token and it can thus transmit multicast data to the group.

7.2 Data channel and addressing

In N-plex multicast connection, SU or TCN can send multicast data packets to the session members as shown in Figure 4.

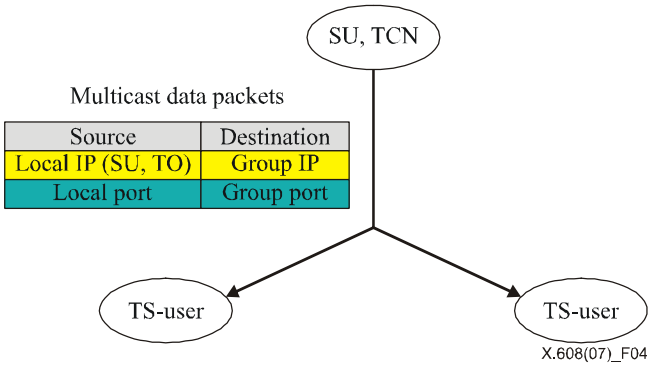

Figure 4 – Multicast data addressing in N-plex multicast connection

In an N-plex multicast connection, the multicast data packets (DTs) use the IP multicast address and the group port number as the destination address. The source address of the multicast data IP packets is the IP address of the sender of the packet. In contrast, the retransmission data packets (RDs) in response to the repair requests (NACKs) are delivered by LO or SU over control channel using unicast with the IP address of the repair requester as the destination.

7.3 Control channel and tree

In an N-plex multicast connection, there are control channels for error recovery. All members participate in one or more control trees. These trees are used as control channels for exchanging control messages among participants. A parent node acts as an agent who helps child nodes to recover from packet loss. It also aggregates the acknowledgement information from its descendant nodes. All control packets such as RD, NACK, and ACK are delivered via control channel using unicast.

An N-plex multicast connection is divided into multiple local groups of the participants for error control. Based on this group approach, participants construct and maintain control trees along which all the error control packets are exchanged. Control trees are constructed from two layers of logical trees. At the bottom layer, members in a local group join an LO-rooted shared logical tree (*intra-group tree*). At the top layer, the LOs of the groups constitute per-source logical trees (*inter-group tree*). That is, every LE joins a local group and it grafts onto LO-rooted logical tree of the group as a child (or descendents) of the LO. Every LO grafts onto logical tree of the LOs, of which root is the LO of the group that the SU belongs to. A control tree for each source is constructed by connecting the inter-group trees and intra-group trees.

Inter-group trees are generated and maintained exactly as required. Any inter-group tree rooted by an LO who has no more SU children should be removed.

Intra-group trees can evolve to multi-level trees that reflect actual multicast routing paths by logical tree adaptation mechanism, which is described in 7.5.

By traversing logical trees starting from an SU, we can get an SU-rooted tree that is the control tree for the SU.

If an SU belongs to another local group, a part of the control tree generated for current local group is the intra-group tree of the group. Control trees for the SUs belonging to the same local group are slightly different from the intra-group tree, because the intermediate nodes between LO and the SU have reversed parent-child relationships.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Assuming that the logical trees of the participants are constructed as in Figure 5, the control tree for source SU1 will be constructed as in Figure 6.

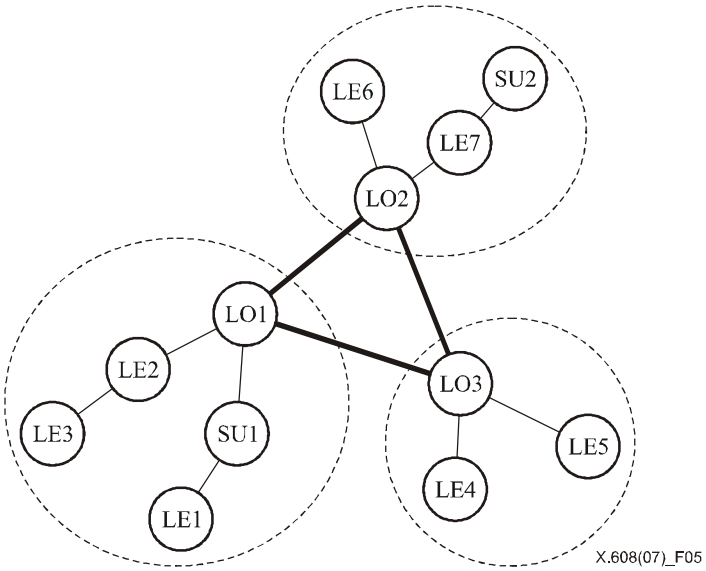

Figure 5 – Two layers of logical trees

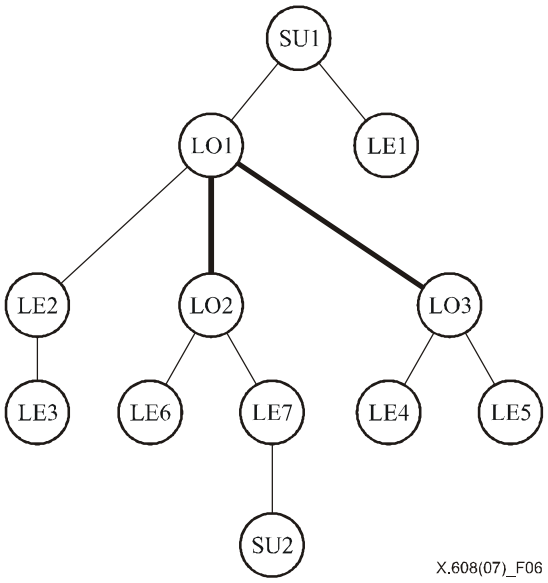

Figure 6 – Control tree when sender is SU1

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

In the same way, Figure 7 shows the control tree whose source is SU2.

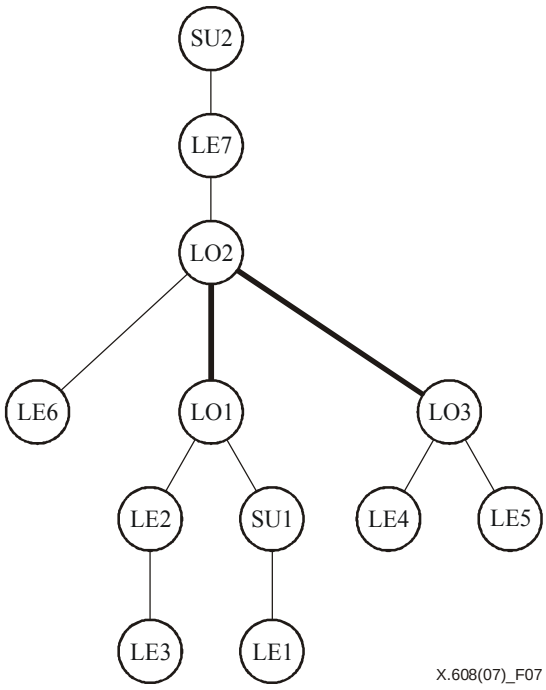

Figure 7 – Control tree when sender is SU2

Control trees are maintained by information obtained through the exchange of NACK and ACK packets. The parent can detect a child's failure by comparing its own LSN and received child's LSN. Children can detect their parent's failure if there is no response from parent for several consecutive repair requests. Then the child finds another parent by contacting LO.

7.4 Tokens

In N-plex multicast connection, a token represents the right for a TS-user to send multicast data to TCN. Before transmitting data, each TS-user must get a token from the TCN, as per the Token Control procedures of N-plex multicast connection. By this procedure, TCN can authorize a TS-user to become a sender so that TS-users can effectively filter out multicast data sent by unauthorized users. However, note that use of token does not provide any protection for IP multicast.

Each token is represented as a 1-byte non-negative integer. Such a token number (or Token ID) will be assigned by TCN when a TS-user requests a token in the connection. Token ID is ranged between 1 and 255. The Token ID of '0' is reserved for use of TCN. At the receiver side, the Token ID can be used to authorize who can send the multicast data.

7.5 Logical tree adaptation

In N-plex multicast connection, intra-group trees may evolve to multi-level trees close to underlying multicast routing trees when TCO is '10'. Loss pattern comparison is used to estimate the underlying multicast routing trees. The estimation is made feasible because if a loss occurs at a parent node of a multicast routing tree, all of its children also experience the same loss. The root of an intra-group tree does not change by the logical tree adaptation process.

For this, each receiver in a multicast session maintains packet delivery status called error bitmap, which indicates the loss pattern of multicast packets. An error bitmap consists of two parts: sequence number (*Ns*) and an actual bitmap (*B*). *Ns* is the sequence number of the first packet in a sequence of packets represented in the bitmap. One bit of *B* indicates the reception status of the corresponding packet; '1' means a successful packet delivery to a receiver without error recovery and '0' otherwise. The bitmap includes loss patterns of multicast packets. For example, if *Ns* = 5 and *B* = 11010, the receiver has successfully received packets 5, 6, and 8. Even packets 7 and 9 have been recovered via retransmissions, these bits are recorded as '0's. Each receiver periodically feeds the error bitmap information to its parent in the tree. If a logical tree matches a multicast routing tree and a bit in the error bitmap at a node is set to 1, the corresponding bit in the error bitmap of its parent is very likely to be '1'. Note that the error bitmap of a source always consists of all '1's.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Every node that has child nodes compares the error bitmaps of itself and its children to check whether the relationships between the node and its child nodes reflect the underlying multicast routing tree closely. A parent node can infer that some child nodes need to change their locations in the logical tree. The child node would be a child of other child node. Or it would not be a descendant of the parent node.

For instance, Figure 8 shows an example of a multicast routing tree and error bitmaps of each node.

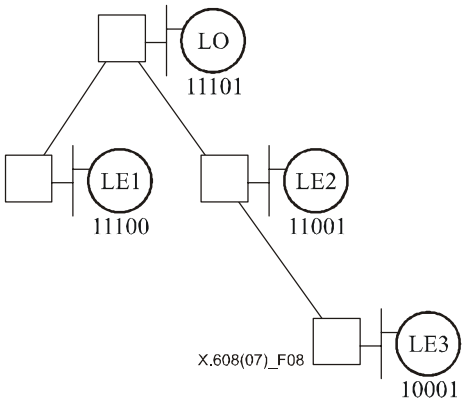

Figure 8 – An example of multicast routing tree

Figures 9 and 10 briefly describe how a logical tree close to a multicast routing tree is constructed by the error bitmap information.

First, it is assumed that a root LO is a strategically deployed server that is usually located nearby the egress point of Internet service provider. Boxes represent routers and numbers below nodes are error bitmaps of each node.

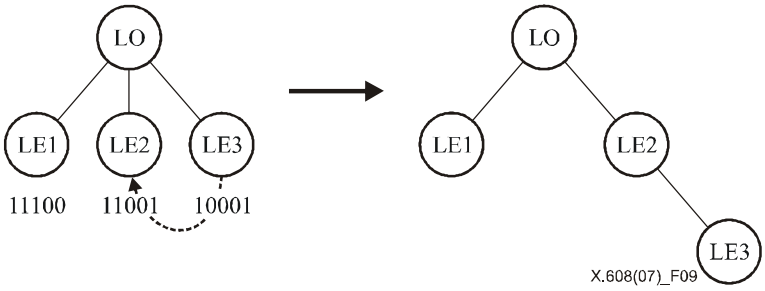

Figure 9 – Logical tree adaptation (LE2 adopts LE3)

The left side of Figure 9 shows an initial one-level intra-group tree of nodes in Figure 8. When LO becomes aware of error bitmaps of its children, it finds out that LE3 is likely to be a child or descendant of LE2 since a set of received packets of LE3 is a subset of that of LE2. Thus, it delegates LE3 to LE2 by sending a TDR (Tree Delegation Request) message. Receiving the message, LE2 judges that LE3 is its child and sends a TCR message (Tree Change Request) to LE3. Receiving the TCR message, LE3 joins as a child of LE2 by sending TJ to LE2 and leaves its previous parent, LO, by sending TLR. Note that a node receiving a TCR message is attached to the new parent before leaving its previous parent. The result is a tree on the right side of Figure 9.

--`,,```,,,,````-`-`,,`,,`,`,,`---
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Figure 10 shows another example of the logical tree adaptation.

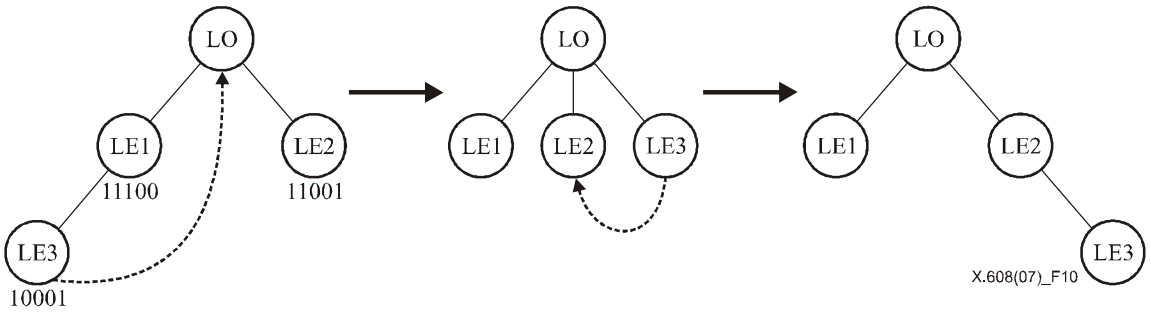

Figure 10 – Logical tree adaptation (LO adopts LE3; LE2 adopts LE3)

On the left side of Figure 10, LE3 has become a child of LE1 by previous tree adaptation mistakenly. However, a current error bitmap of LE3 indicates that LE3 is not a child of LE1 since it received a packet that is not received by LE1. In this case, LE1 delegates LE3 to its parent, LO, to find the proper position of LE3 by sending a TDR message. LO receives the TDR message from LE1 about LE3 and compares error bitmap of LE3 with its children LE1 and LE2. The LO finds out that LE3 is likely to be a child of LE2. Finally, the logical tree converges to the tree on the right side of Figure 10.

In this way, intra-group trees can evolve to multi-level trees that approximate underlying multicast routing trees. Control trees built from these intra-group trees will also be very close to the multicast routing trees.

For this, LO sends test DATA packets when it detects change of the intra-group tree. Then the LEs in the local group construct error bitmap information and send it via ACK packet. The test DATA packets are not destined to any application, and should be sent every *TD_PACKET_INT* and in *TD_PACKET_NUM* times.

8 Packets

An ECTP packet contains a 16-byte base header together with either extension elements or user data. It is noted that the data packets (DTs) do not include any extension elements. The generic packet format of ECTP-5 is illustrated in Figure 11:

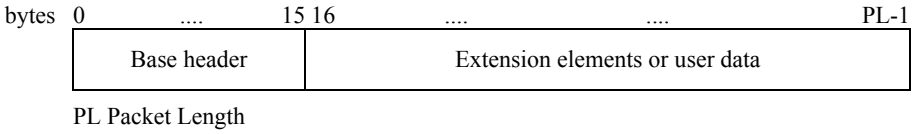

Figure 11 – Packet format of ECTP-5

8.1 Base header

The 16-byte base header contains common information helpful to all the protocol operations, in particular for the data packets. Figure 12 shows the structure of the base header when ECTP operates over IP.

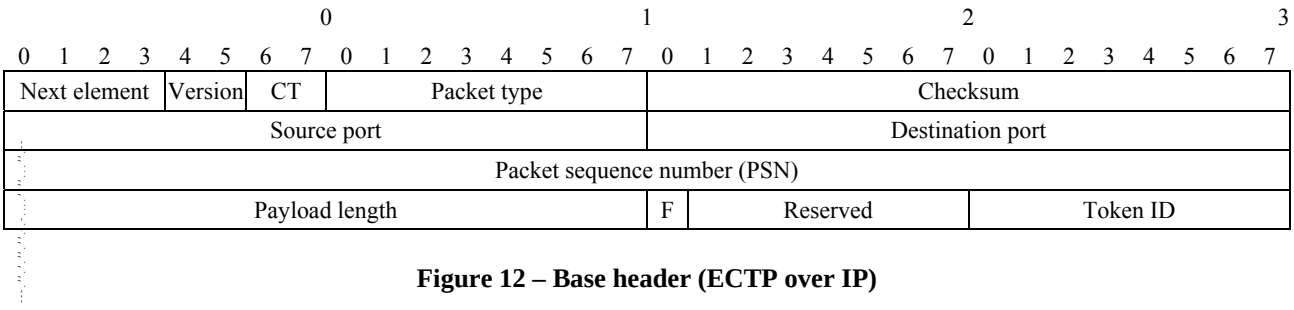

Figure 12 – Base header (ECTP over IP)

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

The base header contains the following information:

Next element (4 bits)

This specifies the type of the extension element immediately following the base header. The encoding values of the extension elements will be described later. The extension element value of '0000' means that there is no more following extension element after this element.

Version (2 bits)

This defines the version of the protocol for this Recommendation | International Standard. Its current version is encoded as '00'.

CT (*Connection Type*) (2 bits)

This specifies the type of the ECTP connection. The encoding value is as follows:

01 – simplex multicast connection (for ECTP-1 and ECTP-2);

10 – duplex multicast connection (for ECTP-3 and ECTP-4);

11 – N-plex multicast connection (for ECTP-5 and ECTP-6).

