

INTERNATIONAL STANDARD

NORME INTERNATIONALE



**Electronic railway equipment – Train communication network (TCN) –
Part 2-3: TCN communication profile**

**Matériel électronique ferroviaire – Réseau embarqué de train (TCN) –
Partie 2-3: Profil de communication TCN**

IECNORM.COM : Click to view the full PDF of IEC 61375-2-3:2015



THIS PUBLICATION IS COPYRIGHT PROTECTED

Copyright © 2015 IEC, Geneva, Switzerland

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 IEC or IEC's member National Committee in the country of the requester. If you have any questions about IEC copyright or have an enquiry about obtaining additional rights to this publication, please contact the address below or your local IEC member National Committee for further information.

Droits de reproduction réservés. Sauf indication contraire, aucune partie de cette publication ne peut être reproduite ni utilisée sous quelque forme que ce soit et par aucun procédé, électronique ou mécanique, y compris la photocopie et les microfilms, sans l'accord écrit de l'IEC ou du Comité national de l'IEC du pays du demandeur. Si vous avez des questions sur le copyright de l'IEC ou si vous désirez obtenir des droits supplémentaires sur cette publication, utilisez les coordonnées ci-après ou contactez le Comité national de l'IEC de votre pays de résidence.

IEC Central Office
3, rue de Varembe
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
info@iec.ch
www.iec.ch

About the IEC

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

IEC Catalogue - webstore.iec.ch/catalogue

The stand-alone application for consulting the entire bibliographical information on IEC International Standards, Technical Specifications, Technical Reports and other documents. Available for PC, Mac OS, Android Tablets and iPad.

IEC publications search - webstore.iec.ch/advsearchform

The advanced search enables to find IEC publications by a variety of criteria (reference number, text, technical committee,...). It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished

Stay up to date on all new IEC publications. Just Published details all new publications released. Available online and also once a month by email.

Electropedia - www.electropedia.org

The world's leading online dictionary of electronic and electrical terms containing 21 000 terms and definitions in English and French, with equivalent terms in 16 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

IEC Glossary - std.iec.ch/glossary

67 000 electrotechnical terminology entries in English and French extracted from the Terms and Definitions clause of IEC publications issued since 2002. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.

IEC Customer Service Centre - webstore.iec.ch/csc

If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: sales@iec.ch.

A propos de l'IEC

La Commission Electrotechnique Internationale (IEC) est la première organisation mondiale qui élabore et publie des Normes internationales pour tout ce qui a trait à l'électricité, à l'électronique et aux technologies apparentées.

A propos des publications IEC

Le contenu technique des publications IEC est constamment revu. Veuillez vous assurer que vous possédez l'édition la plus récente, un corrigendum ou amendement peut avoir été publié.

Catalogue IEC - webstore.iec.ch/catalogue

Application autonome pour consulter tous les renseignements bibliographiques sur les Normes internationales, Spécifications techniques, Rapports techniques et autres documents de l'IEC. Disponible pour PC, Mac OS, tablettes Android et iPad.

Recherche de publications IEC - webstore.iec.ch/advsearchform

La recherche avancée permet de trouver des publications IEC en utilisant différents critères (numéro de référence, texte, comité d'études,...). Elle donne aussi des informations sur les projets et les publications remplacées ou retirées.

IEC Just Published - webstore.iec.ch/justpublished

Restez informé sur les nouvelles publications IEC. Just Published détaille les nouvelles publications parues. Disponible en ligne et aussi une fois par mois par email.

Electropedia - www.electropedia.org

Le premier dictionnaire en ligne de termes électroniques et électriques. Il contient 21 000 termes et définitions en anglais et en français, ainsi que les termes équivalents dans 16 langues additionnelles. Egalement appelé Vocabulaire Electrotechnique International (IEV) en ligne.

Glossaire IEC - std.iec.ch/glossary

67 000 entrées terminologiques électrotechniques, en anglais et en français, extraites des articles Termes et Définitions des publications IEC parues depuis 2002. Plus certaines entrées antérieures extraites des publications des CE 37, 77, 86 et CISPR de l'IEC.

Service Clients - webstore.iec.ch/csc

Si vous désirez nous donner des commentaires sur cette publication ou si vous avez des questions contactez-nous: sales@iec.ch.

INTERNATIONAL STANDARD

NORME INTERNATIONALE



**Electronic railway equipment – Train communication network (TCN) –
Part 2-3: TCN communication profile**

**Matériel électronique ferroviaire – Réseau embarqué de train (TCN) –
Partie 2-3: Profil de communication TCN**

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

COMMISSION
ELECTROTECHNIQUE
INTERNATIONALE

ICS 45.060.01

ISBN 978-2-8322-6098-2

**Warning! Make sure that you obtained this publication from an authorized distributor.
Attention! Veuillez vous assurer que vous avez obtenu cette publication via un distributeur agréé.**

CONTENTS

FOREWORD.....	13
INTRODUCTION.....	15
1 Scope.....	16
2 Normative references.....	17
3 Terms, definitions, abbreviations, acronyms, and conventions	18
3.1 Terms and definitions.....	18
3.2 Abbreviations and acronyms.....	26
3.3 Conventions.....	28
3.3.1 Base of numeric values.....	28
3.3.2 Character strings and citations.....	28
3.3.3 Naming conventions.....	29
3.3.4 Diagram conventions	29
3.3.5 Annotation of data structures	29
4 Architecture	31
4.1 General.....	31
4.2 Physical train architecture (system breakdown).....	31
4.2.1 General	31
4.2.2 Train network architectures	31
4.2.3 Closed Trains	35
4.2.4 Directions	36
4.2.5 Consist and vehicle basic properties	37
4.3 Logical Train Architecture (Functional Breakdown)	38
4.3.1 General	38
4.3.2 Service classification	38
4.3.3 Operational Services Overview.....	39
4.3.4 Service Provider.....	40
5 Common ETB framework.....	40
5.1 General.....	40
5.1.1 Overview	40
5.1.2 Interoperability.....	40
5.2 CSTINFO telegram.....	41
5.2.1 General	41
5.2.2 Closed train support (Option)	41
5.2.3 Protocol	41
5.2.4 CSTINFO classes	41
5.2.5 CSTINFO Notification Message.....	42
5.2.6 CSTINFO Request	43
5.3 Train topology database	45
5.3.1 General	45
5.3.2 Computation of the TTDB.....	47
5.3.3 Data structure	52
5.3.4 Train Topology Database for multiple ETBs (Option)	60
5.4 Service Addressing	62
5.4.1 General	62
5.4.2 TCN Domain Name System (TCN-DNS).....	63
5.4.3 TCN Domain Names	64
5.4.4 TCN-URI Scheme	64

5.4.5	Mapping TCN-URI to IP address	71
5.4.6	Support of other URI schemas	74
5.5	TCN-DNS Server.....	74
5.5.1	General	74
5.5.2	Architecture	75
5.5.3	Functional address resolution.....	75
5.5.4	Protocol	77
5.5.5	Multiple ETBs	78
5.6	Data exchange	78
5.6.1	General	78
5.6.2	Operational network communication	78
5.6.3	OMTS network communication	79
5.6.4	Quality of Service (QoS).....	79
5.7	Service discovery	80
5.8	Train Info Service	80
6	Services of the communication profile – ETB Control Service	80
6.1	General.....	80
6.2	Communication model	80
6.3	ECSP Supervision	81
6.4	ECSP Interconnection	81
6.4.1	General	81
6.4.2	ETBCTRL telegram exchange selection	81
6.4.3	ETBCTRL telegram transmission	82
6.4.4	Structure of the ETBCTRL telegram	82
6.4.5	Operational train directory computation process	85
6.5	Function “Leading”	87
6.5.1	General	87
6.5.2	Function primitives	88
6.5.3	ECSP to ECSP protocol	89
6.6	Function Confirmation/Correction	93
6.6.1	General	93
6.6.2	Function primitives	93
6.6.3	ECSP to ECSP protocol	95
6.6.4	State diagram	98
6.6.5	ECSC Failure	100
6.7	Computation of the operational train directory	100
6.7.1	General	100
6.7.2	Action setCorrInfo	101
6.7.3	Action computeOpTrnDir	104
6.8	Function Sleep Mode (Option)	107
6.8.1	General	107
6.8.2	Sleep Mode Use Case (informal)	107
6.8.3	Exclusivity.....	109
6.8.4	Function primitives	109
6.8.5	ECSP to ECSP protocol	111
Annex A (normative)	Train Real-Time Data Protocol (TRDP)	115
A.1	General.....	115
A.2	Lower Layers	115
A.2.1	Data link layer	115

A.2.2	Network Layer.....	115
A.2.3	Transport Layer	116
A.3	TRDP FCS Computation.....	117
A.4	Interaction between TRDP user and TRDP Layer.....	119
A.5	Communication Identifier (ComId)	119
A.6	Process Data	121
A.6.1	Communication model.....	121
A.6.2	Roles	121
A.6.3	Communication pattern	121
A.6.4	Addressing.....	126
A.6.5	PD-PDU.....	126
A.6.6	Interaction between application and TRDP protocol layer.....	129
A.6.7	Topography counter check	136
A.6.8	State Machine.....	137
A.7	Message Data	141
A.7.1	Communication model.....	141
A.7.2	Roles	142
A.7.3	Communication pattern	142
A.7.4	Addressing.....	143
A.7.5	MD-PDU	143
A.7.6	Interaction between application and TRDP layer	147
A.7.7	Topography counter check	151
A.7.8	MD protocol state machine.....	152
A.7.9	TCP Connection Handling	161
A.8	Message data echo server (option).....	162
Annex B (normative)	Safe Data Transmission (SDTv2)	163
B.1	General.....	163
B.2	Overview of SDTv2 (informal).....	163
B.3	Safety functional requirements	164
B.4	Safety measures	164
B.5	Operational states of the SDTv2 channel	165
B.6	Data presentation.....	166
B.7	SC-32	166
B.8	SID	169
B.9	Vital Data Packet	170
B.10	Exclusivity.....	171
B.11	Configuration time parameters.....	171
B.12	Safe data source (SDSRC).....	171
B.12.1	General	171
B.12.2	Safe Data Preparation (Application)	171
B.12.3	Safe data sending	172
B.13	Safe data sink (SDSINK)	173
B.13.1	General	173
B.13.2	Definitions	174
B.13.3	SDSINK States	175
B.13.4	VDP Sampling.....	176
B.13.5	VDP Integrity Check.....	177
B.13.6	Sink time supervision	178
B.13.7	Guard time check.....	178