In this Recommendation | International Standard, the *CT* must be set as '11'.

Packet type (8 bits)

It indicates the type of the packet.

Checksum (16 bits)

This is used to check the validity of the packet that includes the base header, extension header and/or user data. The checksum is calculated by using the conventional one's complement arithmetic operation, as done in TCP and UDP.

Source port (16 bits) and *destination port* (16 bits)

These port numbers are used to identify the sending and receiving applications for the case of ECTP over IP. When ECTP operates over UDP, these fields are used to represent the connection identifier, as described later.

PSN (32 bits)

For the DT packets, this field is a 32-bit unsigned number that starts with the initial sequence number and increases by '1' for every packet, and wraps back around to '1' after reaching '$2^{32} – 1$'.

For the ACK packets, this field is the LSN of an SU with *Token ID*.

For the RD packets, this field is the PSN of the data packet, which is requested to be retransmitted.

For the TJ, TLR, JR, TGR, TRR, TCR, TDR, TNR and CCR, this field is the sequence number of control packets from a node. It should be unique within the same packet type sent by the node.

For the TC, TLC, JC, TGC, TRC, TCC, TDC, TNC, and CCC, this field is copied from the *PSN* field of the corresponding request packet.

For the other packets, this field should be ignored.

Payload length (16 bits)

This value indicates the total length of the extension headers or user data in byte, following the base header.

F (1 bit)

It is a flag bit. The use of this flag depends on the packet types:

For the JC, TC, TGC, TCC, TDC, TRC and TLC packets, the F = 1 indicates that each of the corresponding join request is accepted. F is set to 0, otherwise.

For the TJ and TLR, the F is set to '1' for inter-group tree join or leave, or set to '0' for the intra-group tree join or leave respectively.

For the LR packet, F is set to '1' for the user-invoked leave, or set to '0' for the troublemaker ejection.

For the CT packet, F is set to '1' for an abnormal termination, or set to '0' for the normal termination after all the data have been transmitted.

For the token control operations, TGR and TRR request messages use this flag so as to indicate whether this is the TCN-initiated or TS-user-initiated token control.

For the TSR packet, the F = 1 indicates that the overall token status is changed by adding a new token or deleting an existing token. On the other hand, the F = 0 means that this TSR packet is for the information.

For the TNR, the F is set to '1' for tree join, or F is set to '0' for tree leave.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

For DT packet, F is set to '1' for test DT from LO for logical tree adaptation, or set to '0' for the DT from normal applications.

For RD packet, F is set to '1' if the sender has no data to repair, otherwise '0'.

For the other packets, this field will be ignored.

Reserved (7 bits)

Token ID (8 bits)

The Token ID is valid only for DT and RD packets. This represents the source of the data packets. The Token ID value is ranged between 0 and 255. Each SU receives a Token ID from TCN via the token get and give procedure and sets this field to be the number assigned by TCN.

On the other hand, when ECTP operates over UDP, the packet header does not need to specify the source and destination ports, which will be referred to from the UDP header. In this case, the 32-bit field for the source and destination ports will be filled with '*Connection ID*'. By default, it may be set to be the IPv4 group address.

The base header format for ECTP over UDP is as shown in Figure 13:

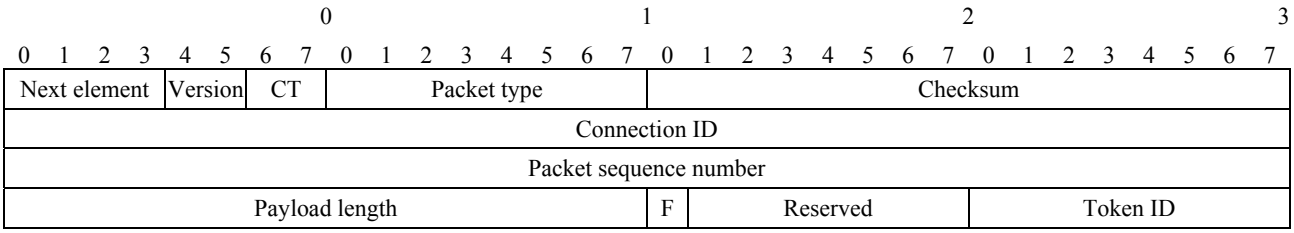

Figure 13 – Base header (ECTP over UDP)

The *Connection ID* is used to identify an ECTP connection by the ECTP host. It may also be used to verify the connection. In the connection setup phase, this information must first be informed by TCN to the other participants via the CR or JC packets. All the other packets in this Recommendation | International Standard must set this field to be the value announced by TCN.

8.2 Extension elements

The ECTP control packets may contain one or more extension elements along with the base header. The based header and each extension element have the field of 'Next Element' that points to the immediately succeeding extension element, if any.

The Next Element field is encoded as shown in Table 1. The last extension element of a packet must set its Next Header field to '0000' (No Element).

Table 1 – Extension elements

Extension element	Encoding value in next element of the preceding element (4 bits)	Length of extension element (bytes)
No Element	0000	0
Connection	0001	4
Error Bitmap	0010	Varied
Timestamp	0100	12
Token	0110	Varied
LO Information	0111	Varied
Negative Acknowledgement	1000	8
Tree Change Information	1001	8

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

8.2.1 Connection element

The connection extension element contains overall information on the transport connection. It is encoded as '0001' in the Next Element field of the preceding element or based header. This extension element must be included in the CR, JC and TGR packets. The element structure is shown in Figure 14, which has the length of '4' bytes:

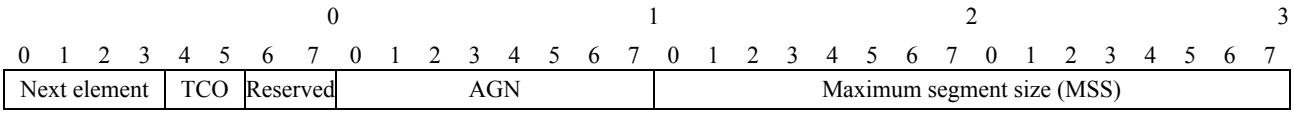

Figure 14 – Connection extension element

Each field is specified as follows:

Next element (4 bits)

This indicates the type of the next extension element, as indicated in Table 1.

TCO (Tree Configuration Option) (2 bits)

This specifies the tree configuration option used in ECTP-5 as follows:

00 – Reserved for future use.

01 – 1-level intra-group tree without logical tree adaptation.

10 – Multiple-level intra-group tree with tree adaptation.

11 – Reserved for future use.

The default value of *TCO* is '10' in N-plex multicast connection.

Reserved (2 bits)

AGN (ACK Generation Number) (8 bits)

This is a positive integer ranged from 1 to 255 (*ACK_GENERATION_NUM*). The *AGN* is used by a child TS-user to generate and transmit an ACK packet to the parent.

MSS (16 bits)

This specifies the maximum size (in byte) of the user data segment (*MAX_SEGMENT_SIZE*) that can be contained in the DT packet. The default value is '1024'.

8.2.2 Error bitmap element

This extension element provides information on the status of the packet reception at a child node. This extension header is attached to ACK packet in response to LO's test traffic for logical tree adaptation as described in 7.5. It is encoded as '0010' in the Next Element field of the preceding element or based header.

This element consists of the fixed 4 bytes and the variable size of *Error Bitmap*, as depicted in Figure 15:

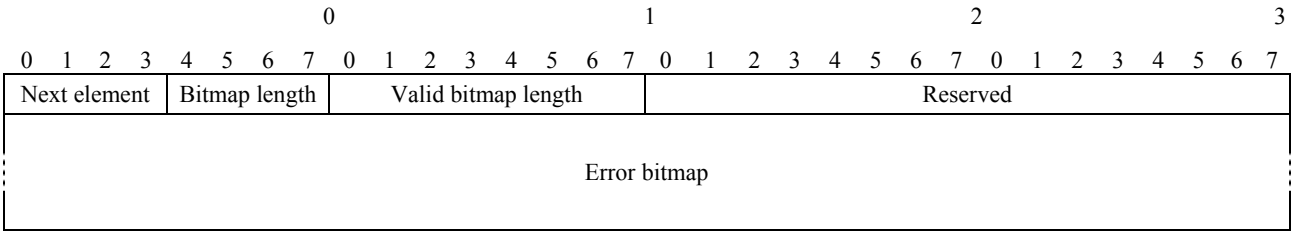

Figure 15 – Error bitmap extension element

Each field is specified as follows:

Next element (4 bits)

This indicates the type of the next extension element, as indicated in Table 1.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Bitmap length (4 bits)

This specifies the total size of the variable *ACK Bitmap* in unit of word (4 bytes).

Valid bitmap length (8 bits)

This represents the length of the actually valid portion in 'bit' for the *ACK Bitmap*.

Reserved (16 bits)

Error bitmap (variable)

This represents information by using '0' or '1' about which data packets have been received (1) or lost (0) at the receiver side. In the *Error Bitmap*, the bitmap information starts with the *LSN* sequence number, which represents the sequence number of the lowest numbered DT packet that has not been received yet at the receiver side. The *LSN* will be specified in the *PSN* field of the base header. The *Error Bitmap* contains the total bitmap information of the size of *Valid Bitmap Length*.

8.2.3 Timestamp element

The Timestamp element is encoded as '0100' in the Next Element field of the preceding element or based header. The ECTP-5 uses the 8-byte timestamp so as to calculate *Round Trip Time* (*RTT*).

The 12-byte timestamp extension element is formatted as shown in Figure 16:

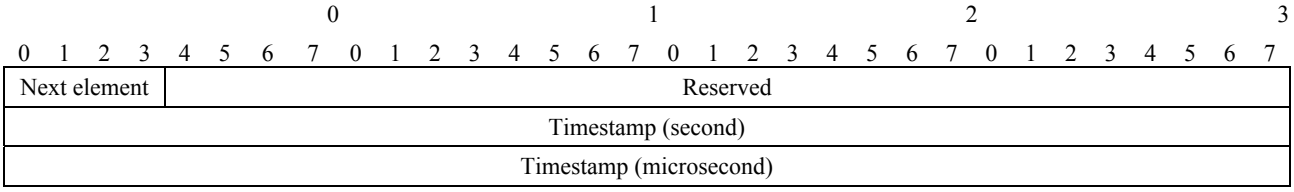

Figure 16 – Timestamp extension element

Each field is specified as follows:

Next element (4 bits)

This indicates the type of the next extension element, as indicated in Table 1.

Reserved (28 bits)

Timestamp (64 bits)

This contains the 8-byte timestamp value. The first 4 bytes represent the time value in second, and the second 4 bytes do in microsecond, as done in the conventional *ping* program.

8.2.4 Token element

This extension element provides information on the status of the tokens currently being used in the connection. This extension header is attached to the TSR packet. It is encoded as '0110' in the Next Element field of the preceding element or based header.

This element consists of the fixed 2 bytes and the variable size of *Valid Token IDs*, as depicted in Figure 17:

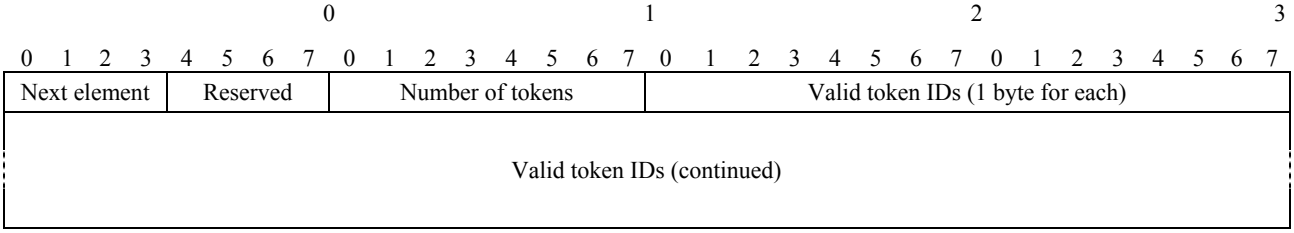

Figure 17 – Token extension element

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

Each field is specified as follows:

Next element (4 bits)

This indicates the type of the next extension element, as indicated in Table 1.

Reserved (4 bits)

Number of tokens (8 bits)

This specifies the total number of *Valid Token IDs* in the connection.

Valid token IDs (varied)

This contains the list of the Token IDs valid in the connection. Each *Token ID* is of 1-byte length.

8.2.5 LO information element

This extension element provides information of an LO and its corresponding token IDs. With this information, an LO can recognize its parent LO along the inter-group tree toward an SU. This extension header is attached to the TSR, TGR and TGC packet. It is encoded as '0111' in the Next Element field of the preceding element or based header. This element consists of the fixed 8 bytes and the variable size of Corresponding Token IDs, as depicted in Figure 18:

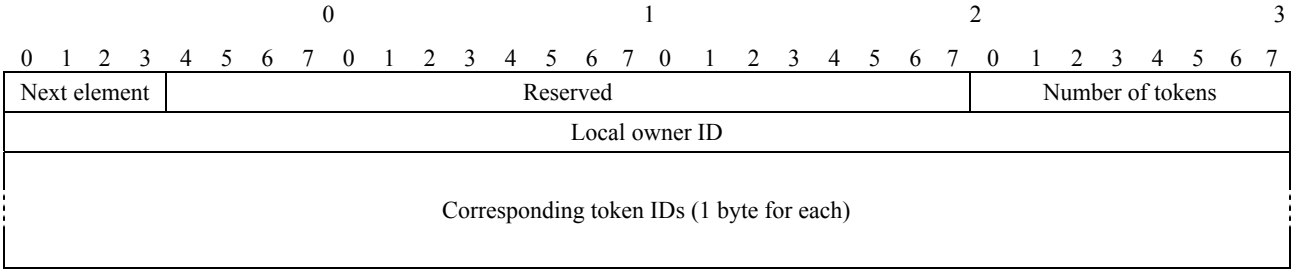

Figure 18 – LO information extension element

Each field is specified as follows:

Next element (4 bits)

This indicates the type of the next extension element, as indicated in Table 1.

Reserved (20 bits)

Number of tokens (8 bits)

This specifies the total number of *Corresponding Token IDs* for the LO whose ID is *Local Owner ID*.

Local owner ID (32 bits)

This indicates the ID of the LO which the SUs of the *Corresponding Token IDs* join. This ID can be assigned in an application-specific purpose.

Corresponding token IDs (varied)

This contains the list of the Token IDs corresponding to the *Local Owner ID* in the connection. Each *Token ID* is of 1-byte length.

8.2.6 Negative acknowledgement element

This extension element provides information on a repair request for the lost DT packets at a child node. If an LE or an LO detects one or more DT packet losses, it requests repair of the lost packet(s) to its parent LE or LO along the control tree. In the case that multiple consecutive packets are detected as lost, LE or LO can request to repair a block of lost packets with a single NACK packet which contains the starting sequence number and the number of consecutively lost packets.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

This extension header is attached to the NACK packet. It is encoded as '1000' in the Next Element field of the preceding element or based header. This element consists of the fixed 8 bytes as depicted in Figure 19:

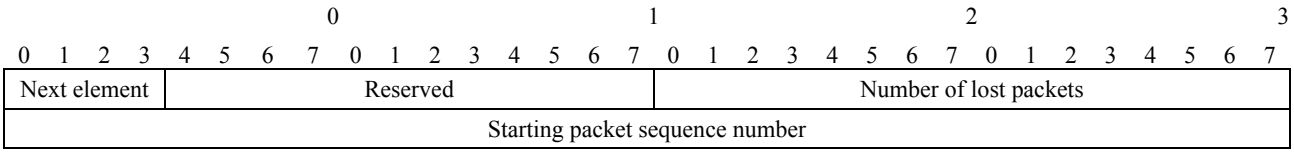

Figure 19 – Negative acknowledgement extension element

Each field is specified as follows:

Next element (4 bits)

This indicates the type of the next extension element, as indicated in Table 1.

Reserved (12 bits)

Number of lost packets (16 bits)

This specifies the number of consecutively lost packets from *Starting Packet Sequence Number*. If it is a request for a single lost packet, this field is set to '1'.

Starting packet sequence number (32 bits)

This specifies the starting packet sequence number of a block of lost packets. If it is a request for a single lost packet, this field is set to PSN of the lost packet.

A NACK packet is generated for the data packets of one SU. The Token ID field in the base header of the NACK packet identifies the associated sender that sent the lost packets.

8.2.7 Tree change information element

This extension element provides information about an intra-group tree change. It represents a change in intra-group tree with respect to the sender and a node with the *Node ID*. This extension header is attached to the TCR, TDR, TNR and CCR packets. It is encoded as '1001' in the Next Element field of the preceding element or base header. This element consists of the fixed 8 bytes.

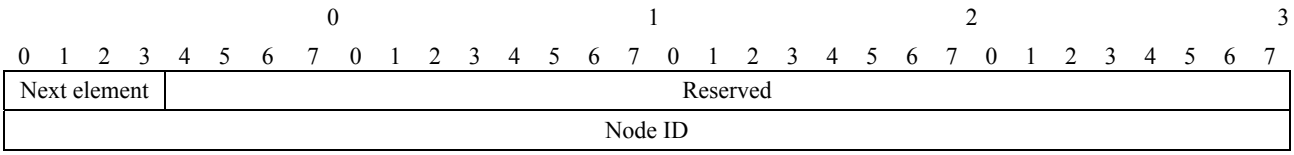

Figure 20 – Tree change information extension element

Each field is specified as follows:

Next element (4 bits)

This indicates the type of the next extension element, as indicated in Table 1.

Reserved (28 bits)

Node ID (32 bits)

This specifies the ID of a node for which intra-group tree should be changed. The context of the change is dependent on packet types.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

8.3 Packet format

In this Recommendation | International Standard, there are a total of 30 packet types: 2 types of data packets and 28 types of control packets. The data packets are DT and RD.

Table 2 – ECTP-5 packets

Full name	Acronym	Transport	From	To
Connection Creation Request	CR	Multicast	TCN	TS-users
Connection Creation Confirm	CC	Unicast	TS-user	TCN
Tree Join Request	TJ	Unicast	LE/LO	LE/LO
Tree Join Confirm	TC	Unicast	LE/LO	LE/LO
Tree Leave Request	TLR	Unicast	LE/LO	LE/LO
Tree Leave Confirm	TLC	Unicast	LE/LO	LE/LO
Data	DT	Multicast	TCN/SU/LO	TS-users
Retransmission Data	RD	Unicast	LE/LO	LE/LO
Acknowledgement	ACK	Unicast	LE/LO	LE/LO
Negative Acknowledgement	NACK	Unicast	LE/LO	LE/LO
Probe	PB	Unicast	TCN	TS-user
Probe Acknowledgement	PBACK	Unicast	TS-user	TCN
Late Join Request	JR	Unicast	TS-user	TCN
Late Join Confirm	JC	Unicast	TCN	TS-user
User Leave Request	LR	Unicast	TCN TS-user	TS-user TCN
Connection Termination Request	CT	Multicast	TCN	TS-users
Token Get Request	TGR	Unicast	SU TCN	TCN SU
Token Get Confirm	TGC	Unicast	TCN SU	SU TCN
Token Return Request	TRR	Unicast	TS-user TCN	TCN TS-user
Token Return Confirm	TRC	Unicast	TCN TS-user	TS-user TCN
Token Status Report	TSR	Multicast Unicast	TCN TCN	TS-users TS-user
Token Status Report Request	TSRR	Unicast	TS-user	TCN
Tree Change Request	TCR	Unicast	LO/LE	LE
Tree Change Confirm	TCC	Unicast	LE	LO/LE
Tree Delegation Request	TDR	Unicast	LO/LE	LO/LE
Tree Delegation Confirm	TDC	Unicast	LO/LE	LO/LE
Tree Change Notification Request	TNR	Unicast	LE	LO
Tree Change Notification Confirm	TNC	Unicast	LO	LE
Control Tree Change Request	CCR	Unicast	LO	LE
Control Tree Change Confirm	CCC	Unicast	LE	LO

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

The encoding value and packet structure are shown for each of the ECTP-5 packets in Table 3. The extension elements are attached to the base header in the order specified in the table.

Table 3 – Format of ECTP-5 packets

Packet type	Encoding value	Extension elements or user data (packet structure)	Length (bytes)	Operational protocol stage
CR	0000 0001	Connection	20	Connection Creation
CC	0000 0010		16	
TJ	0000 0011	Timestamp	28	Logical Tree Management
TC	0000 0100	Timestamp	28	
TLR	0010 0011		16	
TLC	0010 0100		16	
DT	0000 0101	User Data	16+	Data Transport
RD	0000 0111	Timestamp + *User Data*	28+	Error Control
ACK	0000 1000	*Error Bitmap*	16+	
NACK	0001 1000	Negative Acknowledgement + Timestamp	36	
PB	0000 1001		16	Membership Management
PBACK	0000 1110		16	
JR	0000 1010		16	Late Join
JC	0000 1011	Connection	20	
LR	0000 1100		16	User Leave
CT	0000 1101		16	Termination
TGR	0001 0001	LO Information	16+	Token Get
TGC	0001 0010	LO Information	16+	
TRR	0001 0011		16	Token Return
TRC	0001 0100		16	
TSR	0001 0101	*Token + LO Information*	16+	Token Report
TSRR	0010 0101		16	
TCR	0001 0110	Tree Change Information	24	Logical Tree Adaptation
TCC	0001 0111		16	
TDR	0001 1110	Tree Change Information + Error Bitmap	24+	
TDC	0001 1111		16	
TNR	0010 0001	Tree Change Information	24	Control Tree Management
TNC	0010 0010		16	
CCR	0010 1000	Tree Change Information	24	
CCC	0010 1001		16	

It is noted that the italic parts (of the extension elements) indicate that the use of the corresponding extension element is optional, rather than mandatory, in the implementation. In the packet length column of the table, '+' sign signifies that the packet size may get larger by adding the specified optional element or user data.

On the other hand, the following encoding values reserved for future use: '0000 0000', '0000 1111', '0001 0000', '0010 0000', and '0000 0110'.

8.3.1 Connection creation request (CR)

The CR packet is used by TCN so as to create an N-plex multicast connection. TCN sends the CR packet to the group with the following source and destination addresses:

- Source IP: IP address of TCN.
- Source port: Local port number of TCN.
- Destination IP: Multicast IP address of the group.
- Destination port: Group port number.

The length of the CR packet is 20 bytes (16-byte base header + 4-byte Connection element).

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

The CR packet is formatted as shown in Figure 21:

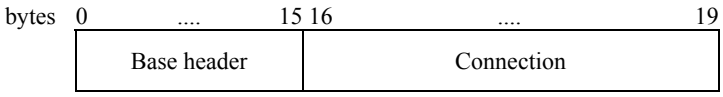

Figure 21 – CR packet

The 16-byte base header of CR packet must be encoded as follows:

Next element: '0001' (Connection element).

Version: '00' (current version of ECTP-5).

CT: '11' (N-plex multicast connection).

Packet type: '0000 0001' (CR).

Checksum: To be calculated.

Source port: Local port number of TCN (or Connection ID).

Destination port: Group port number (or Connection ID).

Payload length: '4'.

F: '0' (to be ignored).

Token ID: '0' (Token ID of TCN is set to '0').

All the fields other than specified above will be set to '0' and ignored at the receiver side.

The 4-byte Connection element must be encoded as follows:

Next element: '0000'.

TCO: As configured by TCN.

AGN: As configured by TCN.

MSS: As configured by TCN (the default value is '1024').

8.3.2 Connection creation confirm (CC)

The CC packet is used by TS-user in response to the CR packet of TCN. A TS-user sends the CC packet to TCN with the following source and destination addresses:

- Source IP: IP address of TS-user.
- Source port: Local port number of TS-user.
- Destination IP: IP address of TCN.
- Destination port: Group port number.

The CC packet contains the 16-byte base header only. The base header of CC packet must be encoded as follows:

Next element: '0000'.

CT: '11'.

Packet type: '0000 0010' (CC).

Checksum: To be calculated.

Source port: Local port number of TS-user (or Connection ID).

Destination port: Group port number (or Connection ID).

All the fields other than specified above will be set to '0' and ignored at the receiver side.

8.3.3 Tree join request (TJ)

The TJ packet is sent by LE or LO to the LO or LE in order to join an intra-group tree or an inter-group tree. This packet is addressed as following:

- Source IP: IP address of LE or LO.
- Source port: Local port number of LE or LO.
- Destination IP: IP address of the LO or LE.

--`,,```,,,,````-`-`,,`,,`,`,,`---

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

- Destination port: Group port number.

The TJ packet contains the 16-byte base header and the 12-byte Timestamp element. The TJ packet is formatted as shown in Figure 22:

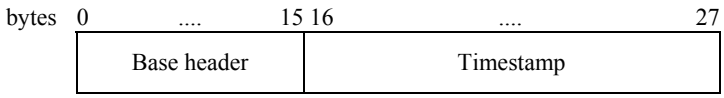

Figure 22 – TJ packet

The base header of TJ packet must be encoded as follows:

Next element: '0100' (Timestamp Element).

CT: '11'.

Packet type: '0000 0011' (TJ).

Checksum: To be calculated.

Source port: Local port number of LE or LO (or Connection ID).

Destination port: Group port number (or Connection ID).

PSN: A sequence number of this packet.

Payload length: '12'.

F: '1' (for inter-group tree join), '0' (for intra-group tree join).

All the fields other than specified above will be set to '0' and ignored at the receiver side.

The 12-byte Timestamp element must be encoded as follows:

Next element: '0000'.

Timestamp: Current time of the packet sender.

8.3.4 Tree join confirm (TC)

The TC packet is sent by the LO or LE node, in response to the TJ packet. The LO or LE sends the TC packet to the requester of tree join (LE or LO) with following source and destination addresses:

- Source IP: IP address of the LO or LE.
- Source port: Group port number.
- Destination IP: IP address of the LE or LO.
- Destination port: Local port of the LE or LO.

The TC packet contains the 16-byte base header, and 12-byte Timestamp element, as shown in Figure 23.

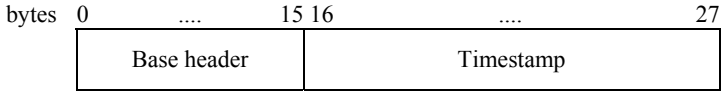

Figure 23 – TC packet

The 16-byte base header of TC packet must be encoded as follows:

Next element: '0100' (Timestamp Element).

CT: '11'.

Packet type: '0000 0100' (TC).

Checksum: To be calculated.

Source port: Group port (or Connection ID).

Destination port: Local port of LE or LO (or Connection ID).

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

PSN: The value copied from *PSN* field of the corresponding request packet..

Payload length: '12'.

F: '1' if the TJ request is accepted, '0' otherwise.

All the fields other than specified above will be set to '0' and ignored at the receiver side.

The 12-byte Timestamp element must be encoded as follows:

Next element: '0000'.

Timestamp: 8-byte time value copied from the corresponding TJ packet.

8.3.5 Tree leave request (TLR)

The TLR packet is sent by LE or LO to the parent LO or LE in order to leave the logical tree. The TLR packet is sent with the following source and destination addresses:

- Source IP: IP address of LE or LO.
- Source port: Local port of LE or LO.
- Destination IP: IP address of LO or LE.
- Destination port: Group port number.

The TLR packet contains the 16-byte base header only. The 16-byte base header of TLR packet must be encoded as follows:

Next element: '0000'.

CT: '11'.

Packet type: '0010 0011' (TLR).

Checksum: To be calculated.

Source port: Local port of LE or LO (or Connection ID).

Destination port: Group port (or Connection ID).

PSN: A sequence number of this packet.

F: '1' (for inter-group tree join), '0' (for intra-group tree join).

All the fields other than specified above will be set to '0' and must be ignored at the receiver side.

8.3.6 Tree leave confirm (TLC)

The TLC packet is sent by the LO or LE node, in response to the TLR packet. The LO or LE sends the TLC packet to the child LE or LO with the following source and destination addresses:

- Source IP: IP address of LO or LE.
- Source port: Group port number.
- Destination IP: IP address of the LE or LO.
- Destination port: Local port of the LE or LO.

The TLC packet contains the 16-byte base header only. The base header of TLC packet must be encoded as follows:

Next element: '0000'.

CT: '11'.

Packet type: '0010 0100' (TLC).

Checksum: To be calculated.

Source port: Group port number (or Connection ID).

Destination port: Local port number of LE or LO (or Connection ID).

PSN: The value copied from PSN field of the corresponding request packet.

F: '1' if the TLC request is accepted, '0' otherwise.

All the fields other than specified above will be set to '0' and must be ignored at the receiver side.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

8.3.7 Data (DT)

The DT packet is used by TCN or SU to transmit the multicast data to the group members. The DT packet is also used by LO to transmit test traffic for logical tree adaptation. The multicast DT packets are sent to the TS-users with the following source and destination addresses:

- Source IP: IP address of TCN or SU or LO.
- Source port: Local port number of TCN or SU or LO.
- Destination IP: Multicast IP address of the group.
- Destination port: Group port number.

The DT packet contains the 16-byte base header and the variable-length user data as shown in Figure 24:

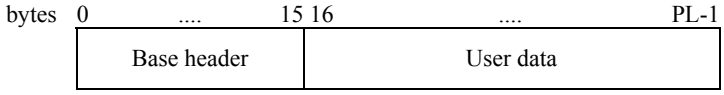

Figure 24 – DT packet

The 16-byte base header of DT packet must be encoded as follows:

Next element: '0000'.

CT: '11'.

Packet type: '0000 0101' (DT).

Checksum: To be calculated.

Source port: Local port number of TCN or SU or LO (or Connection ID).

Destination port: Group port number (or Connection ID).

PSN: The *PSN* of this DT packet, which starts with the initial sequence number and increases by '1', and wraps back around to '1' after reaching '$2^{32} - 1$'.

Payload length: Indicates the length (in byte) of the user data contained in this packet.

F: '1' (for LO's test traffic), '0' otherwise.

Token ID: Token ID of the sender of this data packet ('0' for TCN, or a positive number of SU).

8.3.8 Retransmission data (RD)

The RD packet is sent by LO or LE to retransmit data in response to repair requests (NACK) from child nodes (LE or LO) along the control tree. The packet format is the same as that of the DT packet. In contrast to the DT packet delivery via multicast, however, RD is sent to the requester by unicast with the following source and destination addresses:

- Source IP: IP address of LO or LE.
- Source port: Local port number of LO or LE.
- Destination IP: IP address of the repair requester (LE or LO).
- Destination port: Group port number.

The RD packet contains the 16-byte base header and the variable-length user data as shown in Figure 25:

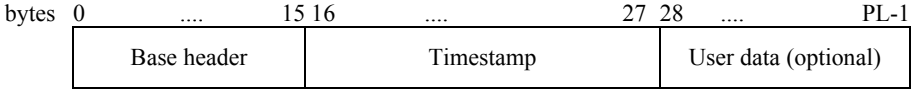

Figure 25 – RD packet

The 16-byte base header of RD packet must be encoded as follows:

Next element: '0100' (Timestamp element).

CT: '11'.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Packet type: '0000 0111' (RD).

Checksum: To be calculated.

Source port: Local port number of LO or LE (or Connection ID).

Destination port: Group port number (or Connection ID).

PSN: The *PSN* of the data packet, which is requested to be retransmitted.

Payload length: Indicates the length (in byte) of the user data contained in this packet.

F: '1' if the sender of this packet has no data to repair, '0' otherwise.

Token ID: Token ID of the sender for this data packet ('0' for TCN, or a positive number of SU).

The 12-byte Timestamp element must be encoded as follows:

Next element: '0000'.

Timestamp: 8-byte time value copied from the corresponding NACK packet.

8.3.9 Acknowledgement (ACK)

The ACK packet is sent to the parent node along the control tree in order to acknowledge the DT packets received from an SU. An ACK packet is generated by the ACK generation rule, which will be described later. If the ACK packet is for LO's test traffic as described in 7.5, the ACK packet should contain an error bitmap element for Logical Tree Adaptation (LTA).

Over the unicast control channel, an ACK packet is sent by an LE or LO to its parent LE or LO along the control tree whose root is the SU of the corresponding data. It is transmitted with the following source and destination addresses:

- Source IP: IP address of LE or LO.
- Source port: Local port number of LE or LO.
- Destination IP: IP address of the parent node (LO or LE).
- Destination port: Group port number.

The ACK packet contains the 16-byte base header, and optionally an Error Bitmap element with variable length as shown in Figure 26:

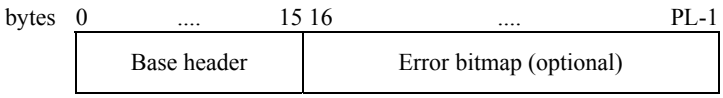

Figure 26 – ACK packet

The base header of ACK packet must be encoded as follows:

Next element: '0000' or '0010' (Error Bitmap element) for ACK packet in response to test traffic from LO.

CT: '11'.

Packet type: '0000 1000' (ACK).

Checksum: To be calculated.

Source port: Local port number of LE or LO (or Connection ID).

Destination port: Group port number (or Connection ID).

PSN: *LSN*, the *PSN* of the lowest numbered DT packet that has not been received yet.

Payload length: Length (in byte) of the extension elements attached to the base header.

F: '0' (to be ignored).

Token ID: Token ID of the corresponding sender ('0' for TCN, or a positive number of SU).

The error bitmap element must be encoded as follows:

Next element: '0000'.

Bitmap length: Represents the total length of the *Error Bitmap* in word (in 4-byte).

Valid bitmap length: The actually valid length of the *Error Bitmap* in bit.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Error bitmap: Represents the bitmap information about which DT packets are lost.

8.3.10 Negative acknowledgement (NACK)

The NACK packet is to request a parent to retransmit data that are detected as lost. A NACK packet is sent immediately at packet loss detection. With the *Number of Lost Packets* and *Starting Packet Sequence Number* fields in the Negative Acknowledgement element, a NACK packet may indicate a repair request for a block of two or more consecutively lost packets.

Over the unicast control channel, a NACK packet is sent by an LE or LO to its parent node along the control tree whose root is the SU of the corresponding data. It is transmitted with the following source and destination addresses:

- Source IP: IP address of LE or LO.
- Source port: Local port number of LE or LO.
- Destination IP: IP address of the parent node (LO or LE).
- Destination port: Group port number.