B.13.8	Latency monitoring.....	179
B.13.9	Channel monitoring.....	181
B.13.10	SDTv2 Application Interface.....	182
B.13.11	Change of operational train composition.....	183
B.14	Diagnosis and statistics.....	183
B.15	Safe data transmission over MVB (informative).....	184
B.15.1	General	184
B.15.2	MVB-VDP	184
B.15.3	SDTV2 protocol deviations for MVB.....	185
B.16	SDTv2 with TRDP message data.....	186
Annex C (informative)	Train Real-Time Data Protocol Configuration (TRDP).....	187
C.1	General.....	187
C.2	Device Parameters.....	188
C.3	Device Configuration Parameters	189
C.4	Bus Interface List	189
C.4.1	General	189
C.4.2	Bus Interface Configuration.....	190
C.5	Mapped Device Parameters.....	201
C.5.1	General	201
C.5.2	Mapped Bus Interface Parameters	202
C.6	Communication Parameters (ComPar).....	204
C.6.1	General	204
C.6.2	Default Communication Parameters.....	205
C.7	DataSet Parameters.....	205
C.7.1	General	205
C.7.2	DataSet Element.....	207
C.7.3	Examples of DataSets.....	209
Annex D (informative)	Access to End Device (ED) statistics.....	213
D.1	General.....	213
D.2	Structures	213
D.2.1	General	213
D.2.2	tlc_getSubsStatistics.....	215
D.2.3	tlc_getPubStatistics.....	215
D.2.4	tlc_getUdpListStatistics, tlc_getTcpListStatistics.....	215
D.2.5	tlc_getRedStatistics	216
D.3	ED interface for statistic data access.....	216
D.3.1	General	216
D.3.2	TRDP interface	216
Annex E (informative)	Service interface	218
E.1	General.....	218
E.2	Service provider	219
E.2.1	Proxies	219
E.2.2	Performance	219
E.3	ECSP interface	219
E.3.1	General	219
E.3.2	ECSP control telegram.....	219
E.3.3	ECSP status telegram.....	221
E.3.4	ECSP Confirmation/Correction Request	224
E.4	TTDB manager interface	227

E.4.1	General	227
E.4.2	TTDB status information	227
E.4.3	TTDB notification	228
E.4.4	TTDB information – train directory	228
E.4.5	TTDB information – static consist information	229
E.4.6	TTDB information – train network directory information	230
E.4.7	Operational train directory information	231
E.4.8	Read TTDB	232
E.5	DNS server interface	233
E.5.1	DNS standard interface	233
E.5.2	DNS TCN interface	233
E.6	ETBN control interface	239
E.6.1	General	239
E.6.2	ETBN control and status data	240
E.6.3	ETBN train network directory	243
Annex F (normative)	Communication profile conformance test guideline	244
F.1	General	244
F.2	Scope of conformance test	244
F.3	Conformance test overview	245
F.4	Test laboratory	245
F.4.1	General	245
F.4.2	Tasks	245
F.5	Guideline for writing conformance test specifications	246
F.5.1	Overview of the main components	246
F.5.2	Protocol Implementation Conformance Statement (PICS)	246
F.5.3	Abstract test architecture	247
F.5.4	Protocol Implementation eXtra Information for Testing (PIXIT)	247
F.5.5	Test suite structure	247
F.6	Abstract test architecture (option)	247
F.6.1	General	247
F.6.2	Test architecture with one ETB	248
F.6.3	Test architecture for multiple ETB	248
F.6.4	Set-up for automatic test	248
F.7	Test of conformity to the common ETB framework	249
F.7.1	General	249
F.7.2	Test of CSTINFO telegram	249
F.7.3	Test of TTDB	249
F.7.4	Test of service addressing and TCN-DNS server	249
F.7.5	Test of data exchange	250
F.7.6	Test of service discovery	251
F.7.7	Test of train info service	251
F.8	ETB Control Service conformity test	251
F.8.1	General	251
F.8.2	Test control interface for the test of ETB control services	251
F.9	Echo function	259
F.9.1	General	259
F.9.2	TRDP echo test	259
F.9.3	Reverse-Echo test	260
F.10	Statement of conformity	261

Annex G (informative) SNMP Management Information Base (MIB)	263
G.1 General.....	263
G.2 TTDB-MIB.....	263
G.3 TRDP-MIB.....	268
Bibliography	279

Figure 1 – IEC 61375-2-3 as connecting element between train backbone and application	17
Figure 2 – Train structure in accordance to IEC 61375-1 (example)	31
Figure 3 – Train structure seen from viewpoint of the communication profile (example).....	31
Figure 4 – Train network (example)	32
Figure 5 – Possible couplings of operational network and multimedia network.....	33
Figure 6 – Gateway between operational network and multimedia network (example).....	34
Figure 7 – Example: three coupled Consists	35
Figure 8 – Example: Closed Train.....	36
Figure 9 – Service classification	39
Figure 10 – CSTINFO notification data	43
Figure 11 – CSTINFOCTRL telegram	45
Figure 12 – TTDB management block diagram	45
Figure 13 – TTDB Content.....	46
Figure 14 – TTDB computation block diagram	47
Figure 15 – Train directory computation state diagram	48
Figure 16 – TTDB class diagram (example).....	52
Figure 17 – TTDB adoption (in this example shown for the first consist).....	61
Figure 18 – TCN-DNS name space with division into zones	63
Figure 19 – TCN-URI Schema	65
Figure 20 – Directions, orientations and numbers in train.....	66
Figure 21 – TCN-URI resolving in a train	75
Figure 22 – DNS protocol (case a without, case b with TTDB interrogation)	78
Figure 23 – ETB control service model	80
Figure 24 – ETBCTRL telegram exchange.....	81
Figure 25 – ETBCTRL telegram.....	82
Figure 26 – Operational train directory computation block diagram.....	85
Figure 27 – ETBCTRL processing state diagram.....	86
Figure 28 – Leading sequence diagram	88
Figure 29 – Leading vehicle function state machine block diagram.....	90
Figure 30 – State diagram of leading function	91
Figure 31 – Confirmation sequence diagram.....	94
Figure 32 – Confirmation/correction function state machine block diagram.....	95
Figure 33 – Correction/confirmation protocol sequence chart (example).....	97
Figure 34 – Unconfirm protocol sequence chart (example).....	98
Figure 35 – Confirmation/correction state diagram	99
Figure 36 – Action “setCorrInfo” block diagram	101
Figure 37 – Train composition consistency check examples.....	104

Figure 38 – Computation of the operational train directory	105
Figure 39 – computeOpTrnDir state chart	106
Figure 40 – Use case “sleep mode” state diagram	109
Figure 41 – Sleep control sequence diagram	110
Figure 42 – Sleep control function state machine block diagram	111
Figure 43 – Sleep request protocol sequence chart (example)	112
Figure 44 – Sleep control state diagram	113
Figure A.1 – Overview of the protocol stack	115
Figure A.2 – FCS Computation	117
Figure A.3 – FCS Table	118
Figure A.4 – TRDP service model	119
Figure A.5 – PD push pattern (point to point)	122
Figure A.6 – PD push pattern (point to multipoint)	122
Figure A.7 – PD pull pattern (point to point, sink knows source)	123
Figure A.8 – PD pull pattern (multipoint to point, sink does not know source)	124
Figure A.9 – PD pull pattern (point to multipoint, sink knows source)	125
Figure A.10 – PD pull pattern (multipoint to multipoint, sink does not know source)	126
Figure A.11 – PD-PDU	127
Figure A.12 – Interaction sequence chart for PD pull pattern	134
Figure A.13 – Interaction sequence chart for PD push pattern	135
Figure A.14 – Interaction sequence chart for redundant PD handling	136
Figure A.15 – PD State diagram publisher	137
Figure A.16 – PD State diagram requester	139
Figure A.17 – PD State diagram subscriber	140
Figure A.18 – Message data transfer options	142
Figure A.19 – MD-PDU	144
Figure A.20 – Interaction sequence chart	151
Figure A.21 – TRDP layer MD caller state chart	154
Figure A.22 – TRDP layer MD replier state chart	157
Figure A.23 – TRDP Layer MD telegram reception	160
Figure B.1 – SDTV2 Channel	163
Figure B.2 – SDTV2 Channel States	166
Figure B.3 – SC-32 Computation	167
Figure B.4 – SC-32 Table	168
Figure B.5 – SID Computation	169
Figure B.6 – ETB-VDP	170
Figure B.7 – Format of ETB-VDP	171
Figure B.8 – Redundancy Group (Example with 2 SDSRC)	173
Figure B.9 – SDSINK state diagram	175
Figure B.10 – Window of expected SSC (example)	177
Figure B.11 – Guard time violation (example)	179
Figure B.12 – Latency violation sequence chart (example)	180
Figure B.13 – MVB-VDP	184