The NACK packet contains the 16-byte base header, 8-byte Negative Acknowledgement element, and 12-byte Timestamp element, as shown in Figure 27.

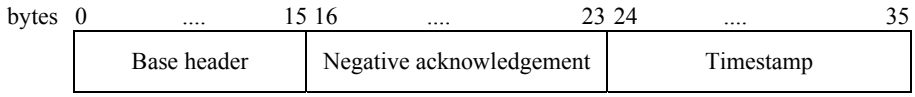

Figure 27 – NACK packet

The base header of NACK packet must be encoded as follows:

Next element: '1000' (Negative Acknowledgement).

CT: '11'.

Packet type: '0001 1000' (NACK).

Checksum: To be calculated.

Source port: Local port number of LE or LO (or Connection ID).

Destination port: Group port number (or Connection ID).

PSN: *LSN* that is the *PSN* of the lowest numbered DT packet that has not been received yet

Payload length: '20'.

F: '0' (to be ignored).

Token ID: Token ID of the corresponding sender.

The Negative Acknowledgement element must be encoded as follows:

Next element: '0100' (Timestamp element).

Number of lost packets: Represents the number of consecutively lost packets from *Starting Packet Sequence Number*.

Starting packet sequence number: The starting packet sequence number of a block of lost packets.

The 12-byte Timestamp element must be encoded as follows:

Next element: '0000'.

Timestamp: Current time of the packet sender.

8.3.11 Probe (PB)

The PB packet is used by TCN for connection maintenance. The TCN sends the periodic PB packet to a selected TS-user in the session with the following source and destination addresses:

- Source IP: IP address of TCN.
- Source port: Local port number of TCN.
- Destination IP: IP address of the selected TS-user.
- Destination port: Group port number.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

The PB packet contains the 16-byte base header only. The base header of the PB packet is formatted as follows:

Next element: '0000'.

CT: '11'.

Packet type: '0000 1001' (PB).

Checksum: To be calculated.

Source port: Local port number of TCN (or Connection ID).

Destination port: Group port number (or Connection ID).

All the fields other than specified above will be set to '0' and ignored at the receiver side.

8.3.12 Probe acknowledgement (PBACK)

The PBACK packet is sent by a TS-user, in response to the PB packet of TCN. When a TS-user receives the PB packet from TCN, it should respond with the PBACK packet. This packet is used to indicate that it is still alive.

The TS-user sends the PBACK packet to TCN with the following source and destination addresses:

- Source IP: IP address of TS-user.
- Source port: Local port number of TS-user.
- Destination IP: IP address of TCN.
- Destination port: Local port number of TCN.

The PBACK packet contains the 16-byte base header only. The base header of the PBACK packet is formatted as follows:

Next element: '0000'.

CT: '11'.

Packet type: '0000 1110' (PBACK).

Checksum: To be calculated.

Source port: Local port number of TS-user (or Connection ID).

Destination port: Local port number of TCN (or Connection ID).

All the fields other than specified above will be set to '0' and ignored at the receiver side.

8.3.13 Late join request (JR)

The JR packet is used by a new joining TS-user in order to join the ECTP connection. The new joining TS-user sends the JR packet to the TCN with the following source and destination addresses:

- Source IP: IP address of TS-user.
- Source port: Local port number of TS-user.
- Destination IP: IP address of the TCN.
- Destination port: Group port number.

The JR packet contains the 16-byte base header only. The base header of JR packet must be encoded as follows:

Next element: '0000'.

CT: '11'.

Packet type: '0000 1010' (JR).

Checksum: To be calculated.

Source port: Local port number of TS-user (or Connection ID).

Destination port: Group port number (or Connection ID).

PSN: A sequence number of this packet.

All the fields other than specified above will be set to '0' and ignored at the receiver side.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

8.3.14 Late join confirm (JC)

The JC packet is used by TCN, in response to the JR packet. The TCN sends the JC packet to the new joining TS-user with the following source and destination addresses:

- Source IP: IP address of the TCN.
- Source port: Local port number of TCN.
- Destination IP: IP address of the TS-user.
- Destination port: Local port of the TS-user.

The JC packet contains the 16-byte base header and the 4-byte Connection element.

The JC packet is formatted as shown in Figure 28:

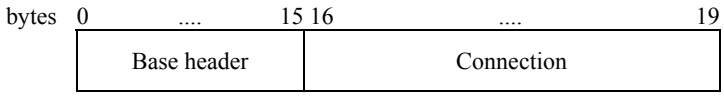

Figure 28 – JC packet

The 16-byte base header of JC packet must be encoded as follows:

Next element: '0001' (Connection element).

CT: '11'.

Packet type: '0000 1011' (JC).

Checksum: To be calculated.

Source port: Local port of TCN (or Connection ID).

Destination port: Local port of TS-user (or Connection ID).

PSN: The value copied from PSN field of the corresponding request packet.

Payload length: '4'.

F: '1' if the JR request is accepted, '0' otherwise.

All the fields other than specified above will be set to '0' and ignored at the receiver side.

The 4-byte Connection element must be encoded as follows:

Next element: '0000'.

TCO: As configured by TCN.

AGN: As configured by TCN.

MSS: As configured by TCN.

8.3.15 User leave request (LR)

The LR packet is used by TS-user to indicate that it will leave the connection, or by TCN to eject a trouble-making TS-user. The LR packet does not need to require the corresponding confirm packet. This packet is sent with the following source and destination addresses:

- Source IP: IP address of TS-user (user leave) or TCN (troublemaker ejection).
- Source port: Local port number of TS-user or TCN.
- Destination IP: IP address of TCN or TS-user.
- Destination port: Group port number or local port number of TS-user.

The LR packet contains the 16-byte base header only. The base header is formatted as follows:

Next element: '0000'.

CT: '11'.

Packet type: '0000 1100' (LR).

Checksum: To be calculated.

Source port: Local port of TS-user or TCN (or Connection ID).

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

Destination port: Group port or local port of TS-user (or Connection ID).

F: '1' for the user-invoked leave, or '0' for the troublemaker ejection.

All the fields other than specified above will be set to '0' and ignored at the receiver side.

8.3.16 Connection termination request (CT)

The CT packet is used by the TCN to terminate the connection. CT packet does not need to require the corresponding confirm packet. This packet is sent by TCN with the following source and destination addresses:

- Source IP: IP address of the TCN.
- Source port: Local port number of TCN.
- Destination IP: Multicast IP address of the group.
- Destination port: Group port number.

The CT packet contains the 16-byte base header only. The base header is formatted as follows:

Next element: '0000'.

CT: '11'.

Packet type: '0000 1101' (CT).

Checksum: To be calculated.

Source port: Local port of the TCN (or Connection ID).

Destination port: Group port (or Connection ID).

F: '1' for an abnormal termination, or '0' for the normal termination (after completing the multicast data transmission).

All the fields other than specified above will be set to '0' and ignored at the receiver side.

8.3.17 Token get request (TGR)

The TGR packet is used by a TS-user to get a token for the multicast data transport (TS-user-initiated Token Get). In this case, the TS-user can request a token to TCN by sending a TGR packet. The TS-user that has a token becomes an SU.

In the TCN-initiated Token Give operation, TCN requests a TS-user to be an SU. In this case, TCN will send a TGR packet to a certain TS-user.

In the case of Token Get, the *LO Information* element should be attached to TGR packet by TS-user. It provides TCN with information about the LO that the TS-user joins. This information is used when sending TSR packets.

The TGR packet is sent with the following source and destination addresses:

- Source IP: IP address of TS-user (Token Get) or IP address of TCN (Token Give).
- Source port: Local port number of TS-user (Token Get) or local port number of TCN (Token Give).
- Destination IP: IP address of the TCN (Token Get) or IP address of TS-user (Token Give).
- Destination port: Group port number (Token Get) or local port number of TS-user (Token Give).

The TGR packet contains the 16-byte base header and the variable size LO Information element.

The TGR packet is formatted as shown in Figure 29:

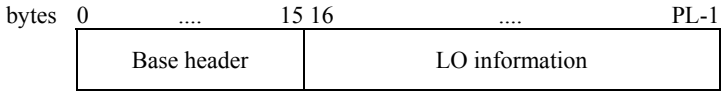

Figure 29 – TGR packet

The 16-byte base header of TGR packet must be encoded as follows:

Next element: '0000' or '0111' (LO Information element, in the case of Token Get).

CT: '11'.

Packet type: '0001 0001' (TGR).

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Checksum: To be calculated.

Source port: Local port number of TS-user or TCN (or Connection ID).

Destination port: Group port number or local port of TS-user (or Connection ID).

PSN: A sequence number of this packet.

Payload length: Length (in byte) of the extension elements attached to the base header.

F: To be ignored.

Token ID: Token ID allocated by TCN (Token Give); in the Token Get case, this field is ignored.

If the LO Information element is added for the next element in the case of Token Get, it must be encoded as follows:

Next element: '0000'.

Number of tokens: '1'.

Local owner ID: Represents the ID of the LO which the LE joins.

All the fields other than specified above will be set to '0' and must be ignored at the receiver side.

8.3.18 Token get confirm (TGC)

In response to a TGR, TCN sends a TGC packet to the TS-user that sent the TGR (TS-user-initiated Token Get). In the TCN-initiated Token Give operation, the TGC packet is used by TS-user to confirm the request.

In the case of Token Give, the *LO Information* element should be attached to TGC packet by TS-user. It provides TCN with information about the LO that the LE joins. This information is used when sending TSR packets.

The TGC packet is sent with the following source and destination addresses:

- Source IP: IP address of TCN (Token Get) or IP address of LE (Token Give).
- Source port: Group port number (Token Get) or local port number of LE (Token Give).
- Destination IP: IP address of the LE (Token Get) or IP address of TCN (Token Give).
- Destination port: Local port of LE (Token Get) or local port of TCN (Token Give).

The TGC packet contains the 16-byte base header and the variable size LO Information element.

The TGC packet is formatted as shown in Figure 30:

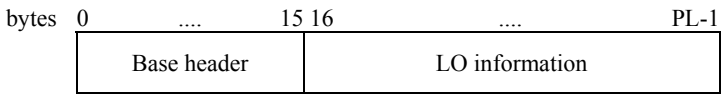

Figure 30 – TGC packet

The 16-byte base header of TGC packet must be encoded as follows:

Next element: '0000' or '0111' (LO Information element, in the case of Token Give).

CT: '11'.

Packet type: '0001 0010' (TGC).

Checksum: To be calculated.

Source port: Group port or local port number of LE (or Connection ID).

Destination port: Group port number or local port of LE (or Connection ID).

PSN: The value copied from PSN field of the corresponding request packet.

Payload length: Length (in byte) of the extension elements attached to the base header.

F: '0' (for acceptance) or '1' (for rejection).

Token ID: Token ID allocated by TCN (Token Get); in the Token Give case, this field is ignored.

If the LO Information element is added as the next element in the case of Token Give, it must be encoded as follows:

Next element: '0000'.

Number of tokens: '1'.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

Local owner ID: Represents the ID of the LO that the LE joins.

All the fields other than specified above will be set to '0' and must be ignored at the receiver side.

8.3.19 Token return request (TRR)

The TRR packet is used by a TS-user to return a token to TCN (TS-user-initiated Token Return). In this case, the TS-user sends a TRR packet. In TCN-initiated Token Withdrawal case, TCN may first request a TS-user to return the token. In this case, TCN will send a TRR packet to the concerned TS-user.

The TRR packet is sent with the following source and destination addresses:

- Source IP: IP address of TS-user (Token Return) or IP address of TCN (Token Withdrawal).
- Source port: Local port of TS-user (Token Return) or local port of TCN (Token Withdrawal).
- Destination IP: IP address of the TCN (Token Return) or IP address of TS-user (Token Withdrawal).
- Destination port: Group port (Token Return) or local port of TS-user (Token Withdrawal).

The TRR packet contains the 16-byte base header only, which must be encoded as follows:

Next element: '0000'.

CT: '11'.

Packet type: '0001 0011' (TRR).

Checksum: To be calculated.

Source port: Local port number of TS-user or TCN (or Connection ID).

Destination port: Group port number or local port of TS-user (or Connection ID).

PSN: A sequence number of this packet.

Token ID: Token ID of TS-user.

All the fields other than specified above will be set to '0' and ignored at the receiver side.

8.3.20 Token return confirm (TRC)

The TRC packet is used by TCN or TS-user to confirm the associated TRR request. The TRC packet is sent with the following source and destination addresses:

- Source IP: IP address of TCN (Token Return) or IP address of TS-user (Token Withdrawal).
- Source port: Group port (Token Return) or local port of TS-user (Token Withdrawal).
- Destination IP: IP address of the TS-user (Token Return) or IP address of TCN (Token Withdrawal).
- Destination port: Local port of TS-user (Token Return) or local port of TCN (Token Withdrawal).

The TRC packet contains the 16-byte base header only, which must be encoded as follows:

Next element: '0000'.

CT: '11'.

Packet type: '0001 0100' (TRC).

Checksum: To be calculated.

Source port: Group port or local port number of TS-user (or Connection ID).

Destination port: Group port number or local port of TS-user (or Connection ID).

PSN: The value copied from PSN field of the corresponding request packet.

Token ID: Token ID of TS-user.

All the fields other than specified above will be set to '0' and ignored at the receiver side.

8.3.21 Token status report (TSR)

The TSR packet is used by TCN to announce the currently valid Token IDs in the connection. It also provides LOs with the information about corresponding token IDs of each LO by attaching the LO Information element. The TSR packet is sent with the following source and destination addresses:

- Source IP: IP address of TCN.
- Source port: Local port of TCN or group port (replying to TSRR).
- Destination IP: IP multicast address of the group or IP address of the TS-user (replying to TSRR).

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

- Destination port: Group port or local port of TS-user (replying to TSRR).

The TSR packet is formatted as shown in Figure 31:

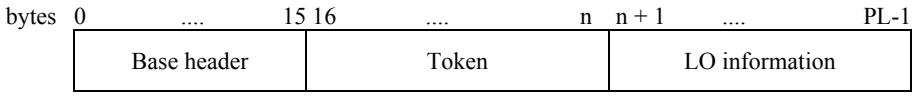

Figure 31 – TSR packet

The TSR packet contains the 16-byte base header, the variable-size Token element, and the variable-size LO Information element. The base header must be encoded as follows:

Next element: '0110' (Token Element).

CT: '11'.

Packet type: '0001 0101' (TSR).

Checksum: To be calculated.

Source port: Local port number of TCN (or Connection ID) or group port (replying to TSRR).

Destination port: Group port number (or Connection ID) or local port of TS-user (replying to TSRR).

Payload length: Length (in byte) of the extension elements attached to the base header.

All the fields other than specified above will be set to '0' and must be ignored at the receiver side.

The variable size Token element must be encoded as follows:

Next element: '0111' (LO Information element).

Number of tokens: Specifies the total number of *Valid Token IDs* in the connection.

Valid token IDs: Contains the list of the Token IDs valid in the connection. Each *Token ID* is of 1-byte length.

All the fields other than specified above will be set to '0' and must be ignored at the receiver side.

The variable size LO Information element must be encoded as follows:

Next element: '0000' or '0111' (If another LO Information element is followed).

Number of tokens: Total number of *Corresponding Token IDs* for the LO whose ID is *Local Owner ID*.

Local owner ID: Represents the ID of the LO that the LE joins.

Corresponding token IDs: Contains the list of the Token IDs corresponding to the *Local Owner ID* in the connection. Each *Token ID* is of 1-byte length.

All the fields other than specified above will be set to '0' and must be ignored at the receiver side.

8.3.22 Token status report request (TSRR)

The TSRR packet is used by TS-user to request TCN to update the token list. A TS-user sends the TSRR packet to TCN with the following source and destination addresses:

- Source IP: IP address of TS-user.
- Source port: Local port number of TS-user.
- Destination IP: IP address of TCN.
- Destination port: Group port number.

The TSRR packet contains the 16-byte base header only. The base header of TSRR packet must be encoded as follows:

Next element: '0000'.

CT: '11'.

Packet type: '0010 0101' (TSRR).

Checksum: To be calculated.

Source port: Local port number of TS-user (or Connection ID).

Destination port: Group port number (or Connection ID).

All the fields other than specified above will be set to '0' and ignored at the receiver side.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

8.3.23 Tree change request (TCR)

The TCR packet is used for changing parent-child relationship of nodes in the intra-group trees. This packet instructs the node receiving this packet to become a child of another node designated by *Node ID*. The TCR packet is sent over unicast control channel with the following source and destination addresses:

- Source IP: IP address of LE or LO.
- Source port: Local port number of LE or LO.
- Destination IP: IP address of the target node (LE).
- Destination port: Group port number.

The TCR packet contains the 16-byte base header and 8-byte Tree Change Information element as shown in Figure 32.

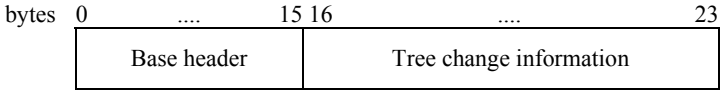

Figure 32 – TCR packet

The base header of TCR packet must be encoded as follows:

Next element: '1001' (Tree Change Information element).

CT: '11'.

Packet type: '0001 0110' (TCR).

Checksum: To be calculated.

Source port: Local port number of LE or LO (or Connection ID).

Destination port: Group port number (or Connection ID).

PSN: A sequence number of this packet.

Payload length: '8'.

All the fields other than specified above will be set to '0' and ignored at the receiver side.

The 8-byte Tree Change Information element must be encoded as follows:

Next element: '0000'.

Node ID: Represents a new parent node of the node which receives this packet.

8.3.24 Tree change confirm (TCC)

The TCC packet is sent by LE as a reply to a TCR packet. The node sends the TCC packet to a node that sent the TCR packet with the following source and destination addresses:

- Source IP: IP address of LE.
- Source port: Group port number.
- Destination IP: IP address of the LE or LO.
- Destination port: Local port of the LE or LO.

The TCC packet contains the 16-byte base header only, which must be encoded as follows:

Next element: '0000'.

CT: '11'.

Packet type: '0001 0111' (TCC).

Checksum: To be calculated.

Source port: Group port number (or Connection ID).

Destination port: Local port number of LE or LO (or Connection ID).

PSN: The value copied from PSN field of the corresponding request packet.

F: Is set to '1' if the TCC request is accepted, '0' otherwise.

All the fields other than specified above will be set to '0' and must be ignored at the receiver side.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

8.3.25 Tree delegation request (TDR)

The TDR packet is used to delegate the process to determine a proper position for a child node to another node in the intra-group trees. The packet includes *Node ID*. The node receiving this packet should find better parent-child relationship for a node with *Node ID* using its available error bitmaps.

Via unicast control channel, a TDR packet is sent by any node that has child nodes to its parent or child nodes in the same intra-group tree. It is transmitted with the following source and destination addresses:

- Source IP: IP address of LE or LO.
- Source port: Local port number of LE or LO.
- Destination IP: IP address of the potential parent node (LO or LE).
- Destination port: Group port number.

The TDR packet contains the 16-byte base header, 8-byte Tree Change Information element, and a variable length of Error Bitmap element, as shown in Figure 33.

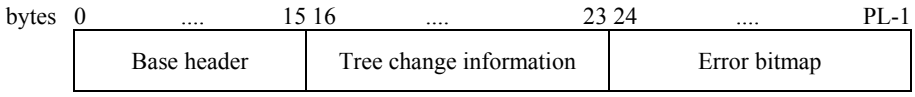

Figure 33 – TDR packet

The base header of TDR packet must be encoded as follows:

Next element: '1001' (Tree Change Information element).

CT: '11'.

Packet type: '0001 1110' (TDR).

Checksum: To be calculated.

Source port: Local port number of LE or LO (or Connection ID).

Destination port: Group port number (or Connection ID).

PSN: A sequence number of this packet.

Payload length: Length (in byte) of the extension elements attached to the base header.

All the fields other than specified above will be set to '0' and ignored at the receiver side.

The 8-byte Tree Change Information element must be encoded as follows:

Next element: '0010' (Error Bitmap element).

Node ID: Represents a potential child of the node which received the TDR packet.

The error bitmap element must be encoded as follows:

Next element: '0000'.

Bitmap length: Represents the total length of the *Error Bitmap* in word (in 4-byte).

Valid bitmap length: The actually valid length of the *Error Bitmap* in bit.

Error bitmap: Represents the bitmap information about which DT packets are lost.

8.3.26 Tree delegation confirm (TDC)

The TDC packet is sent as a reply to a TDR packet. The node sends the TDC packet to a node that sent the TDR packet with the following source and destination addresses:

- Source IP: IP address of LO or LE.
- Source port: Group port number.
- Destination IP: IP address of the LE or LO.
- Destination port: Local port of the LE or LO.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

The TDC packet contains the 16-byte base header only which must be encoded as follows:

Next element: '0000'.

CT: '11'.

Packet type: '0001 1111' (TDC).

Checksum: To be calculated.

Source port: Group port number (or Connection ID).

Destination port: Local port number of LE or LO (or Connection ID).

PSN: The value copied from PSN field of the corresponding request packet.

F: Is set to '1' if the TDC request is accepted, '0' otherwise.

All the fields other than specified above will be set to '0' and must be ignored at the receiver side.

8.3.27 Tree change notification request (TNR)

The TNR packet is used to notify the change of logical tree (intra-group tree) to LO. When receiving this packet, LO should reflect the change to its intra-group tree with respect to the sender and a node with *Node ID*.

Via unicast control channel, a TNR packet is sent by the LE that changes its parent by tree change or tree join. It is also used for a node to prune its child after detecting the child's failure. It is sent with the following source and destination addresses:

- Source IP: IP address of LE.
- Source port: Local port number of LE.
- Destination IP: IP address of the LO.
- Destination port: Group port number.

The TNR packet contains the 16-byte base header and 8-byte Tree Change Information element, as shown in Figure 34.

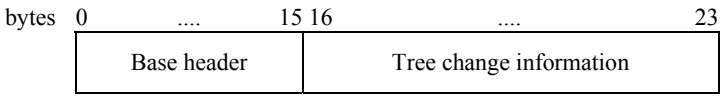

Figure 34 – TNR packet

The base header of TNR packet must be encoded as follows:

Next element: '1001' (Tree Change Information element).

CT: '11'.

Packet type: '0010 0001' (TNR).

Checksum: To be calculated.

Source port: Local port number of LE (or Connection ID).

Destination port: Group port number (or Connection ID).

PSN: A sequence number of this packet.

Payload length: '8'.

F: F is set to '0' if it notifies a change of its parent node, or set to '1' if it notifies a pruning of its child node.

All the fields other than specified above will be set to '0' and ignored at the receiver side.

The 8-byte Tree Change Information element must be encoded as follows:

Next element: '0000'.

Node ID: Its new parent node ID when *F* is '0', or its child node ID which is pruned when *F* is '1'.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

8.3.28 Tree change notification confirm (TNC)

The TNC packet is sent as a reply to a TNR packet. LO sends TNC packet to a node that sent the TNR packet with the following source and destination addresses:

- Source IP: IP address of LO.
- Source port: Group port number.
- Destination IP: IP address of the LE.
- Destination port: Local port of the LE.

The TNC packet contains the 16-byte base header only. The base header of TNC packet must be encoded as follows:

Next element: '0000'.

CT: '11'.

Packet type: '0010 0010' (TNC).

Checksum: To be calculated.

Source port: Group port number (or Connection ID).

Destination port: Local port number of LE (or Connection ID).

PSN: The value copied from PSN field of the corresponding request packet.

All the fields other than specified above will be set to '0' and must be ignored at the receiver side.

8.3.29 Control tree change request (CCR)

The CCR packet is sent by LO in order to change the control tree of LE. Via unicast control channel, a LO sends a CCR packet to an LE that should change its control tree information. It is sent with the following source and destination addresses:

- Source IP: IP address of LO.
- Source port: Local port number of LO.
- Destination IP: IP address of the target node (LE).
- Destination port: Group port number.

The CCR packet contains the 16-byte base header and 8-byte Tree Change Information element, as shown in Figure 35:

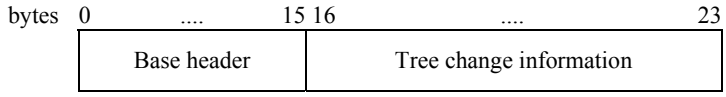

Figure 35 – CCR packet

The base header of CCR packet must be encoded as follows:

Next element: '1001' (Tree Change Information element).

CT: '11'.

Packet type: '0010 1000' (CCR).

Checksum: To be calculated.

Source port: Local port number of LO (or Connection ID).

Destination port: Group port number (or Connection ID).

PSN: A sequence number of this packet.

Payload length: '8'.

Token ID: Token ID of the corresponding SU.

All the fields other than specified above will be set to '0' and ignored at the receiver side.

The 8-byte Tree Change Information element must be encoded as follows:

Next element: '0000'.

Node ID: Represents new parent node in the control tree for the SU specified by *Token ID* in the base header.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

8.3.30 Control tree change confirm (CCC)

The CCC packet is sent as a reply to a CCR packet. An LE sends a CCC packet to LO that sent the CCR packet with the following source and destination addresses:

- Source IP: IP address of LE.
- Source port: Local port number of LE.
- Destination IP: IP address of the LO.
- Destination port: Group port number.

The CCC packet contains the 16-byte base header only. The base header of CCC packet must be encoded as follows:

Next element: '0000'.

CT: '11'.

Packet type: '0010 1001' (CCC).

Checksum: To be calculated.

Source port: Group port number (or Connection ID).

Destination port: Local port number of LO (or Connection ID).

PSN: The value copied from PSN field of the corresponding request packet.

Token ID: Token ID of the corresponding SU.

All the fields other than specified above will be set to '0' and must be ignored at the receiver side.

9 Procedures

This clause describes the protocol procedures of ECTP-5. Before an N-plex multicast connection is created, the following address information should be announced to the prospective participants: TS-users:

a) multicast IP address of the group;

b) group port number;

c) IP address of TCN;

d) IP address of a corresponding LO (to LE only).

This information may be announced to the prospective participants via an out-of-band signalling mechanism such as Web announcement. Accordingly, the prospective TS-user should be able to bind the group IP address and port so as to receive the CR packet from the TCN. A prospective late-joiner TS-user should also send a JR packet to the TCN.

9.1 Connection management

9.1.1 Connection creation

An N-plex multicast connection will begin when TCN is activated to manage session information and tokens.

If TCN is informed of a participant list prior to starting the session, it starts a connection creation phase by sending CR to the group over the multicast group IP address and port.

The overall operations for connection creation are shown in Figure 36:

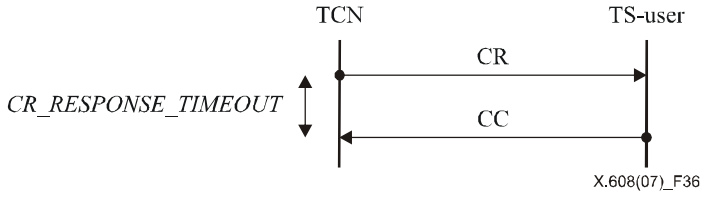

Figure 36 – Connection creation procedures

The CR packet contains the generic information on the connection element such as TCO (Tree Configuration Option), and MSS (Maximum Segment Size).

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

If all of the CC packets do not arrive within the *CR_RESPONSE_TIMEOUT*, TCN sends a CR packet again. This process can be repeated up to *CR_MAX_RETRY* times. If TCN has not received CC packets from all TS-users in the participants list, it gives up the connection creation procedures and terminates the N-plex multicast connection by sending CT (Connection Termination) packet to a group. If there is no predetermined participant prior to starting the session, TCN starts the data transmission phase without connection creation operations.

9.1.2 Late join

Some of the prospective participants may join the N-plex multicast connection as a late joiner. The overall operations for a late joiner are shown in Figure 37:

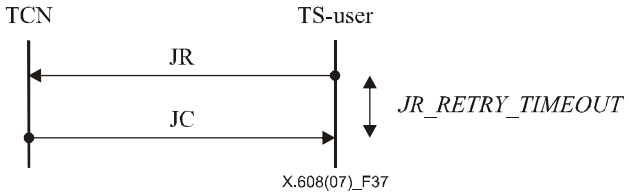

Figure 37 – Late joining procedures

The late joiner TS-user sends a JR packet to TCN. In response to the JR packet, the TCN sends a JC packet to the TS-user. The TJ packet should indicate whether the request is accepted or not by using the *F* flag of the base header.

If the JC packet does not arrive within the *JR_RETRY_TIMEOUT*, the late joiner sends the JR packet again. This process can be repeated up to *JR_MAX_RETRY* times. If the node has not received any JC packet, it gives up the late joining procedures and terminates the N-plex multicast connection.

9.1.3 Connection maintenance

An N-plex multicast connection is maintained using the PB and PBACK packets. The TCN sends periodic PB packets for every *PB_PACKET_INT* to a selected TS-user in the session. The corresponding TS-user should respond with the PBACK packet. The selection method of TS-user for sending PB should be designed to cover every TS-user in the session, e.g., round robin manners.

Figure 38 shows the operation for membership management using PB and PBACK packets.

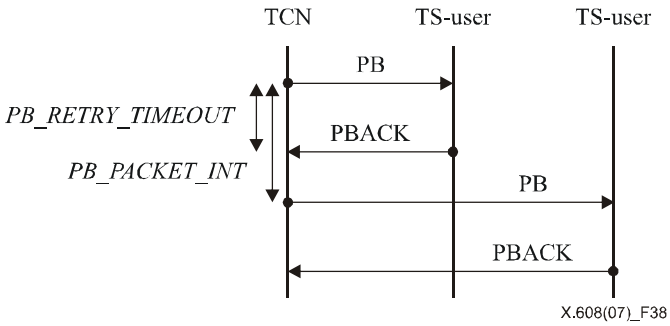

Figure 38 – Connection maintenance using PB and PBACK packets

If the PBACK packet does not arrive within the *PB_RETRY_TIMEOUT*, TCN sends the PB packet again. This process can be repeated up to *PB_MAX_RETRY* times. If TCN has not received any PBACK packet, it enforces the TS-user to leave the connection by sending LR message, which is called the troublemaker ejection.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

9.1.4 User leave

Figure 39 illustrates the operations for user-initiated leave and troublemaker ejection.

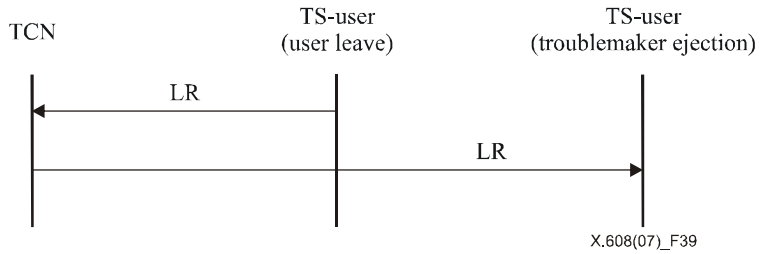

Figure 39 – User leave and troublemaker ejection procedures

In the User Leave case, the TS-user will send an LR message to TCN. In the Troublemaker Ejection case, the TCN will request the concerned TS-user to leave the connection. In both cases, the LR message does not require the corresponding confirm message.

The troublemaker ejection is applied to the TS-user that has not been responding during a certain time interval in the PB and PBACK operation for connection maintenance.

9.1.5 Connection termination

In ECTP-5, the TCN may also terminate the connection when it has concluded to end it. The TCN performs the connection termination by sending a CT message to the group. Figure 40 shows the operations for connection termination. The CT packets do not require any confirm messages.

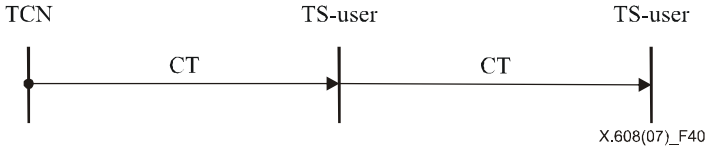

Figure 40 – Connection termination procedures

9.2 Logical tree management

9.2.1 Intra-group tree join

Every LE should join an intra-group tree for error control after join either through initial connection creation or late join.

An LE initiates an intra-group tree join procedure by sending a TJ packet to the corresponding LO. LO then responds with a TC packet. The TC packet should indicate whether the tree join request is accepted or not by using the F flag of the base header. The overall operations for intra-group tree join are shown in Figure 41:

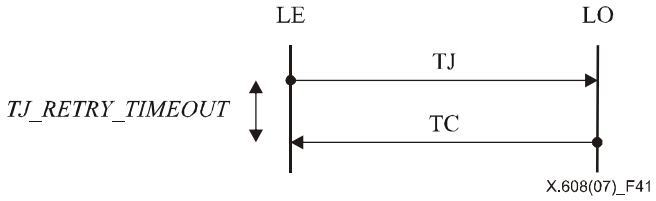

Figure 41 – Intra-group tree join procedures

If no TC packet is received from the LO in response to a TJ within *TJ_RETRY_TIMEOUT*, the node sends TJ packet again. This process can be repeated up to *TJ_MAX_RETRY* times. If the node fails to receive TC packet, it gives up the tree join procedures and returns an error with status information to the application.

This procedure can also be used among LEs at tree change by logical tree adaptation as described in 9.2.4.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

9.2.2 Inter-group tree join

At receiving TSR packets, if an LO sees a new LO that has one or more SUs, it should join an inter-group tree whose root is the new LO.

In order to be an SU, an LE should notify TCN of its LO during Token Get or Token Give procedures. The TCN maintains and provides LOs with this information by sending TSR packets periodically. The TSR packet contains the LO Information element that consists of a list of token IDs in each local group.

Using this information, an LO joins each LO-rooted inter-group tree by sending a TJ packet to every LO that has one or more SUs in its local group.

Each LO that is the root of the inter-group tree then responds with a TC packet. The TC packet should indicate whether the tree join request is accepted or not by using the F flag of the base header. The overall operations for inter-group tree join are shown in Figure 42:

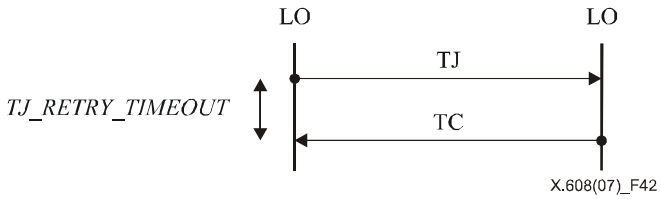

Figure 42 – Inter-group tree join procedures

If the TC packet does not arrive within the *TJ_RETRY_TIMEOUT*, the sender of TJ packet may try to send the TC packet again. This process can be repeated up to *TJ_MAX_RETRY* times. If the node has not received TC packet, it gives up the tree join procedures and terminates the N-plex multicast connection.