Figure B.14 – Format of MVB-VDP	185
Figure C.1 – TRDP configuration block diagram	187
Figure C.2 – Exchange Parameters with the central key ComId.....	195
Figure C.3 – DataSet structure	206
Figure D.1 – TRDP statistics data telegrams.....	216
Figure E.1 – Service interfaces block diagram	218
Figure E.2 – ECSP interface telegrams.....	219
Figure E.3 – ECSP control data	220
Figure E.4 – ECSP status data	222
Figure E.5 – ECSP confirm/correction request data	225
Figure E.6 – ECSP confirm/correction reply data	226
Figure E.7 – TTDB manager interface telegrams	227
Figure E.8 – TCN-URI resolving	233
Figure E.9 – DNS resolving request data	236
Figure E.10 – DNS resolving reply data	238
Figure E.11 – ETBN control interface telegrams	239
Figure E.12 – ETBN control request data.....	240
Figure E.13 – ETBN status reply data	241
Figure F.1 – Consist interface on ETB level	244
Figure F.2 – Scope of conformance test	245
Figure F.3 – Abstract test architecture (1 ETB).....	248
Figure F.4 – Abstract test architecture (2 ETBs)	248
Figure F.5 – Unit under test abstract architecture	249
Figure F.6 – Conformance test control telegram	252
Figure F.7 – Conformance test control telegram data.....	252
Figure F.8 – Conformance test status telegram	253
Figure F.9 – Conformance test status telegram data.....	254
Figure F.10 – (Un-)confirmation request	254
Figure F.11 – Conformance test confirmation/correction request data	255
Figure F.12 – Conformance test confirmation/correction reply data	256
Figure F.13 – Conformance test operational train directory request.....	257
Figure F.14 – Conformance test operational train directory request data	257
Figure F.15 – Conformance test operational train directory reply data	258
Figure F.16 – Echo test	259
Figure F.17 – Reverse-Echo test	260
Figure F.18 – Conformance test message data telegram data.....	261
Table 1 – Data type keywords and notations	30
Table 2 – ETB control service.....	40
Table 3 – Train directory computation – triggers	49
Table 4 – Train directory computation – guards	49
Table 5 – Train directory computation – actions	49
Table 6 – TCN URI basic syntax.....	64

Table 7 – General schema syntax.....	64
Table 8 – Device label syntax.....	67
Table 9 – Device label definition.....	67
Table 10 – vehicle label syntax.....	68
Table 11 – Veh (vehicle) label definition.....	68
Table 12 – Consist label syntax.....	69
Table 13 – Consist label definition.....	69
Table 14 – Closed train label syntax.....	70
Table 15 – Closed train label definition.....	70
Table 16 – Train label syntax.....	70
Table 17 – Train label definition.....	70
Table 18 – General decomposition of IP MC groups addresses.....	71
Table 19 – Decomposition of all-train groups.....	71
Table 20 – Decomposition of ETB-related groups.....	72
Table 21 – Decomposition of consist-limited groups.....	72
Table 22 – Well-known TCN-URI.....	73
Table 23 – TCN-URI resolving – Example 1.....	76
Table 24 – TCN-URI resolving – Example 2.....	76
Table 25 – TCN-URI resolving – Example 3.....	77
Table 26 – TCN-URI resolving – Example 4.....	77
Table 27 – Data class priorities.....	79
Table 28 – ETBCTRL processing – triggers.....	86
Table 29 – ETBCTRL processing – guards.....	86
Table 30 – ETBCTRL processing – actions.....	86
Table 31 – Leading function primitives – F_leadingStatusRequest.....	88
Table 32 – Leading function primitives – F_leadingSetRequest.....	88
Table 33 – Leading function primitives – F_leadingResetRequest.....	89
Table 34 – Leading function control flags.....	89
Table 35 – Leading function – triggers.....	92
Table 36 – Leading function – guards.....	92
Table 37 – Leading function – actions.....	92
Table 38 – Confirmation function primitives – F_confirmStatusRequest.....	94
Table 39 – Confirmation function primitives – F_confirmRequest.....	94
Table 40 – Confirmation function primitives – F_unconfirmRequest.....	94
Table 41 – Confirmation function control flags.....	95
Table 42 – Confirmation/correction state diagram – Trigger.....	99
Table 43 – Confirmation/correction state diagram – Guard.....	99
Table 44 – Confirmation/correction state diagram – Action.....	100
Table 45 – Confirmation/Correction rules.....	101
Table 46 – Operation Train Directory computation state diagram – Trigger.....	106
Table 47 – Operation Train Directory computation state diagram – Guards.....	106
Table 48 – Operation Train Directory computation state diagram – Action.....	107
Table 49 – Example of operational train directory.....	107

Table 50 – ETBN operating conditions	108
Table 51 – Sleep mode function primitives – F_sleepStatus	110
Table 52 – Sleep mode function primitives – F_sleepRequest	110
Table 53 – Sleep mode function primitives – F_sleepCancel	110
Table 54 – Sleep mode function primitives – F_nodeAwake	111
Table 55 – Sleep control function control flags	111
Table 56 – Sleep control state diagram – trigger	113
Table 57 – Sleep control state diagram – guards	113
Table 58 – Sleep control state diagram – action	114
Table A.1 – UDP/TCP port assignments	116
Table A.2 – Reserved ComIds	120
Table A.3 – PD-PDU parameters	128
Table A.4 – TRDP service primitives	129
Table A.5 – Topography counter check	136
Table A.6 – PD publisher state diagram – guards	137
Table A.7 – PD publisher state diagram – triggers	138
Table A.8 – PD publisher state diagram – actions	138
Table A.9 – PD publisher state diagram – states	138
Table A.10 – PD publisher state diagram – guards	139
Table A.11 – PD requester state diagram – triggers	139
Table A.12 – PD requester state diagram – actions	139
Table A.13 – PD requester state diagram – states	139
Table A.14 – PD subscriber state diagram – triggers	140
Table A.15 – PD subscriber state diagram – guards	140
Table A.16 – PD subscriber state diagram – actions	141
Table A.17 – PD subscriber state diagram – states	141
Table A.18 – MD-PDU parameters	145
Table A.19 – TRDP service primitives – Caller	147
Table A.20 – TRDP service primitives – Replier	148
Table A.21 – Topography counter check	152
Table A.22 – MD caller state diagram – triggers	154
Table A.23 – MD caller state diagram – guards	154
Table A.24 – MD caller state diagram – actions	155
Table A.25 – MD caller state diagram – states	155
Table A.26 – MD replier state diagram – triggers	158
Table A.27 – MD replier state diagram – guards	158
Table A.28 – MD replier state diagram – actions	158
Table A.29 – MD replier state diagram – states	159
Table A.30 – MD receiver state diagram – triggers	160
Table A.31 – MD receiver state diagram – guards	160
Table A.32 – MD receiver state diagram – actions	161
Table A.33 – MD receiver state diagram – states	161
Table B.1 – Deployed measures to communication errors	165

Table B.2 – SDSINK state diagram – triggers	176
Table B.3 – SDSINK state diagram – guards	176
Table B.4 – SDSINK state diagram – operations	176
Table B.5 – SDTV2 statistic counters	184
Table C.1 – Attributes for device tag	189
Table C.2 – Attributes for device-configuration tag	189
Table C.3 – Attributes for bus-interface tag	191
Table C.4 – Attributes for trdp-process tag	191
Table C.5 – Default values for thread/task	192
Table C.6 – Attributes for pd-com-parameter tag	192
Table C.7 – Default values for pd-com-parameter	193
Table C.8 – Attributes for md-com-parameter tag	194
Table C.9 – Default values for md-com-parameter	195
Table C.10 – Attributes for telegram tag	196
Table C.11 – Attributes for md-parameter tag	197
Table C.12 – Attributes for pd-parameter tag	198
Table C.13 – Attributes for source tag	199
Table C.14 – Attributes for destination tag	200
Table C.15 – Attributes for sdt-parameter tag	200
Table C.16 – Default values for sdt-parameter tag	201
Table C.17 – Attributes for mapped-device tag	202
Table C.18 – Attributes for mapped-bus-interface tag	203
Table C.19 – Attributes for mapped-telegram tag	203
Table C.20 – Attributes for mapped-pd-parameter tag	203
Table C.21 – Attributes for mapped-source tag	203
Table C.22 – Attributes for mapped-destination tag	204
Table C.23 – Attributes for mapped-SDTV2-parameter tag	204
Table C.24 – Attributes for com-parameter tag	205
Table C.25 – Default communication parameters	205
Table C.26 – Basic data types	206
Table C.27 – Attributes for data-set tag	207
Table C.28 – Attributes for element tag	208
Table C.29 – Use of element array size	209
Table F.1 – Conformance testing summary	262

INTERNATIONAL ELECTROTECHNICAL COMMISSION

**ELECTRONIC RAILWAY EQUIPMENT –
TRAIN COMMUNICATION NETWORK (TCN) –****Part 2-3: TCN communication profile****FOREWORD**

- 1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
- 2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
- 3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
- 4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
- 5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
- 6) All users should ensure that they have the latest edition of this publication.
- 7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
- 8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
- 9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 61375-2-3 has been prepared by IEC technical committee 9: Electrical equipment and systems for railways.

This bilingual version (2019-01) corresponds to the monolingual English version, published in 2015-07.

The text of this standard is based on the following documents:

FDIS	Report on voting
9/2029/FDIS	9/2048/RVD

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

The French version of this standard has not been voted upon.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 2.

A list of all parts in the IEC 61375 series, published under the general title *Electronic railway equipment – Train communication network (TCN)*, can be found on the IEC website.

The committee has decided that the contents of this publication will remain unchanged until the stability date indicated on the IEC website under "<http://webstore.iec.ch>" in the data related to the specific publication. At this date, the publication will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

The contents of the corrigenda of December 2015 and October 2016 have been included in this copy.

IMPORTANT – The 'colour inside' logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.

INTRODUCTION

The IEC 61375 standard series specifies a Train Communication Network for usage in railway vehicles (trains) mainly intended for the exchange of TCMS related information, but not restricted to it. The specification starts from the physical layer up to the application layer and it involves different communication technologies.

This part of IEC 61375 (IEC 61375-2-3) defines the communication profile of the Train Communication Network so as to achieve interoperability between Consists connected by Ethernet Train Backbones as defined in IEC 61375-2-5.

The reasons for prompting the preparation of this part of IEC 61375 are:

- definition of the requirements necessary for communication interoperability on Ethernet Train Backbone level;
- full documentation of the requirements of all users, aligning them and setting them out in standard form;
- providing guidelines for the technical solution adopted for the train backbone interoperable communication;
- defining a conformance test guideline (Annex F) which gives guidance for checking the conformity of consists to the communication profile.

Concrete train applications for certain functionalities are not dealt with in this part of IEC 61375. They are contained in IEC 61375-2-4.