9.2.3 Logical tree leave

Before leaving a session (User Leave) or changing a parent node by logical tree adaptation procedure, an LE without child nodes should leave the logical tree (intra-group tree) by sending a TLR packet to its parent node. The parent node then removes the LE from the list of its children and responds with a TLC packet.

The overall operations for intra-group tree leave are shown in Figure 43:

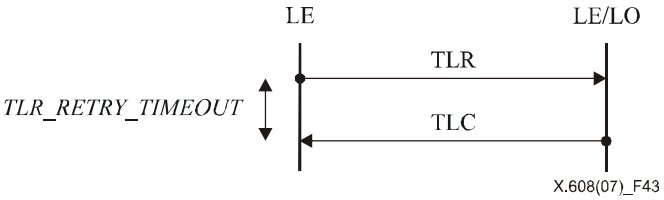

Figure 43 – Intra-group tree leave procedures

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

When an LE with one or more children leaves a session, it should attach its children to its parent by sending a TCR message to its children before leaving. A child node that receives the TCR message replies with a TCC message. In addition to this, the child nodes send TJ messages to their new parent to join as children. After joining the new parent, child nodes send TLR messages to its previous parent. Then the leaving LE replies to each TLR message with TLC messages. After replying to TLR messages from all of its child nodes, the leaving LE now can leave the tree by sending a TLR message to its parent. The overall operations for intra-group tree leave are shown in Figure 44:

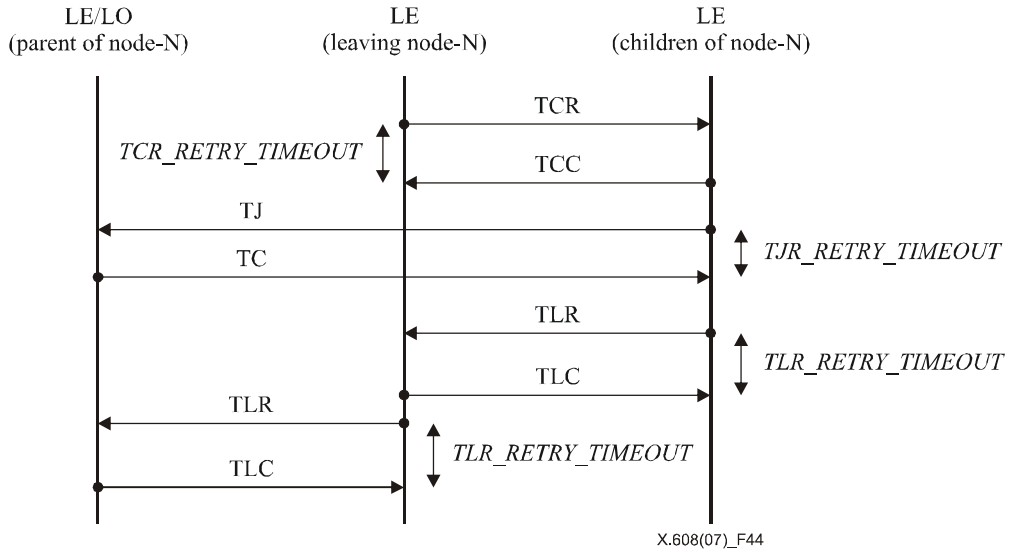

Figure 44 – Intra-group tree leave procedures for node *N* with children

An LO should leave all the inter-group trees if it has no child in its local group and it has no application to consume the session data. If an LO has no SU, all other LOs should prune themselves from the inter-group tree rooted by the LO.

An LO can leave an inter-group tree by sending a TLR packet to its parent LO. The parent LO that receives the TLR packet then removes the LO from the list of its children and responds to the LO with a TLC packet. The overall operations for inter-group tree leave are shown in Figure 45:

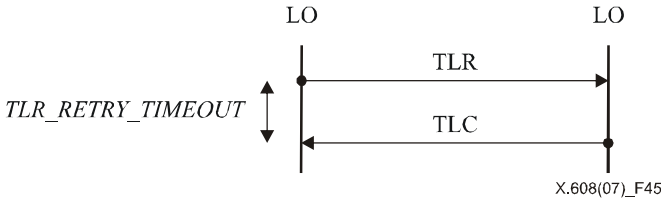

Figure 45 – Inter-group tree leave procedures

If the responding TLC message has not arrived within *TLR_RETRY_TIMEOUT*, the LE or LO may send the TLR message to its parent LE or LO again. This process can be repeated up to *TLR_MAX_RETRY* times. If the node fails to receive TLC packet, it gives up the tree leave procedures and prunes itself from the parent.

9.2.4 Logical tree adaptation

When using TCO of '10', intra-group trees may evolve to multi-level trees close to underlying multicast routing trees by comparing loss patterns of parent and child nodes in the tree. A node can determine relationships between itself and its children by comparing error bitmaps of them.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

In order to describe the logical tree adaptation mechanism, three relational operators between a pair of nodes in an intra-group tree are defined as follows:

$B(N) = B(M)$

Node N and node M have the relationship $B(N) = B(M)$ (which is read as node N is potentially equal to node M) if and only if $B_k(N) = B_k(M)$ for all $k = 1, 2, \ldots, n$. Here $B_k(N)$ denotes the k-th bit (from the left) of the bitmap of node N and $B(N)$ is a string of bits $B_1(N), B_2(N), \ldots, B_n(N)$, while the bitmap length is assumed to be n.

$B(N) \supset B(M)$

Node N and node M have the relationship $B(N) \supset B(M)$ (which is read as node N is a potential parent of node M) if and only if $B_k(N) \geq B_k(M)$ for all $k = 1, 2, \ldots, n$, but not $B(N) = B(M)$.

$B(N) \subset B(M)$

Similarly, $B(N) \subset B(M)$ (which is read as node N is a potential child of node M) if and only if $B_k(N) \leq B_k(M)$ for all $k = 1, 2, \ldots, n$, but not $B(N) = B(M)$.

In Figure 46, ERROR_BITMAP(N) represents a message containing the error bitmap of node N. TDR(N) message is generated by a node that cannot determine the position of node N. This message contains ERROR_BITMAP(N) for further delegations. TCR(N) indicates to the node receiving the message that it should join node N as a child. Parent(N) and child(N) represent parent node and child node of node N, respectively. The following pseudo code describes the logical tree adaptation algorithm.

```
// node D receives a message m;
case (m is ERROR BITMAP(N))
        if (B(D) ⊂ B(N))
                send a TDR(N) message to parent(D);
        else
                if ( ∃ a child C such that B(N) ⊃ B(C))
                        send a TDR(C) message to N;
                else if ( ∃ a child C such that B(C) ⊃ B(N))
                        send a TDR(N) message to C;
                else
                        do nothing;
case (m is TDR(N))
        if (B(D) ⊂ B(N))
                send a TDR(N) message to parent(D)
        else if (B(D) = B(N))
                send a TCR(D) message to N;
        else
                if ( ∃ a child C such that B(N) ⊃ B(C))
                        send a TDR(C) message to N;
                        send a TCR(D) message to N;
                else if ( ∃ a child C such that B(C) ⊃ B(N))
                        send a TDR(N) message to C;
                else
                        send a TCR(D) message to N;
```

Figure 46 – Pseudo code for logical tree adaptation

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

TDR(*N*) message is delivered to the potential parent of node *N*. The same process is repeated upward or downward the intra-group tree until node *N* finds proper position. Figure 47 shows a procedure for delegating a node to the potential parent.

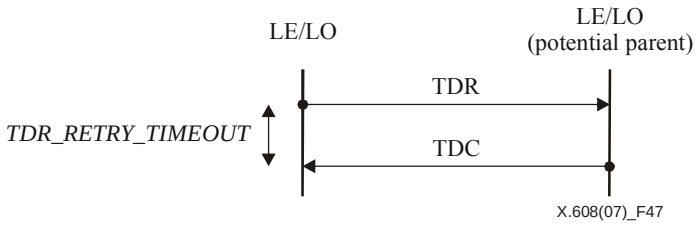

Figure 47 – Tree delegation procedures

If the TDC packet does not arrive within *TDR_RETRY_TIMEOUT*, the sender of TDR packet may try to send the TDR packet again. This process can be repeated up to *TDR_MAX_RETRY* times. If the node fails to receive TDC packet, it gives up the tree delegation procedures.

If a node can determine a proper position of node *N*, it sends a TCR message to the node *N*. Receiving the TCR message, the node *N* replies with a TCC message to confirm the TCR message from its new parent. Node *N* joins the new parent by sending a TJ message. Then, node *N* leaves its old parent. By leaving its old parent after joining a new parent, node *N* can continuously recover lost packets from the old parent in the middle of the delegation and tree change process.

If the TCC packet does not arrive within *TCR_RETRY_TIMEOUT*, the sender of TCR packet may try to send the TCR packet again. This process can be repeated up to *TCR_MAX_RETRY* times. If the node fails to receive TCC packet, it gives up the tree change procedures.

The overall operations for tree change procedures are shown in Figure 48:

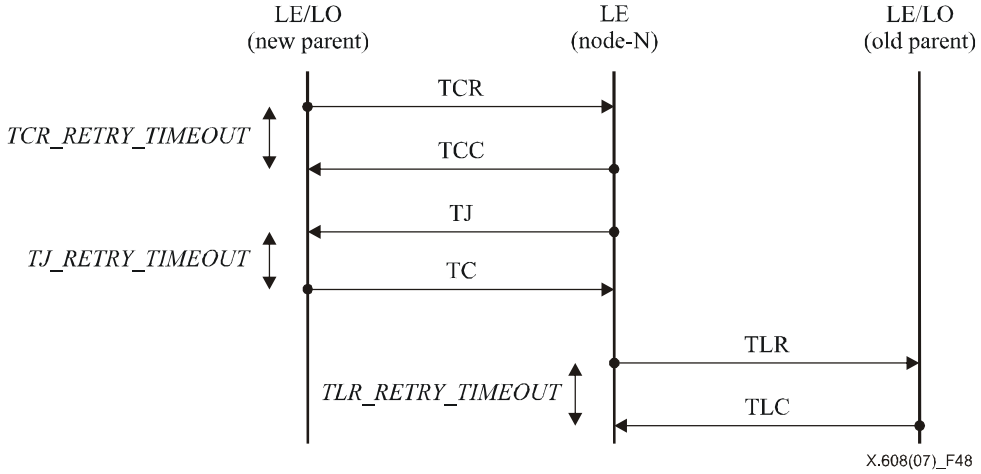

Figure 48 – Tree change procedures

9.2.5 Logical tree change notification

LO should manage its intra-group tree for generating control trees for SUs. Thus, LO should be informed of the change of intra-group tree which has occurred by initial logical tree join procedure, the logical tree adaptation mechanism, or adapting to the node failure.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

TNR is sent to LO by LEs at receiving TJC or detecting a failure of its child. The overall operations for tree change procedures are shown in Figure 49:

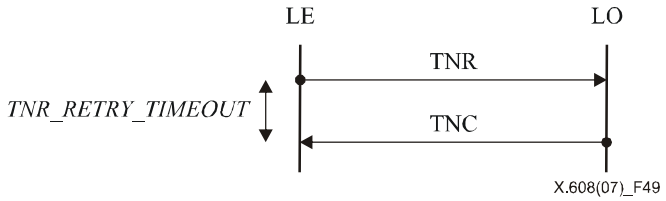

Figure 49 – Tree change notification procedures

If the TNC packet does not arrive within *TNR_RETRY_TIMEOUT*, the sender of TNR packet may try to send the TNR packet again. This process can be repeated up to *TNR_MAX_RETRY* times. If the node fails to receive TNC packet, it gives up the tree change notification procedures, returns an error with status information to the application and terminates the N-plex multicast connection.

At receiving TNR by LO, LO should update its logical tree information. With this concluded logical tree, it examines the control trees to notify each LEs that they should change parent-child relationship in the control tree via CCR.

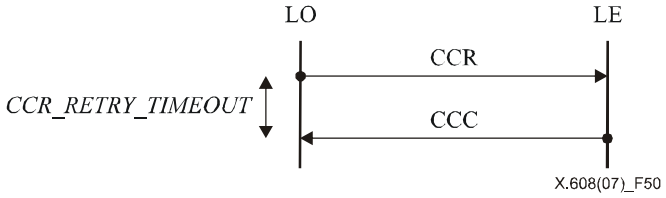

Figure 50 – Control tree change procedures

If the CCR packet does not arrive within *CCR_RETRY_TIMEOUT*, the LO of CCR packet may try to send the CCR packet again. This process can be repeated up to *CCR_MAX_RETRY* times. If the LO has not received CCC packet, it gives up the control tree change procedures and sends LR packet to the corresponding receiver.

9.2.6 Logical tree maintenance

In order to maintain logical tree, TS-users exploit the information obtained from NACK and ACK packets. If a TS-user has not received any RD packets after sending *NACK_MAX_RETRY* NACK packets, it gives up the error recovery procedures and presumes that its parent is failed. Then it tries to find other appropriate parent by contacting LO. A TS-user examines the LSNs of its children, and if the LSN of a child node lags behind its own LSN by *MAX_LSN_LAG*, it presumes that the child is failed. Then it prunes the child from the logical tree, and informs LO.

9.3 Multicast data transport

In the ECTP-5 multicast data channel, the TCN or SU can send multicast DT packets to the group. When a data packet loss is detected by the receiving TS-user, the retransmission for error recovery will be performed by a parent node along the control tree.

9.3.1 Multicast data transmission

TCN or SU will generate DT packets by the segmentation procedure. To do this, the sender splits a multicast data stream of application into multiple DT packets. Each DT packet has its own *Token ID* and sequence number.

Each TS-user delivers all the data packets received to the application in the order sent by TCN. Each receiver reassembles the received packets. Corrupted and lost packets are detected by using a checksum and sequence number. A corrupted packet is also considered as a loss. The lost DT packets are recovered in the error control function.

It is noted that the ACK packets are generated for each SU, which is identified by Token ID.

ECTP-5 uses the flow control based on a fixed-size *window*. The *window size* represents the number of unacknowledged data packets in the sending buffer. The sender can maximally transmit the *window size* of data packets at the configured data transmission rate. In ECTP-5, the transmission rate of multicast data is controlled by the rate-based congestion control mechanisms.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

A new DT packet is sequentially numbered by the multicast sender. The sequence number of the DT packet starts from initial PSN and increases by '1'. The sequence number is used to detect lost data packets by receivers. The initial PSN is randomly generated other than '0'. The sequence number of '0' is reserved. The packet sequence number is increased for each new DT packet. Modulo 2^{32} arithmetic is used and the sequence number wraps back around to '1' after reaching "$2^{32} – 1$".

9.3.2 Reliability control for reliable transport

a) Every TS-user should buffer the data stream from each SU for error recovery of child nodes along the control tree.

b) When a TS-user detects one or more packet losses, it requests the retransmission of the lost packets to its parent node along the control tree rooted by the corresponding SU via a NACK control message.

c) A TS-user should retransmit the data by RD unicast when it received a NACK from a child node along the control tree.

d) A TS-user should acknowledge the DT packets received by sending ACK to the parent node along the control tree.

e) Data in the buffer managed by TS-user can be released when all the child nodes along the control tree acknowledged it by ACK.

f) All SU should buffer a set of its data stream even if all the child nodes have acknowledged it. This is used for error recovery of a TS-user who has failed to recover losses from its parent node along the control tree. The set of the data stream is determined by the semantic specified by application at the upper layer.

9.3.2.1 Error detection

The checksum field of the base header is used for detection of packet corruption, and the *PSN* field is for detection of a packet loss. When a data packet is received, each receiver examines the checksum. If the checksum field is invalid, the packet is regarded as a corruption and shall be discarded. A corruption is treated as a loss. The loss can be detected as a gap of two consecutive sequence numbers for DT packets. NACK packets are used for the retransmission requests at loss detection.

9.3.2.2 Error recovery by NACK and RD unicast

Figure 51 shows the error recovery operations over control channel.

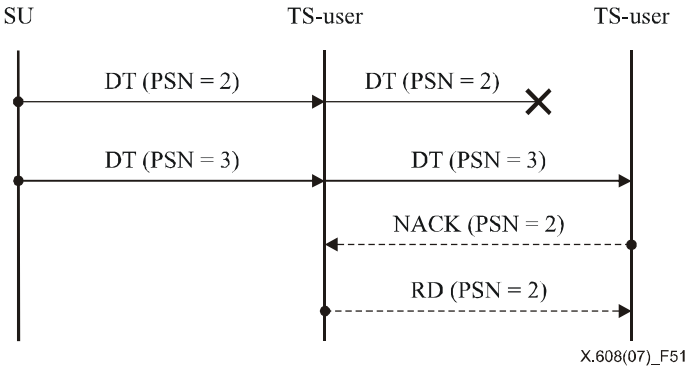

Figure 51 – Error recovery procedures

If a participant detects one or more packet losses, it immediately transmits a NACK packet to the parent along the control tree. In response to the NACK, the parent transmits one or more RD packets to the child via unicast.