IECNORM.COM : Click to view the full PDF of IEC 61375-2-3:2015

ELECTRONIC RAILWAY EQUIPMENT – TRAIN COMMUNICATION NETWORK (TCN) –

Part 2-3: TCN communication profile

1 Scope

This part of IEC 61375 specifies rules for the data exchange between consists in trains. The aggregation of these rules defines the TCN communication profile.

The objective of the communication profile is to ensure interoperability between consists of the said trains with respect to the exchange of information. For this it defines all those items which are necessary for communication interoperability:

- an architecture with defined train directions related to different train views;
- a common functional addressing concept;
- common communication protocol for data exchange between functions;
- a set of services for train communication control.

As a restriction, this communication profile is adhered to the Ethernet Train Backbone (ETB) technology as defined in IEC 61375-2-5. Towards the consist networks, a more abstract interface is defined which does not restrict the appliance of any consist network technology as for instance MVB (IEC 61375-3-1), CANOpen (IEC 61375-3-3) or ECN (IEC 61375-3-4).

It is not within the scope of the communication profile to define application data content and its meaning (e.g. syntax and semantics). This is within the responsibility of the application profiles. Namely two application profiles are explicitly supported as shown in Figure 1: the TCMS application profile as defined in IEC 61375-2-4, and the onboard multimedia and telematic services (OMTS) related profiles as defined in the IEC 62580 series.

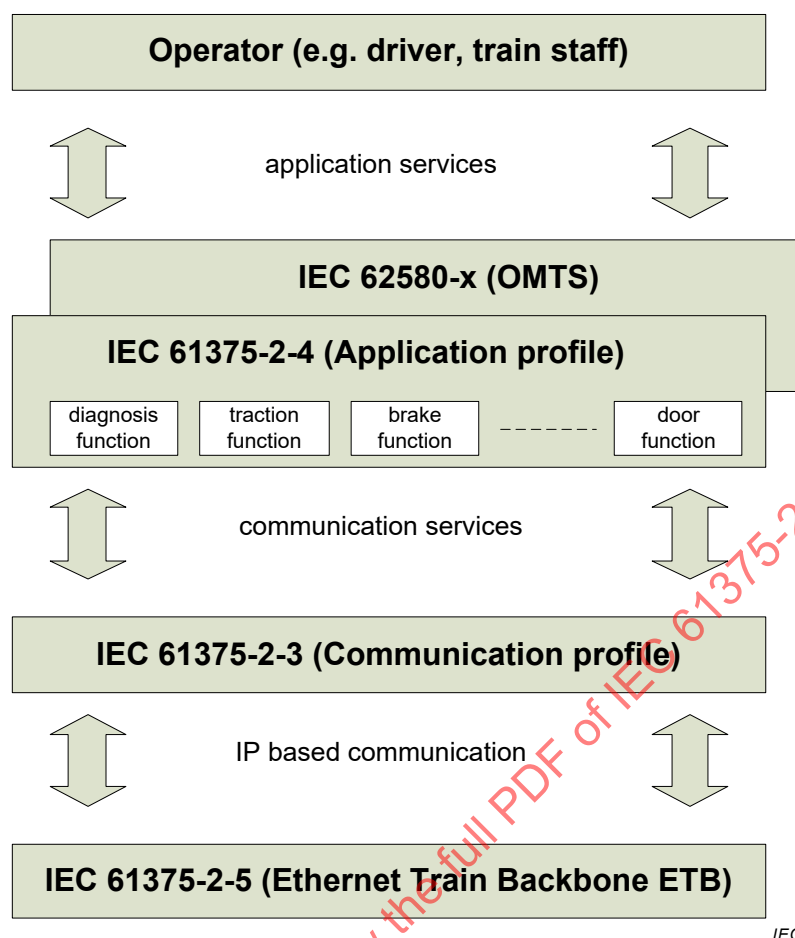


Figure 1 – IEC 61375-2-3 as connecting element between train backbone and application

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 61375-1, *Electronic railway equipment – Train communication network (TCN) – Part 1: General architecture*

IEC 61375-2-1, *Electronic railway equipment – Train communication network (TCN) – Part 2-1: Wire Train Bus (WTB)*

IEC 61375-2-4, *Electronic railway equipment – Train communication network (TCN) – Part 2-4: Application Profile*

IEC 61375-2-5, *Electronic railway equipment – Train communication network (TCN) – Part 2-5: Ethernet train backbone*

IEC 62280, *Railway applications – Communication, signalling and processing systems – Safety related communication in transmission systems*

ISO/IEC 9646-6:1994, *Information technology – Open Systems Interconnection – Conformance testing methodology and framework – Part 6: Protocol profile test specification*

ISO/IEC 9646-7:1995, *Information technology – Open Systems Interconnection – Conformance testing methodology and framework – Part 7: Implementation Conformance Statements*

ISO/IEC 17011:2004, *Conformity assessment – General requirements for accreditation bodies accrediting conformity assessment bodies*

ISO/IEC 17025:2005, *General requirements for the competence of testing and calibration laboratories*

3 Terms, definitions, abbreviations, acronyms, and conventions

3.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1.1

application layer

upper layer in the OSI model, interfacing directly to the application

3.1.2

application layer interface

definition of the services offered by the application layer

3.1.3

application process

element within a real open system which performs the information processing for a particular application

3.1.4

big endian

definition where the most significant byte of multi byte data is stored in the lowest address memory location and sent first in the sequence of serialized transmissions

3.1.5

bridge

device which stores and forwards frames from one bus to another on the base of their Link Layer addresses

3.1.6

broadcast

nearly simultaneous transmission of the same information to several destinations

Note 1 to entry: Broadcast in the TCN is not considered reliable, i.e. some destinations may receive the information and others not.

3.1.7

closed train

train composed of one or a set of Consists, where the composition does not change during normal operation

EXAMPLE metro, sub-urban train, high speed train units.

3.1.8

closed train consist

consist within a closed train

3.1.9**communication devices**

devices connected to Consist Network or Train Backbone with the ability to transport, source or sink data

3.1.10**composition**

number, sequence, orientation and characteristics of the consists forming a train

3.1.11**configuration**

definition of the topology of a network, the devices connected to it, their capabilities and the traffic they produce

3.1.12**consist**

single vehicle or a group of vehicles which are not separated during normal operation

Note 1 to entry: A consist contains no, one or several consist networks.

3.1.13**consist network**

communication network interconnecting communication devices in one Consist

3.1.14**consist network address**

network address, which does not change after inauguration and which is used to address communication device in the own consist network

3.1.15**consist sequence number**

sequence number of the Consist in the train as obtained during TCN inauguration

3.1.16**consumer**

receiver of a message at the Transport Layer

3.1.17**destination device**

receiver of a data packet

3.1.18**discovered vehicle/consist**

vehicle/consist discovered during train inauguration

3.1.19**end device**

unit connected to one Consist Network or to one set of Consist Networks prepared for redundancy reasons

3.1.20**Ethernet Train Backbone**

train backbone based on switched Ethernet technology

Note 1 to entry: Specified in IEC 61375-2-5.

3.1.21

Ethernet Train Backbone Node

node connected to the Ethernet train backbone receiving a sequence number during inauguration and capable of forwarding user data packets between Consist Network and Train Backbone

3.1.22

function

application process which exchanges messages with another application process

3.1.23

gateway

connection between different communication technologies

3.1.24

group address

address of a multicast group to which a device belongs

3.1.25

integrity

property of a system to recognize and to reject wrong data in case of malfunction of its parts

3.1.26

leading vehicle

vehicle which controls the movements of the train

3.1.27

linear topology

topology where the nodes are connected in series, with two nodes each connected to only one other node and all others each connected to two other nodes (that is, connected in the shape of a line)

3.1.28

Local Area Network

LAN

part of a network characterized by a common medium access and address space

3.1.29

medium access control

MAC

sublayer of the Link Layer, which controls the access to the medium (arbitration, mastership transfer, polling)

3.1.30

medium

physical carrier of the signal: electrical wires, optical fibers, etc.

3.1.31

message

data item transmitted in one or several packets

3.1.32

multicast

transmission of the same message to a group of receivers, identified by their Group Address

Note 1 to entry: The word "multicast" is used even if the group includes all receivers.

3.1.33**multimedia network**

network in a train which is used for data transmission of multimedia and telematic data

3.1.34**multimedia services**

services which are used for multimedia

Note 1 to entry: Onboard multimedia and telematic services are defined in IEC 62580-1.

3.1.35**multiple unit**

number of vehicles put together to form a unit for the conveyance of passengers and/or freight

Note 1 to entry: The driving cabs of the multiple unit can be located in special power vehicles (e.g. locomotives with only one cab) or in end vehicles fitted with driving cabs. Groups of vehicles of the multiple unit can be assembled into consists.

3.1.36**multiple unit train**

train consisting of a set of closed trains, where the composition of the set may change during normal operation

3.1.37**network**

set of possibly different communication systems which interchange information in a commonly agreed way

3.1.38**network address**

address which identifies a communication device on network layer

3.1.39**network device**

components used to set up Consist Networks and Train Networks

Note 1 to entry: These may be passive components like cables or connectors, active unmanaged components like repeaters, media converters or (unmanaged) switches, or managed active components like gateways, routers and (managed) switches.

3.1.40**network layer**

layer in the OSI model responsible for routing between different busses

3.1.41**network management**

operations necessary to remotely configure, monitor, diagnose and maintain the network

3.1.42**node**

device on the Train Backbone, which may act as a gateway between Train Backbone and Consist Network

3.1.43**operational network**

network in a train which is used for the transmission of train operation and safety critical data

3.1.44

operational train view

driver or train attendance view of a train as a sequence of vehicles

3.1.45

octet

byte

8-bit word stored in memory or transmitted as a unit

3.1.46

onboard multimedia and telematics subsystem

collection of train on-board services and functions for multimedia and telematics

Note 1 to entry: A definition of those services is given in IEC 62580.

3.1.47

operational services

all services which are needed to run a train safely

3.1.48

operational train direction

direction of train as seen by driver or train attendance

3.1.49

operational train directory

data structure describing the train composition as seen by driver or train attendance, including information about the leading vehicle and vehicle gaps

Note 1 to entry: The operational train directory is a part of the train topology database.

3.1.50

operator

enterprise or organization which is operating trains

3.1.51

packet

unit of a message (information, acknowledgement or control) transmitted by protocols on network or transport layer

3.1.52

PD-PDU

data unit for the transmission of process data

Note 1 to entry: There are two types of PD-PDUs. All types of PD-PDUs have the same PDU format. PD-PDU(data) is a PD-PDU containing user data and sent from a publisher to subscriber(s). This is used in push pattern and in pull pattern. PD-PDU(data) in pull pattern is also referred as PD-PDU(reply). PD-PDU(request) is a PD-PDU sent from a requester to publisher(s). This may or may not contain user data.

3.1.53

PD publisher

device which cyclically sends PD telegrams with a defined ComId to a defined address or address range

3.1.54

PD requester

device which requests a single PD telegram of a specified ComId to be sent to a specified address or address range

3.1.55**PD subscriber**

device which receives PD telegrams with a defined ComId and a possible source address filter

3.1.56**producer**

sender of a message at the Transport Layer

3.1.57**publisher**

source of a dataset for broadcasting (see subscriber)

3.1.58**receiver**

electronic device which may receive signals from the physical medium

3.1.59**redundancy group**

group of devices, where only one device is active and the others are in standby

Note 1 to entry: If the active device fails, one of the standby devices takes over. Typically, a redundancy group comprises two devices.

3.1.60**regular transmission**

data transmission of non safety related data

3.1.61**repeater**

connection at the Physical Layer between bus segments, providing an extension of the bus beyond the limits permitted by passive means

Note 1 to entry: The connected segments operate at the same speed and with the same protocol. The delay introduced by a repeater is in the order of one bit duration.

3.1.62**residual error rate**

probability of integrity breach (unrecognized wrong bit) per transmitted bit

3.1.63**ring topology**

active network where each node is connected in series to two other nodes

Note 1 to entry: Ring may also be referred to as loop.

[SOURCE: IEC 61918:2013,3.1.63]

3.1.64**router**

connection between two buses at the Network Layer, which forwards telegrams from one bus to another on the base of their network address

3.1.65**safe data transmission**

data transmission of safety related data

3.1.66**service**

capabilities and features of a sub-system (provided to a user)

3.1.67

sporadic data

transmission of data on a demand basis

3.1.68

subscriber

one of the sinks of a broadcast dataset (see publisher)

3.1.69

switch

MAC bridge

Note 1 to entry: As defined in IEEE 802.1D.

3.1.70

topology

possible cable interconnection and number of devices in a given network

3.1.71

topography counter

unique identifier of a train composition

3.1.72

train

composition of one or a set of Consists, where the configuration can change during normal operation

3.1.73

train communication network

data communication network for connecting programmable electronic equipment on-board rail vehicles

3.1.74

train communication system

train communication network plus its communication and application profile for the exchange of information trainwide

3.1.75

train control and monitoring system

collection of train on-board services and functions for the operation of a train.

Note 1 to entry: A definition of those services is given in IEC 61375-2-4.

3.1.76

train backbone

bus connecting the consists of a train

Note 1 to entry: Conforms to IEC 61375-2-1 (WTB) or IEC 61375-2-5 (ETB).

3.1.77

train backbone node

device on the train backbone which receives a train backbone node number during inauguration.

Note 1 to entry: A train backbone node can be used to connect an End Device or a Consist Network to the Train Backbone.

3.1.78**train backbone node number
node address
node number**

number assigned to each train backbone node during inauguration, which indicates the position of the train backbone node on the train backbone

3.1.79**train backbone view**

train seen as a sequence of consist networks determined by ETB inauguration

3.1.80**train directory**

data structure describing the train composition after a train inauguration

Note 1 to entry: The train directory is a part of the train topology database.

3.1.81**train network address**

dynamic network address, which is used to address communication devices in other consist networks.

Note 1 to entry: This address can change after each inauguration.

3.1.82**train network management**

services of the network management for TCN

3.1.83**train directory computation**

protocol to determine the actual train composition as a set of consists containing vehicles containing function devices

Note 1 to entry: Optional closed trains represented as consists can be detected and interpreted as a set of consists. Result of the train directory computation is the train directory data structure which is a part of the train topology database.

3.1.84**train reference direction**

direction determined by inauguration

3.1.85**train topology database**

data base collecting all the information about the actual train composition

Note 1 to entry: An instance of this database is computed within each consist of the train.

3.1.86**train view**

train seen as sequence of consists determined by the ETB inauguration

3.1.87**transport layer**

layer of the OSI model responsible for end-to-end flow control and error recovery

3.1.88**UIC vehicle identification number**

12-digit number for the unique identification of vehicles

Note 1 to entry: As defined by the UIC.

3.1.89

unspecified PDU

TRDP PDU with ComId=0

Note 1 to entry: The user data content of an unspecified PDU is undetermined for the communication profile. Different PDUs and their content can be distinguished only at application level.

3.1.90

vehicle

single coach or locomotive

3.1.91

vehicle gap

gap

vehicle listed in the confirmed vehicle list of the operational train directory which is not reachable over ETB

Note 1 to entry: A vehicle gap can be at one train end or in the middle of a train. There may also exist more than one vehicle gap in a train.

Note 2 to entry: Vehicle gaps may be determined by the driver while comparing the indicated unconfirmed train composition with the real train composition, leading to a correction of the train composition.

3.2 Abbreviations and acronyms

ALG	Application Layer Gateway
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
CN	Consist Network
ComId	Communication Identifier
CRC	Cyclic Redundancy Check
CST	Consist
CSTINFO	Consist Information telegram
DC	Direct Current
DEV	Device
DNS	Domain Name System
DoS	Denial of Service
FCS	Frame Check Sequence
ECN	Ethernet Consist Network
ECSC	ETB Control Service Client
ECSP	ETB Control Service Provider
ED	End Device
ETB	Ethernet Train Backbone
ETBCTRL	ETB Control packet

ETBN	Ethernet Train Backbone Node
FCT	Function
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IP	Internet Protocol
LEN	Length
MAC	Medium Access Control
MC	Multicast
MD	Message Data
MD-PDU	Message Data Protocol Data Unit
MIB	Management Information Base
MTU	Maximal Transmission Unit
MVB	Multifunction Vehicle Bus
NAT	Network Address Translation
NOK	Not OK
OID	Object Identifier
OMTS	Onboard Multimedia and Telematic Subsystems
OSI	Open System Interconnection
PD	Process Data
PD-PDU	Process Data Protocol Data Unit
PICS	Protocol Implementation Conformance Statement
PROP	Properties
QoS	Quality of Service
RFC	Request For Comments
SDSINK	Safe Data Sink
SDSRC	Safe Data Source
SDTv2	Safe Data Transmission version 2
SID	Source Identifier
SIL	Safety Integrity Level
SMI	Safe Message Identifier
SNMP	Simple Network Management Protocol
SSC	Safety Sequence Counter
STC	Safe Topography Counter

TCN	Train Communication Network
TCMS	Train Control and Monitoring System
TCP	Transmittion Control Protocol
TP	Test Procedure
TSS&TP	Test Suite Structure and Test Purpose
TRDP	Train Real Time Data Protocol
TTDB	Train Topology Database
TTL	Time to Live
TTDP	Train Topology Discovery Protocol
UDP	User Datagram Protocol
UIC	Union Internationale des Chemins de Fer
UML	Unified Modeling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UUID	Universally Unique Identifier
UuT	Unit under Test
VDP	Vital Data Packet
VLAN	Virtual Local Area Network
WTB	Wire Train Bus
XML	Extended Markup Language
XSD	XML Schema Definition

3.3 Conventions

3.3.1 Base of numeric values

This part of IEC 61375 uses a decimal representation for all numeric values unless otherwise noted.

Analog and fractional values include a comma.

EXAMPLE The voltage is 20,0 V.

Binary and hexadecimal values are represented using the ASN.1 (ISO/IEC 8824) convention.

EXAMPLE Decimal 20 coded on 8 bits = '0001 0100'B = '14'H.

3.3.2 Character strings and citations

Character strings are placed within double quotes.

EXAMPLE "ComProfTestApp".

Single quotes are used for citations (quotations).

EXAMPLE parameter 'etbTopoCnt'.

Sometimes character strings are parameter values, in which case they can be written as citations.

EXAMPLE telegram type 'Mn'.

3.3.3 Naming conventions

Keywords are written with a capital letter at the beginning.

If the keyword name is composed, the different parts of the name are united with a space, and all parts begin with a capital letter.

EXAMPLES "Train Backbone", "Consist", "Consist Network".

Parameters are written with a capital letter at the beginning.

If the parameter name is composed, the different parts of the name are united without a space, and all parts begin with a capital letter except for the first part which begins with a lower case letter.

EXAMPLE 'etbTopoCnt'.

3.3.4 Diagram conventions

Whenever appropriate, UML diagram types are used. In particular, the following UML diagram types are applied:

- UML state machines
- UML sequence diagram

For conditions used in these diagram types the following conventions are applied:

operators	=	Assignment operator: set a value
	==	Equality operator: equal
	!=	Equality operator: not equal
	OR	Logical operator: {expression} OR {expression}
	AND	Logical operator: {expression} AND {expression}
boolean	FALSE	Antivalent2 value 0: 01 _b
	TRUE	Antivalent2 value 1: 10 _b

For triggers and guards used in these diagram types the following conventions are applied:

- A trigger is an external event, e.g. a user command, which triggers a state transition
- In the special case that a trigger is the reception of a (periodic) ETBCTRL packet, the control flags within the ETBCTRL packet determine the state transition. For those cases only the guards are indicated in the UML state machines

3.3.5 Annotation of data structures

3.3.5.1 General

Data structures are defined following the presentation and encoding of transmitted and stored data as ruled in IEC 61375-2-1, which bases on ISO ASN.1 syntax.