If the parent node has received a NACK packet requesting repair for already released data packet, it responds with an RD packet with flag '1'. The child receiving the RD packet with flag '1' sends another NACK packet to corresponding SU. For this, SU should buffer enough amounts of data so that it can repair any losses.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

9.3.2.3 NACK retransmission with a timer

An RD packet may not be delivered to the repair requester due to losses of a NACK packet from the requester or an RD from the parent. Then a NACK packet can be retransmitted as shown in Figure 52:

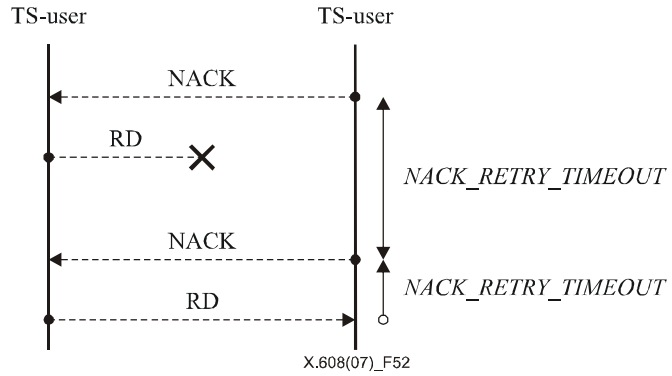

Figure 52 – NACK retransmission procedure

A NACK packet can be retransmitted to the parent if the repair requester did not receive corresponding RD after *NACK_RETRY_TIMEOUT*. The timer with *NACK_RETRY_TIMEOUT* is started when sending every NACK packet, and is cancelled when receiving the corresponding RD packet. The process can be repeated up to *NACK_MAX_RETRY* times. If the node has not received any RD packet, it gives up the error recovery procedures and presumes that its parent is failed. Then it tries to find other appropriate parent by contacting LO.

9.3.2.4 ACK generation

Figure 53 shows the acknowledgement operations over control channel.

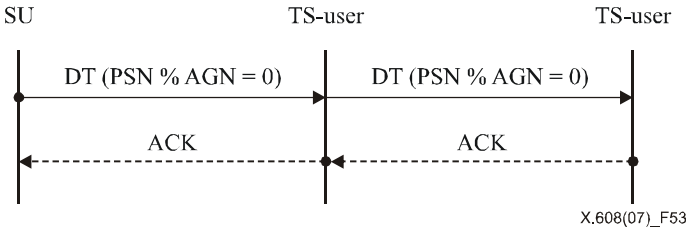

Figure 53 – ACK control procedure

Each child generates an ACK packet by *ACK_GENERATION_NUM (AGN)*. At every *AGN* number of packets, it examines the packet delivery status from the previously acknowledged packet to the last received packet. If all the packets in the range are successfully received, it sends an ACK packet to its parent.

Each child sends an ACK packet to its parent, if the *PSN* number of a DT packet modulo *AGN* equals zero, i.e., if:

$$PSN \% AGN = 0$$

Supposing *AGN* = 8, the child generates an ACK packet for the DT packets whose sequence numbers are 8, 16, 24, 32, etc. This ACK generation rule is applied when the corresponding DT or RD packets are received by the child.

9.3.2.5 Acknowledgement of data reception and ACK aggregation

Each parent uses ACK packets to gather status information of data reception by TS-users. Each time a parent receives an ACK packet from any of its children, it records and updates the status information on which packets have been successfully received by its children.

A DT packet is defined as a 'stable' packet if all of the children have received it. The stable DT packets may be released out of the buffer memory of the parent.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

9.4 Token control

In ECTP-5, a token represents the right for a TS-user to transmit multicast data. Each TS-user who wants to transmit data must get a token from the TCN. The TS-user will be an SU after getting a token from TCN. In this way, TCN can authorize a TS-user to become a sender so that TS-users can effectively filter out multicast data sent by unauthorized users. However, note that use of token does not provide any protection for IP multicast.

An SU should return the token after completing the data transmission.

9.4.1 Token get

The TS-user can get a token in two different ways: TS-user-initiated Token Get and TCN-initiated Token Give. In the Token Get operation, the TS-user first requests a token to TCN, whereas in the Token Give case the TCN first gives a token to a prospective TS-user.

Figure 54 shows the operations for Token Get and Token Give.

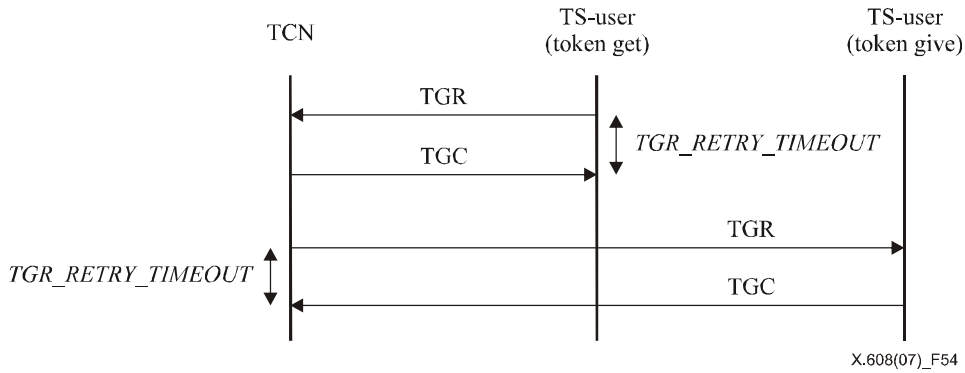

Figure 54 – Token get and give procedures

To get a token in the Token Get operation, a TS-user sends a TGR message to TCN, and then waits for the corresponding TGC message. In response to the TGR packet, the TCN should send a TGC message to the TS-user. The TGC message should indicate whether the request is accepted or not by using the *F* flag of the base header. In case of the acceptance, the message will also contain a valid *Token ID* in the base header. If the responding TGC message has not arrived within *TGR_RETRY_TIMEOUT*, the TS-user may send the TGR message to TCN again. This process can be repeated up to *TGR_MAX_RETRY* times. If the node fails to receive TGC packet, it gives up the token get procedures and returns an error with status information to the application.

In the Toke Give operation, the TCN will send a TGR message to a TS-user. The TGR message should contain the *Token ID* in the base header.

The TS-user (i.e., SU) should respond with the TGC message that sets the *F* flag to '1' (acceptance). If the responding TGC message has not arrived from the SU within *TGR_RETRY_TIMEOUT*, the TCN may send the TGR message to SU again. This process can be repeated up to *TGR_MAX_RETRY* times. If TCN fails to receive TGC packet, it gives up the token give procedures and excludes the TS-user from the valid SU list so that the following TSR packets do not include the TS-user.

9.4.2 Token return

When completing data transmission, the SU may return the token to TCN. The SU can return its token to TCN in two different ways: TS-user-initiated Token Return and TCN-initiated Token Withdrawal. In the Token Return operation, the TS-user sends the TRR packet to TCN, whereas in the Token Withdrawal case the TCN will first send the TRR message to a TS-user.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Figure 55 shows the operations for Token Return and Token Withdrawal.

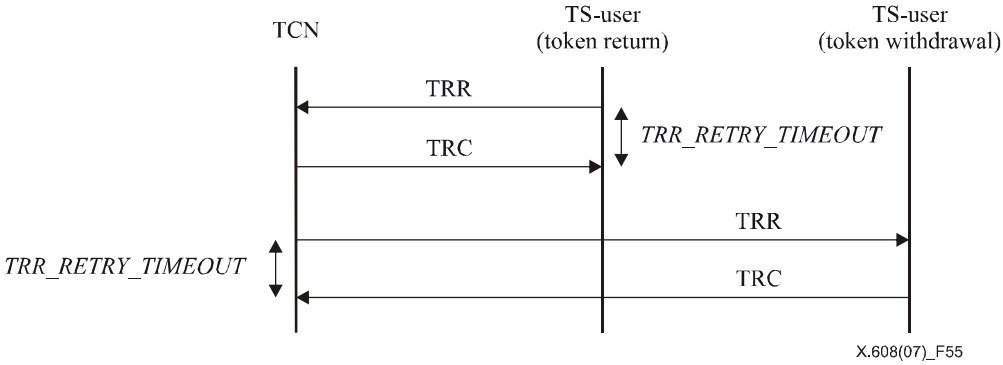

Figure 55 – Token return and withdrawal procedures

In the Token Return case, the SU sends a TRR message to TCN. The TCN then responds with the TRC message. On the other hand, in the Token Withdrawal case, the TCN may enforce the concerned SU to return the token by sending a TRR message. If the responding TRC message has not arrived within *TRR_RETRY_TIMEOUT*, the TRR message may be sent again. This process can be repeated up to *TRR_MAX_RETRY* times.

In the Token Return case, if the TS-user fails to receive a TRC packet, it gives up the procedures. In the Token Withdrawal case, if TCN fails to receive a TRC packet, it gives up the procedures and removes the token from the valid token list and sends the updated TSR promptly in order to inform the participants.

9.4.3 Token status report

TCN reports the status of the valid token IDs in the connection by sending a TSR packet. A TSR packet is transmitted by TCN when a new SU is coming up, or when an existing SU stops the data transmission, or periodically with the interval of *TSR_PACKET_INT* for the token maintenance.

Figure 56 shows the operations for Token Status Report.

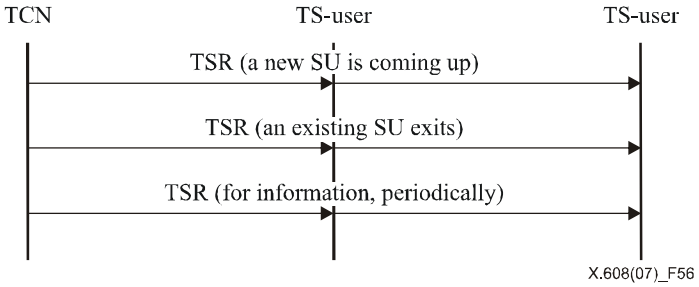

Figure 56 – Token status report procedures

If a TS-user receives DT from an SU that is currently not listed in the token list, it requests TCN to update the token list by sending TSRR (TSR Request) to TCN. At receiving TSRR, TCN replies with TSR including updated token list. If a TSR packet does not arrive within *TSRR_RETRY_TIMEOUT*, the TS-user of TSRR packet may try to send the TSRR packet again. This process can be repeated up to *TSRR_MAX_RETRY* times. If the node has not received TSR packet, it gives up the token status report request procedures and ignores DT packets from the SU.

If a TS-user does not receive next TSR within *TSR_ARRIVAL_TIMEOUT* after receiving the last TSR, it requests TCN to confirm the connection is valid by sending TSRR (TSR Request) to TCN. At receiving TSRR, TCN replies with TSR with confirmation of availability of the connection. If a TSR packet does not arrive within *TSRR_RETRY_TIMEOUT*, the TS-user of TSRR packet may try to send the TSRR packet again. This process can be repeated up to *TSRR_MAX_RETRY* times. If the node has not received TSR packet, it gives up the procedures, returns an error with status information to the application and terminates the N-plex multicast connection.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Figure 57 shows the operations for Token Status Report Request.

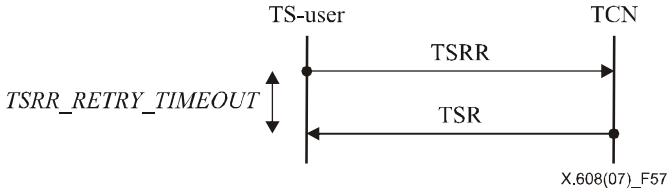

Figure 57 – Token status report request procedures

9.5 RTT measurement

RTT between a parent and a child node may be used to determine the retry interval of control packets which are exchanged between them. In order to estimate RTT value, the sender's current time is carried on TJ and NACK. Then the node receiving the packet responds respectively with a TC and RD message containing the time value copied from the received packet.

10 System parameters

Table 4 shows the ECTP-5 system parameters.

Table 4 – ECTP-5 system parameters

Name	Description
ACK_GENERATION_NUM	The minimum gap of PSNs to send next ACK
CCR_MAX_RETRY	The maximum number of retries for CCR
CCR_RETRY_TIMEOUT	The retry interval for CCR
CR_MAX_RETRY	The maximum number of retries for CR
CR_RESPONSE_TIMEOUT	The waiting time for the responses to CR from TS-users
JR_MAX_RETRY	The maximum number of retries for JR
JR_RETRY_TIMEOUT	The retry interval for JR
MAX_LSN_LAG	The maximum allowed lag of child node's LSN
MAX_SEGMENT_SIZE	The maximum size of the user data segment
NACK_MAX_RETRY	The maximum number of retries for NACK
NACK_RETRY_TIMEOUT	The retry interval for NACK
PB_MAX_RETRY	The maximum number of retries for PB
PB_PACKET_INT	The interval of sending PB
PB_RETRY_TIMEOUT	The retry interval for PB
TCR_MAX_RETRY	The maximum number of retries for TCR
TCR_RETRY_TIMEOUT	The retry interval for TCR
TD_PACKET_INT	The interval of sending test data packets
TD_PACKET_NUM	The number of test data packets for one logical tree adaptation process
TD_PACKET_SIZE	The payload size of test data packet
TDR_MAX_RETRY	The maximum number of retries for TDR
TDR_RETRY_TIMEOUT	The retry interval for TDR
TGR_MAX_RETRY	The maximum number of retries for TGR
TGR_RETRY_TIMEOUT	The retry interval for TGR
TJ_MAX_RETRY	The maximum number of retries for TJ
TJ_RETRY_TIMEOUT	The retry interval for TJ
TLR_MAX_RETRY	The maximum number of retries for TLR

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Table 4 – ECTP-5 system parameters

Name	Description
TLR_RETRY_TIMEOUT	The retry interval for TLR
TNR_MAX_RETRY	The maximum number of retries for TNR
TNR_RETRY_TIMEOUT	The retry interval for TNR
TRR_MAX_RETRY	The maximum number of retries for TRR
TRR_RETRY_TIMEOUT	The retry interval for TRR
TSR_ARRIVAL_TIMEOUT	The maximum latency to next TSR arrival
TSR_PACKET_INT	The interval of sending TSR
TSRR_MAX_RETRY	The maximum number of retries for TSRR
TSRR_RETRY_TIMEOUT	The retry interval for TSRR

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

Annex A

Application programming interfaces

(This annex does not form an integral part of this Recommendation | International Standard)

This annex specifies the application programming interfaces (API). The APIs described in this Recommendation | International Standard can be used by applications that utilize the transport capabilities of ECTP part 5 (ITU-T Rec. X.608 | ISO/IEC 14476-5). This API is derived from Annex B of ITU-T Rec. X.606.1 | ISO/IEC 14476-2.

A.1 Overview

A.1.1 API functions

Table A.1 summarizes the API functions.

Table A.1 – API functions

Function Name	Description
`msocket()`	Creates a new ECTP-5 socket.
`mbind()`	Associates a set of local and group addresses/ports with the socket.
`mconnect()`	TCN initiates a connection creation to a specified foreign address. Late-joining TS-user initiates a join process.
`maccept()`	Prospective TS-users join the N-plex multicast connection by accepting the connection creation signal from TCN.
`msend()`	Sends application data to a destination group.
`mrecv()`	Delivers received data to application. Delivers some indication messages for control to application during the data transfer phase.
`mclose()`	Terminates connection and releases socket.

A.1.2 Use of API functions

For typical sequence of using API functions, you may refer to clause B.1.2 of ITU-T Rec. X.606.1 | ISO/IEC 14476-2.

A.2 ECTP-5 API functions

A.2.1 msocket()

To use the protocol defined in this Recommendation | International Standard, an application MUST invoke the `msocket` function firstly, which specifies the type of communication protocol desired such as ECTP using IPv4 or ECTP using IPv6.

```
int msocket(int family, int type, int protocol);
```

Parameter description:

- *family*: specifies the protocol family and is one of the constants shown in Table A.2;
- *type*: specifies the type of socket and one of the constants shown in Table A.3;
- *protocol*: is set to 0.

Table A.2 – Protocol family constants used for `msocket` function

Family	Description
AF_INET	IPv4 protocols
AF_INET6	IPv6 protocols

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Table A.3 – *type* of socket used for `msocket` function

Type	Description
SOCK_ECTP5	Socket for ECTP part 5

The `msocket` call returns non-negative ECTP-5 socket descriptor if success, or –1 on the following cases as listed in Table A.4.

Table A.4 – Error codes for `msocket` call

Error code	Description
EPROTONOSUPPORT	The protocol type or the specified protocol is not supported within this domain.
EMFILE	The per-process descriptor table is full.
ENFILE	The system file table is full.
EACCES	Permission to create a socket of the specified type and/or protocol is denied.
ENOBUFS	Insufficient buffer space is available.

A.2.2 mbind()

The `mbind` function assigns a set of local, group, control addresses, and the role of the node in the session to a socket.