All data within a data structure are organized in big-endian format (most significant octet of a data item first).

All data within a data structure are naturally aligned (stored at offset address which is a multiple of their size).

Data values of parameters within data or telegram structures which are marked 'application defined' are either defined by IEC 61375-2-4 or are left to the user to be defined.

3.3.5.2 Additional data types

In addition to the data types defined in IEC 61375-2-1, the following data types are used:

- (1) LABEL ::= ARRAY [16] OF CHAR
- (2) AntivalentType ::= ANTIVALENT8

with the encoding:

7	6	5	4	3	2	1	0	interpretation
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
0	0	0	0	0	0	0	1	FALSE
0	0	0	0	0	0	1	0	TRUE
0	0	0	0	0	0	1	1	UNDEFINED
any other value								ERROR

3.3.5.3 Data type keywords

In addition to the keywords/notations defined in IEC 61375-2-1 as an extension to ASN.1, some shorter, but equivalent keywords/notations are used within this part of IEC 61375 in order to improve the readability of data structures. These keywords are (see Table 1):

Table 1 – Data type keywords and notations

Keyword/notation used in this part of IEC 61375	Equivalent keyword/notation used in IEC 61375-2-1
UINT#	UNSIGNED#
INT#	INTEGER#
CHAR	CHARACTER8
Type [IndexList]	ARRAY [IndexList] OF Type
(# = any unsigned integer)	

4 Architecture

4.1 General

This clause defines the train related communication architecture, which is the basis for the communication profile. The train related communication architecture is defined from two different viewpoints, the physical train architecture, and the logical train architecture. The physical train architecture provides a system breakdown by considering components and interfaces in between, whereas the logical train architecture is based on functional breakdown dealing with functions and their interactions. Both views are necessary for defining important communication profile concepts, like for instance the definition of the train topology database or the definition of a train wide logical addressing concept based on URIs.

4.2 Physical train architecture (system breakdown)

4.2.1 General

The physical train architecture defines a train as composed of closed trains, consists, vehicles, devices and the interfaces in between.

4.2.2 Train network architectures

4.2.2.1 Train structure

IEC 61375-1 defines a train as a composition of closed trains and consists, each consist having one or several vehicles, and each closed train having one or several consists as shown in Figure 2.

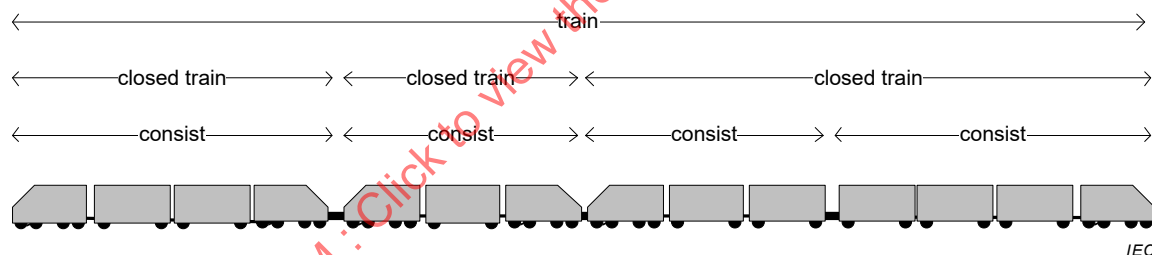


Figure 2 – Train structure in accordance to IEC 61375-1 (example)

For this part of the standard, the following restriction is made:

Although a closed train is a composition of one or several consists (closed train consists), it shall be seen as one operational consist for the scope of this communication profile. In this sense, a train is only composed of (operational) consists as shown in Figure 3, whereby some of the operational consists may be a closed train. Consequently, every closed train has also to behave like a consist with respect to this communication profile.

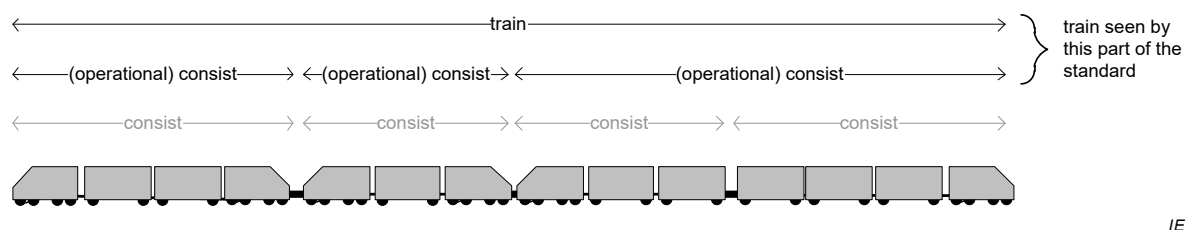


Figure 3 – Train structure seen from viewpoint of the communication profile (example)

NOTE 1 Communication inside a closed train, e.g. between the closed train consists which constitute the closed train, is not in the scope of this standard.

This communication profile is applicable for trains with a maximum number of 63 vehicles.

NOTE 2 How vehicles are allocated to consists depends on the chosen consist design. For example, each vehicle in a train can be a consist, in which case a train may also contain 63 consists, or two trainsets with 32 and 31 vehicles form each one consist, in which case the train contains only 2 consists.

4.2.2.2 Train network allocation

IEC 61375-1 defines a train wide network as a composition of consist networks interconnected by a common train backbone as shown in the example of Figure 4, where one or several consist networks per consist are connected to the ETB via a redundant pair of ETBN.

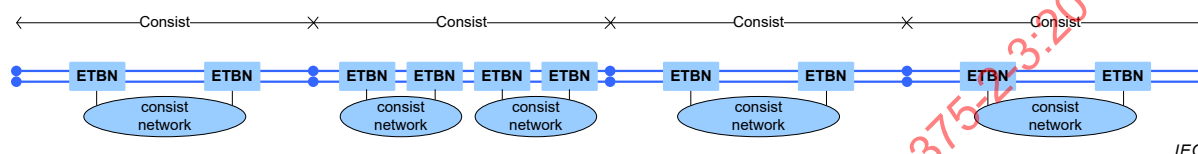


Figure 4 – Train network (example)

Furthermore, IEC 61375-1 allows applying up to 4 train backbones in parallel.

This communication profile defines the usage of the train backbones in the following way:

- One ETB (with subordinated consist networks) shall be used as operational network, supporting TCMS services.
- No, one or several ETBs (with subordinated consist networks) shall be used as multimedia network supporting OMTS services

The operational ETB may also be used for OMTS services.

NOTE 1 To have TCMS services and OMTS services sharing one (operational) ETB may be used for simple applications.

NOTE 2 If one physical network is used for both TCMS services and OMTS services, a separation can be done on an upper layer, e.g. on link layer by using VLAN. The expectation is, however, that requirements related to safety and security are much harder to fulfill for such type of network.

4.2.2.3 ETBN redundancy

At least one ETBN shall be provided per consist network. For redundancy purposes, it is recommended that two ETBNs are at least provided on traction units and for the backbone connection of the driving cab equipment of driving trailers (this also applies to multiple units).

4.2.2.4 Safety and security aspects

The operational network shall be a category 1 (preferred) or a category 2 transmission system as defined by IEC 62280.

4.2.2.5 IP Address space

Both operational network and multimedia network shall share one common train network address space as defined in IEC 61375-2-5 (i.e. 10.128/9). The train network address space for each network shall be identified.

NOTE If both networks are physically separated, the backbone id, 10th and 11th bit in train network address, is used for identification.

4.2.2.6 Interconnecting operational network and multimedia network

4.2.2.6.1 ETB Dependency

In order to ensure interoperability between trains with multiple ETBs, all functions connected to those ETBs shall share a common train view and a common operational train view as defined in 4.2.4.1.

NOTE The impact of this requirement on the TTDB is defined in 5.3.4.2.

4.2.2.6.2 Network couplings

There are several ways of coupling operational network and multimedia network (see Figure 5).

- Both networks can share a common CN, in which case the two networks are coupled on OSI layer 2.
- Both networks can be interconnected by an IP router (eventually with firewall functions), thus providing a coupling on OSI layer 3.
- Both networks can be separated by an application layer gateway (ALG), meaning a coupling on OSI layer 7.

NOTE Modern IP routers with firewall functionality allow also a filtering on (application data) content. In this case the networks are coupled on both layer 3 and layer 7.

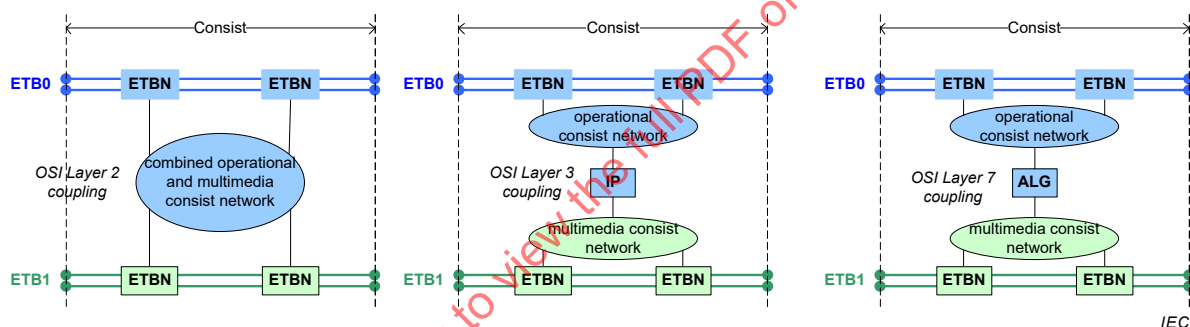


Figure 5 – Possible couplings of operational network and multimedia network

If there is a coupling on at least OSI layer 3, EDs connected to the operational consist network are said to be related to the operational ETB (ETB0), and EDs connected to the multimedia consist network are said to be related to the multimedia ETB (ETB1).

In case of OSI layer 2 coupling this relationship is undetermined.

NOTE The relationship of EDs is an information that is relevant for IP MC group assignments, see 5.4.5.2.

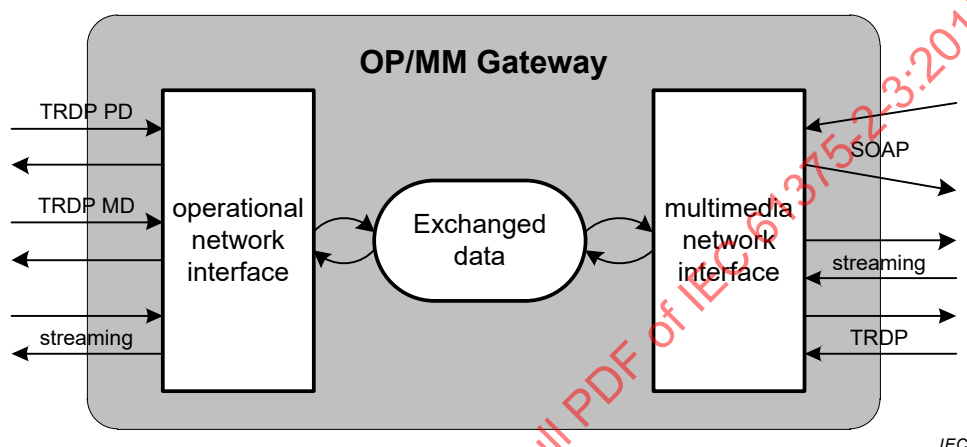
4.2.2.6.3 Data exchange between operational network and multimedia network

The way of data exchange between the operational network and multimedia network depends on the coupling of the two networks. While couplings on OSI layer 2 and 3 allow a data exchange on IP telegram base that is mostly handled within the communication layers of modern operating systems, a coupling on OSI layer 7 provides a better application control of exchanged data.

The following example describes a possible ALG design of an operational/multimedia gateway:

EXAMPLE

The gateway for interconnecting the operational network and the multimedia network (Figure 6) belongs formally to the operational network. This gateway is not only needed for the reasons of security and closed system. It also serves as entry point for multimedia network applications to operational network applications and vice versa, offering services to the respective side, which allows exchanging information with the respective, other side. This also means that the gateway has to provide two different interfaces: the interface to operational section based on the communication protocols used in the operational section, and the interface to the multimedia section implementing the protocols of the multimedia section. As an example, the gateway may receive the train directory information from the operational section ECSP within a TRDP Message Data packet, and offers this information in a web service to the multimedia section participants. Other interface service examples are to provide data streaming (video and audio) or direct exchange of TRDP data packets. So both client/server related interface services as well as real-time data interface services might be needed. As real-time data requires a high throughput, those data are normally routed on OSI layer 3 rather than processed on layer 7. In order to cope with the security, additional security measures are then required as for instance firewall functionality.



IEC

Figure 6 – Gateway between operational network and multimedia network (example)

4.2.2.7 Safety and security aspects

An architecture with a coupled operational network and multimedia network shall not compromise the transmission system categorization of the operational network as defined in 4.2.2.4.

In case the multimedia network is a category 3 transmission system in accordance to IEC 62280, at least a coupling on OSI layer 3 with firewall capabilities shall be envisaged, better a coupling on OSI layer 7.

It is not within the scope of this standard to define the necessary capabilities of the firewall as this is highly related to the system design and its security requirements.

The suitability of the coupling to preserve the transmission system categorization of the operational network shall be demonstrated within the system safety case.

4.2.2.8 Network Reconciliation

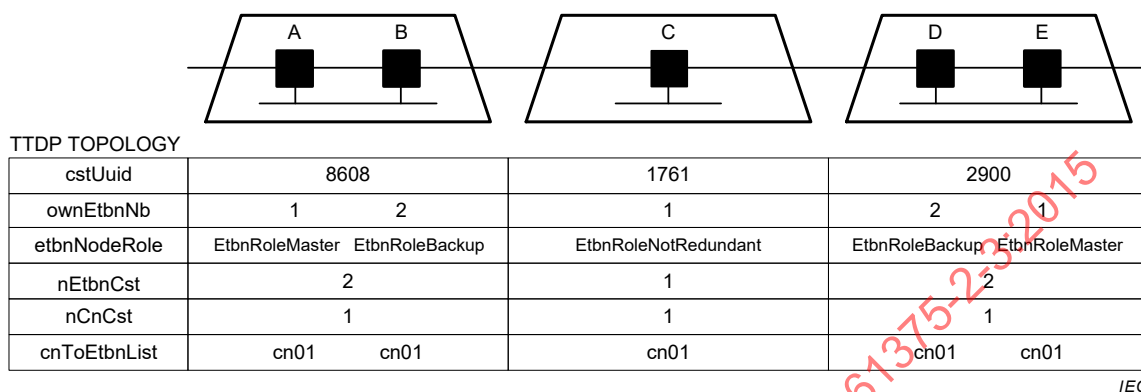
For applications of the multimedia network, it will be necessary to have the same operational train view (see 4.2.4.1) as for an application of the operational network, such as the sequence of consists and the position of the leading vehicle, in order to realize an unambiguous addressing scheme within the train.

As a common rule, the operational train view determined by the operational network and manifested within the operational train directory of the TTDB as defined in 5.3 shall be authoritative for the multimedia network. Hence, consist sequence and position of the leading vehicle shall be adopted from the operational network.

4.2.3 Closed Trains

4.2.3.1 General

As defined in IEC 61375-1, a closed train is composed by consists. The example in Figure 7 shows three coupled consists with UUID 8608, 1761, 2900. Each consist takes part in the ETB inauguration as defined in IEC 61375-2-5.



NOTE The table TTDP TOPOLOGY in Figure 7 contains only the relevant elements.

Figure 7 – Example: three coupled Consists

Consist with UUID 8608 contains two ETBNs “A” and “B” with the static relative positions 1 and 2 in the consist. ETBNs “A” and “B” are redundant. Both ETBNs are connected to the first and only consist network.

Consist with UUID 1761 contains the ETBN “C” with the static relative position 1 in the consist. ETBN “C” is non-redundant. The ETBN is connected to the first and only consist network.

Consist with UUID 2900 contains two ETBNs “D” and “E” with the static relative positions 1 and 2 in the consist. ETBNs “D” and “E” are redundant. Both ETBNs are connected to the first and only consist network.

4.2.3.2 Closed Train Representation in ETB inauguration

For the ETB inauguration according to IEC 61375-2-5, the closed train shall be represented as a single consist.

For the composition of a closed train the parameters for the TTDP TOPOLOGY frame contents as defined in IEC 61375-2-5 shall be adapted. Figure 8 shows the external view of the closed train example and the relevant TTDP TOPOLOGY frame elements.

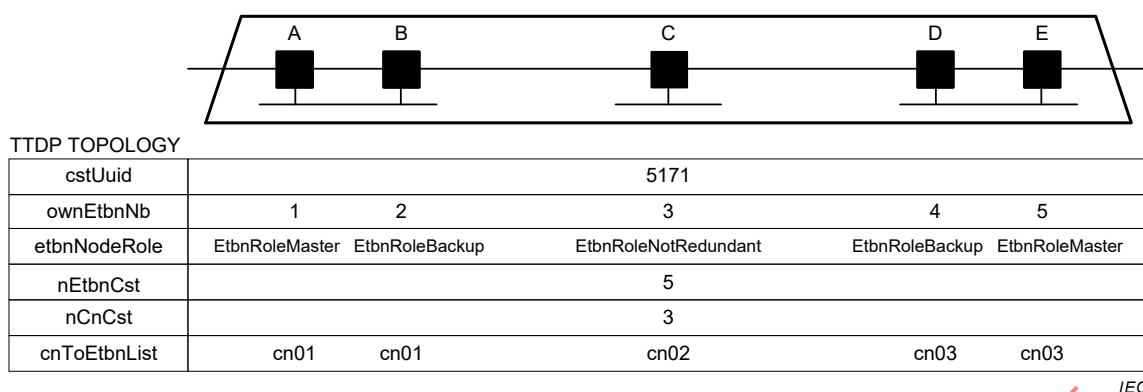


Figure 8 – Example: Closed Train

In the example in Figure 8 the consist with UUID 5171 contains the five ETBNs “A”, “B”, “C”, “D” and “E” from Figure 7 with the static relative positions 1, 2, 3, 4 and 5 in the Consist, respectively. ETBNs “A”/“B” and “D”/“E” are redundant. The total number of ETBNs in the Consist is five. The total number of Consist Networks in the Consist is three. The ETBNs “A” and “B” are connected to the first Consist Network. The ETBN “C” is connected to the second Consist Network. The ETBNs “D” and “E” are connected to the third Consist Network.

During ETB inauguration each ETBN of the closed train (i.e. consist with UUID 5171) sends its TTDP TOPOLOGY frame. Therefore another externally coupled consist knows all about the network topology of the closed train.

4.2.3.3 Deficit of information about train composition

Since the closed train in Figure 8 is represented as a single consist, the information about the mapping from ETBNs and connected consist networks to vehicles is lost. This loss of information can be recovered during computation of the TTDB.

4.2.4 Directions

4.2.4.1 Train views

A train can be viewed from different perspectives, which depend on the role of the viewer.

In the train backbone view, a train is composed of consist networks which are connected to one or more ETBs via ETBNs. This view represents the network topology of the train. A train backbone topology will be changed if ETBNs are connected or removed. This view is defined in IEC 61375-2-5.

In the train view, a train is composed of consists and the consists are composed of vehicles. A TCN train composition will be changed if consists are removed or appended. Its defined extremities determine the directions in the train.

In the operational train view (driver’s and attendance view), a train is composed of a series of consists as well as a series of vehicles. Directions in the train are defined by the main view direction of the driver’s cab in the leading vehicle. An operational train view will be changed if the leading cab is changed or if consists are appended or removed and there is no leading cab.

4.2.4.2 Train reference directions

The TCN directions and orientation within a train, consist and vehicle are specified in IEC 61375-1.

IEC 61375-1 specifies as well the numbering of vehicles in a consist and the numbering of consists within a train. The numbering is related to the Extremity 1 of a vehicle, a consist, or a train.