With the Internet protocols the protocol address is the combination of either a 32-bit IPv4 address or a 128-bit IPv6 address, along with a 16-bit port number.

```
int mbind(int msockfd, const struct sockaddr *laddr, socklen_t laddrlen, const
struct sockaddr *gaddr, socklen_t gaddrlen, int role, struct ectp5_option
*options);
```

Parameter description:

- *msockfd*: is a socket descriptor that was returned by the `msocket` function;
- *laddr*: is a pointer to a protocol-specific address to bind a local address to the above socket;
- *laddrlen*: is the size of the above address structure;
- *gaddr*: is a pointer to a protocol-specific address to bind a target group address to the socket;
- *gaddrlen*: is the size of the group address structure;
- *role*: specifies the role of this calling initiator such as TCN, LO or LE;
- *options*: specifies the options of an N-plex multicast connection if *role* is TCN.

Table A.5 – *role* of the socket user for `mbind` function

role	Description
TCN	Connection creator and sender as the owner in the ECTP-5 communications.
LO	Receiver taking a responsible for retransmissions in the tree-based hierarchy.
LE	Receiver that is not designated LO.

Table A.6 – The fields of *options* parameter in `mbind` function

Option fields	Description
TCO	Tree configuration option. '10' by default.
AGN	ACK generation number. '32' by default.
MSS	Maximum segment size. '1024' by default.

An application can `mbind` a specific IP address and a group network address to its socket. The source and group addresses must belong to an interface on the host.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

The mbind call returns zero if success or –1 on the following reasons as listed in Table A.7.

Table A.7 – Error codes that mbind may cause

Error code	Description
EAGAIN	Kernel resources to complete the request are temporarily unavailable.
EBADF	*msockfd* is not a valid descriptor.
ENOTSOCK	*msockfd* is not a socket.
EADDRNOTAVAIL	The specified address is not available from the local machine.
EADDRINUSE	The specified address is already in use.
EACCES	The requested address is protected, and the current user has inadequate permission to access it.
EFAULT	The address parameter is not in a valid part of the user address space.
EROLE	The requested role is not valid.

A.2.3 maccept()

Only LE and LO can invoke this function. It can wait TCN's initiation for a period specified by *timeout* parameter and informs whether the multicast connection has established or not.

```
int maccept(int msockfd, struct sockaddr *raddr, socklen_t *raddrlen, int
timeout );
```

Parameter description:

- *msockfd*: is a socket descriptor that was returned by the msocket function;
- *raddr*: returns the protocol address of the remote connection initiator (the sender or TCN);
- *raddrlen*: is a pointer to the size of the socket address structure pointed by *raddr*;
- *timeout*: is timeout (in seconds) value for the period waiting for TCN's CR.

If maccept is successful, it returns the same value as the first argument, *msockfd.* After that, we call this return value the *connected socket* descriptor.

The maccept call returns non-negative descriptor if success, or –1 on the following reasons as listed in Table A.8.

Table A.8 – Error codes used for maccept

Error code	Description
EBADF	The descriptor is invalid.
EINTR	The maccept operation was interrupted.
EMFILE	The per-process descriptor table is full.
ENFILE	The system file table is full.
ENOTSOCK	The descriptor references a file, not a socket.
EFAULT	The *addr* parameter is not in a writable part of the user address space.
EWOULDBLOCK	The socket is marked non-blocking and no connections are present to be accepted.
ECONNABORTED	A connection arrived, but it was closed while waiting on the listen queue.
ECRTIMEOUT	Indicates that the CR waiting time has expired.

A.2.4 mconnect()

The mconnect function is used by TCN or late joining LE to establish a connection.

```
int mconnect(int msockfd, const struct sockaddr *daddr, socklen_t daddrlen);
```

Parameter description:

- *msockfd*: is a socket descriptor that was returned by the msocket function;
- *daddr*: is a pointer to a group address;
- *daddrlen*: is the size of *daddr*.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

The `mconnect` returns zero if success or –1 in the abnormal cases as listed in Table A.9.

Table A.9 – Error numbers of `mconnect` function

Error code	Description
EBADF	*msockfd* is not a valid descriptor.
ENOTSOCK	*msockfd* is a descriptor for a file, not a socket.
EADDRNOTAVAIL	The specified address is not available on this machine.
EAFNOSUPPORT	Addresses in the specified address family cannot be used with this socket.
EISCONN	The socket is already connected.
ECONNREFUSED	The attempt to connect was forcefully rejected.
ENETUNREACH	The network is not reachable from this host.
EADDRINUSE	The address is already in use.
EFAULT	The name parameter specifies an area outside the process address space.
EALREADY	The socket is non-blocking and a previous connection attempt has not yet been completed.
EDENIED	Indicates that the TCN has declined LE's or LO's join request.
ETIMEDOUT	Connection establishment timed out without establishing a connection. Indicates that there is no response from TCN

A.2.5 msend()

This `msend` function writes data from a buffer into the connected socket.

```
ssize_t msend (int msockfd, const void *buf, size_t buflen, int *flags);
```

Parameter description:

- *msockfd*: is a socket descriptor that was returned by the `msocket` function;
- *buf*: is a pointer to buffer to write from;
- *buflen*: is the size of *buf*; and
- *flags*: is not used.

The `msend` returns the number of bytes written if success or –1 in the abnormal cases as listed in Table A.10.

Table A.10 – Error codes of `msend`

Error code	Description
EBADF	An invalid descriptor was specified.
EACCES	The destination address is a broadcast address, and SO_BROADCAST has not been set on the socket.
ENOTSOCK	The argument *msockfd* is not a socket.
EFAULT	An invalid user space address was specified for a parameter.
EMSGSIZE	The socket requires that message be sent atomically, and the size of the message to be sent made this impossible.
EAGAIN	The socket is marked non-blocking and the requested operation would block.
ENOBUFS	The system was unable to allocate an internal buffer. The operation may succeed when buffers become available.
ENOBUFS	The output queue for a network interface was full. This generally indicates that the interface has stopped sending, but may be caused by transient congestion.
EPARTITIONED	Indicates that the session has been partitioned.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

A.2.6 mrecv()

The `mrecv` function is used to receive the multicast data and indication signals for control purposes.

```
ssize_t mrecv (int msockfd, void *buf, size_t buflen, int *flags, struct
sockaddr *fromaddr, socklen_t *fromaddrlen);
```

Parameter description:

- *msockfd*: is a socket descriptor that was returned by the `msocket` function;
- *buf*: is a pointer to buffer to read into;
- *buflen*: is size of *buf*;
- *flags*: is not defined yet;
- *fromaddr*: is a pointer to a protocol-specific address to specify the SU;
- *fromaddrlen*: is the size of *fromaddr*.

When an application receives data from the buffer, it can identify the corresponding sender (or SU) by using *fromaddr*.

The `mrecv` returns the number of bytes received if success. Otherwise it returns –1, if an error occurs or there is a control message that will be delivered to application. Error codes are listed in Table A.11.

Table A.11 – Error codes of `mrecv` function

Error code	Description
EBADF	The argument *msockfd* is an invalid descriptor.
ENOTCONN	The socket is associated with a connection-oriented protocol and has not been connected (see `mconnect` and `maccept`).
ENOTSOCK	The argument *msockfd* does not refer to a socket.
EAGAIN	The socket is marked non-blocking, and the receive operation would block, or a receive timeout had been set, and the timeout expired before data were received.
EINTR	Operation was interrupted by delivery of a signal before any data were available.
EFAULT	The receive buffer pointer(s) point outside the process's address space.
ETOTERM	TCN has terminated the session.
ETOEXPEL	TCN has expelled the LE or LO.

A.2.7 mclose()

The `mclose` function is used to leave or terminate an N-plex multicast connection by closing the socket. The default action of `mclose` with an ECTP-5 socket is to mark the socket as closed and return it to the process immediately. The socket descriptor is no longer usable by the process.

```
int mclose (int msockfd);
```

Parameter description:

- *msockfd*: is a socket descriptor that was returned by the `msocket` function.

The `mclose` returns zero if success or –1 in the case of error as listed in Table A.12.

Table A.12 – Error codes of `mclose`

Error code	Description
EBADF	*msockfd* is not an active descriptor.
EINTR	An interrupt was received.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Annex B

State transition diagrams

(This annex does not form an integral part of this Recommendation | International Standard)

This annex gives a sketch of the state transition diagrams of ECTP-5 nodes, TCN, LO, and TS-user, so as to facilitate an implementation of this Recommendation | International Standard.

Figure B.1 shows the state transition diagram of ECTP-5 TCN, which is described based on clause 8.

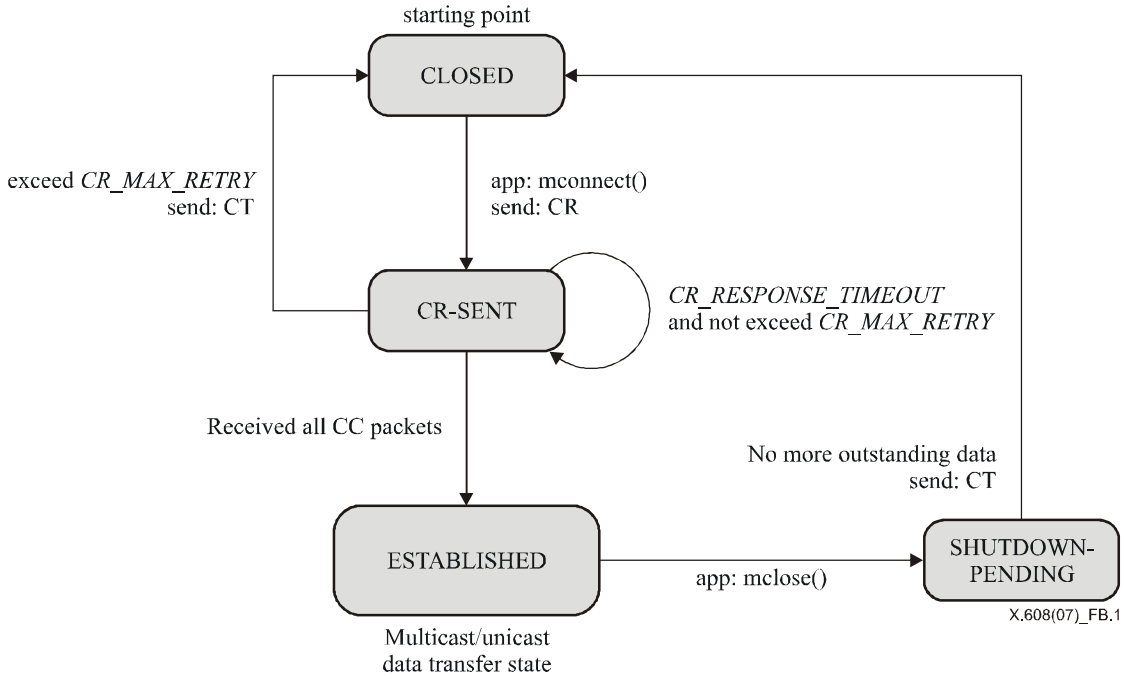

Figure B.1 – State transition diagram for TCN

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Figure B.2 below shows the state transition diagram of ECTP-5 LO and TS-user, which is described based on clause 8.

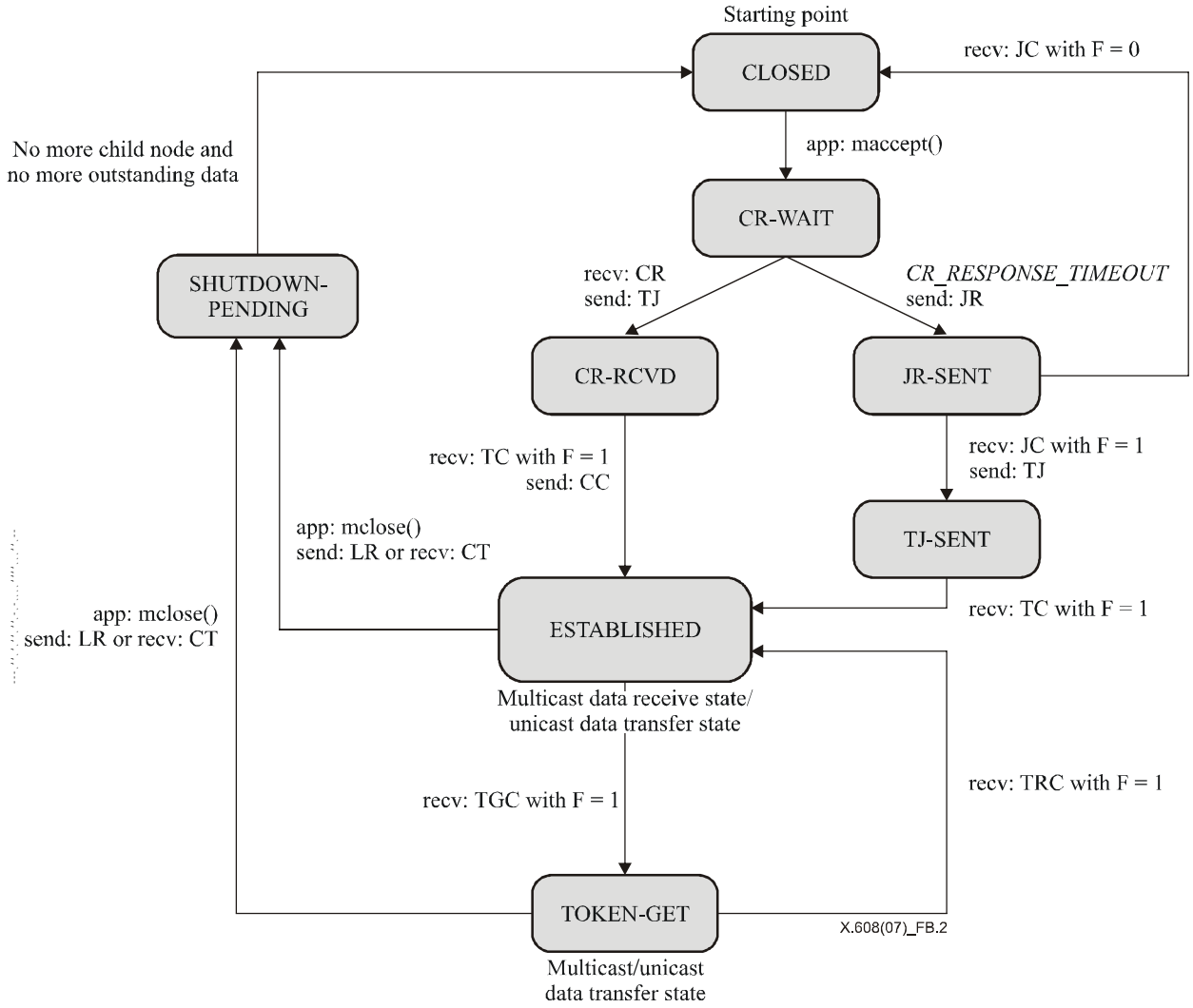

Figure B.2 – State transition diagram for ECTP-5 LO and TS-user

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Annex C

An example of system parameters values in ECTP-5

(This annex does not form an integral part of this Recommendation | International Standard)

This annex gives example values of ECTP-5 system parameters that might be referred by the implementers of ECTP-5. See Table C.1

Table C.1 – Example values of ECTP-5 system parameters

Name	Default value
ACK_GENERATION_NUM	32
CCR_MAX_RETRY	5
CCR_RETRY_TIMEOUT	200 ms
CR_MAX_RETRY	5
CR_RESPONSE_TIMEOUT	5 s
JR_MAX_RETRY	5
JR_RETRY_TIMEOUT	200 ms
MAX_SEGMENT_SIZE	1024 bytes
NACK_MAX_RETRY	5
NACK_RETRY_TIMEOUT	200 ms
PB_MAX_RETRY	5
PB_PACKET_INT	3 s
PB_RETRY_TIMEOUT	500 ms
TCR_MAX_RETRY	5
TCR_RETRY_TIMEOUT	200 ms
TD_PACKET_INT	5 ms
TD_PACKET_NUM	1000
TD_PACKET_SIZE	512 bytes
TDR_MAX_RETRY	5
TDR_RETRY_TIMEOUT	200 ms
TGR_MAX_RETRY	5
TGR_RETRY_TIMEOUT	200 ms
TJ_MAX_RETRY	5
TJ_RETRY_TIMEOUT	200 ms
TLR_MAX_RETRY	5
TLR_RETRY_TIMEOUT	200 ms
TNR_MAX_RETRY	5
TNR_RETRY_TIMEOUT	200 ms
TRR_MAX_RETRY	5
TRR_RETRY_TIMEOUT	200 ms
TSR_ARRIVAL_TIMEOUT	15 s
TSR_PACKET_INT	5 s
TSRR_MAX_RETRY	5
TSRR_RETRY_TIMEOUT	500 ms

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

These values are selected for the following environment:

- Session size: 30 TS-users (30 SUs);
- Number of local groups: 3 LOs and 3 local groups;
- Sending rates: 512 kbit/s;
- Link bandwidth: 100 Mbit/s;
- Link delay: 40~50 ms between local groups, and 10~25 ms in a local group;
- End-to-end error rates: 0.05~0.25.

--`,,```,,,,````-`-`,,`,,`,`,,`---

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

ICS 35.100.40

Price based on 57 pages

--`,,```,,,,````-`-`,,`,,`,`,,`---

© ISO/IEC 2008 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS

Not for Resale