IEC 61375-1 leaves open how the extremities will be assigned. The following rules for assigning the extremities shall apply:

- Extremity 1 of a vehicle shall be statically predefined by the design of that vehicle.
- Extremity 1 of a consist shall be statically predefined by the design of that consist.
- Extremity 1 of a train is the train end, which is pointed to by the ETB reference direction 1 as defined in IEC 61375-2-5.

4.2.4.3 Operational directions

From operator/driver point of view, one end of a train is the train head and the opposite end the train rear, which are determined by the operational train direction. The operational train direction points from the train rear to the train head. The operational train directions are defined according to the rules a) to e).

- a) If there is a leading vehicle in the train, the operational train direction is determined by the viewing direction of the driving cab of the leading vehicle.

If there is no leading vehicle, then the operational train direction shall be determined by the following rules:

- b) If there was a previously leading vehicle, then this vehicle together with its previously operational train direction determines the operational train direction.
- c) If there were several previously leading vehicles, e.g. after a train lengthening, and all of these have the same viewing direction, then this viewing direction determines the operational train direction.
- d) If there are multiple consists in the train and a traction consist at one end, then this end defines the train head and traction consist includes vehicle 01 and determines the operational train direction.
- e) In all other cases, the operational train direction shall correspond to the ETB reference direction according to IEC 61375-2-5.

4.2.5 Consist and vehicle basic properties

4.2.5.1 General

Consists and vehicles shall be described by a basic set of consist and vehicle properties, which will be stored within the Train topology database. For this a distinction should be made between static, topological and dynamic properties:

- Static properties are features built into the vehicle or consist and thus only dependent on its class or design, e.g. “vehicle identifier”, “vehicle has first class seats”, “traction unit with electric drive”.
- Topological properties are properties, which present the physical configuration of train structure, e.g. vehicle sequence, vehicle orientation.
- Dynamic properties are properties which define a train assignment, e.g. “leading vehicle”, “vehicle sequence number” or “train reference direction”.

4.2.5.2 Vehicle and consist type information

Vehicles and consists may receive a type classifier which makes it easier for applications to check whether vehicles or consists have common static properties. Those type classifiers are defined as follows:

VehType	LABEL	-- Type of the vehicle Zero terminated string
CstType	LABEL	-- Type of the consist Zero terminated string

The definition of type classifiers is application specific. For the scope of this communication profile, the type classifier is only informal.

4.2.5.3 Static Properties

A list of static consist and vehicle properties is provided in IEC 61375-2-4.

4.2.5.4 Identification

A consist shall be uniquely identified by a consist UUID as defined in IEC 61375-2-5.

An application defined vehicle identifier shall uniquely identify a vehicle.

EXAMPLE UIC defined a 12-digit wagon number for the unique identification of coaches, as for example the number "50 80 31 – 34 025 – 2".

4.3 Logical Train Architecture (Functional Breakdown)

4.3.1 General

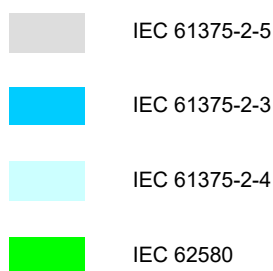
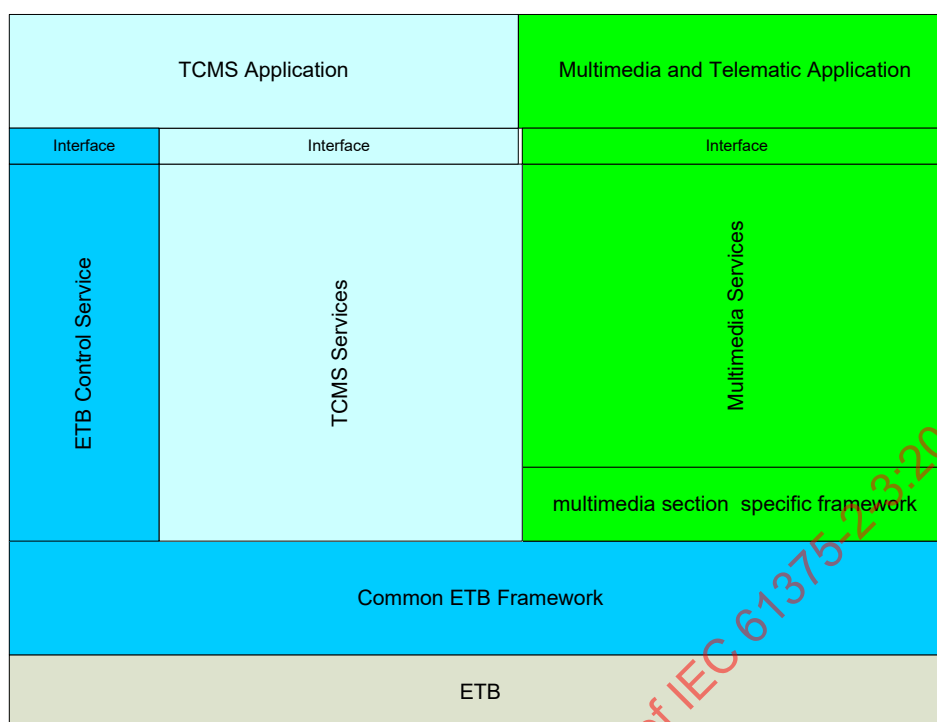
The logical train architecture defines a train as composed of services and the information exchanged between the services and between user and services.

4.3.2 Service classification

Services can be classified in (see Figure 9):

- Operational services, as defined in this part of IEC 61375 and in IEC 61375-2-4.
- Multimedia services, as defined in IEC 62580.

Subject of this part of the standard are ETB control services and shared ETB system services. Operational services are defined in IEC 61375-2-4.



IEC

Figure 9 – Service classification

All the services use functions, protocols, and definitions from a common ETB framework, which is defined in Clause 5.

For multimedia services, a multimedia specific framework may be added which provides multimedia specific functions, protocols, and definitions. The definition of the multimedia specific framework is not within the scope of this standard.

4.3.3 Operational Services Overview

Besides TCMS services as specified in IEC 61375-2-4, this part of the standard defines in Clause 6 an ETB control service whose service functions are listed in Table 2.

NOTE The functional breakdown of TCMS services is related to EN 15380-4.

Table 2 – ETB control service

Function	Short form	Classification	Location	Related subclause	Description
Manage leading vehicle information	Function Leading	control	Consist	6.5	Set or reset leading vehicle on request and handle leading vehicle conflicts
Confirm train configuration	Function Confirmation/Correction	control	Consist	6.6	Allow a user to confirm or to correct the indicated train composition
Manage energy saving mode Manage battery protection mode	Function Sleep Mode	control	Consist	6.8	Set train to low power mode, wake up on command or on ETB activity

4.3.4 Service Provider

The ETB control service shall be provided by the ECSP (see also 6.2).

In consists with several consist networks, the ETB control services shall only be provided within one consist network. This will be the consist network which also hosts the end device responsible for ETB control (ECSC). The reason for this restriction is to avoid control conflicts within a consist.

For redundant ECSP only one ECSP is actively providing the service.

5 Common ETB framework

5.1 General

5.1.1 Overview

The common ETB framework defines basic functions and basic concepts, which together provide the infrastructure the services are bedded in. In particular, it defines:

- CSTINFO Telegram (see 5.2)
- Train topology database TTDB (see 5.3)
- Service addressing (see 5.4)
- TCN-DNS Server (see 5.5)
- Data exchange (see 5.6)
- Service discovery (see 5.6.4)
- Train Info service (see 5.8)

5.1.2 Interoperability

Telegrams exchanged on ETB between consists and their related data structures are versioned using a data structure for version information which is defined as:

```

VERSION ::= RECORD
{
    mainVersion      UINT8      -- main version, incremented for
                                incompatible changes

```

```

        value range: 1..255 (0 = reserved)
subVersion      UINT8      -- sub version, incremented for
                           compatible changes
                           value range: 0..255
    }

```

In order to ensure interoperability between consists, all changes affecting normative clauses of this standard shall be treated as incompatible changes, leading to an increment of parameter 'mainVersion', unless otherwise specified.

This standard does not define a “negotiation” of the supported versions when two consists are coupled implementing different versions of the standard's normative clauses. Such a negotiation would be needed if compatible changes were allowed.

Compatible changes might be applied for informal parts, like for instance the data exchange between ECSC and ECSP defined in Annex E.

5.2 CSTINFO telegram

5.2.1 General

As defined in part IEC 61375-2-5, following an ETB train inauguration, at least one entitled device in each consist shall send out a CSTINFO telegram over ETB to all other entitled devices within the train to inform about consist properties.

For the sake of simplicity, the ECSP shall be entitled to send and receive the CSTINFO telegrams.

5.2.2 Closed train support (Option)

For the ETB inauguration a closed train is represented as a single consist (see 4.2.3.2). Therefore at least one ECSP within the closed train shall send a CSTINFO telegram which describes the closed train as one consist (consist info class 2, see 5.2.4).

In order to learn the consist structure of a closed train, at least one ECSP within a closed train consist shall send a CSTINFO telegram (consist info class 3, see 5.2.4).

5.2.3 Protocol

The protocol for sending CSTINFO telegrams is defined in 5.3.2.

5.2.4 CSTINFO classes

CSTINFO classes are defined to distinguish the CSTINFO telegrams sent by consists, closed trains and closed train consists. The following three classes are defined:

- CSTINFO class 1: CSTINFO telegrams sent by consists which are no closed trains
- CSTINFO class 2: CSTINFO telegrams sent by closed trains
- CSTINFO class 3: CSTINFO telegrams sent by closed train consists

Consists which are no closed trains shall be able to receive and process CSTINFO classes 1 and 2, but need not be able to receive consist info class 3 telegrams.

Only CSTINFO class 2 telegrams shall provide a closed train consist list (array “cltrCstList” CSTINFO telegram). For CSTINFO classes 1 and 3 this array shall be omitted (parameter “cltrCstListCnt” set to 0, see 5.2.5).

5.2.5 CSTINFO Notification Message

CSTINFO messages shall be sent within the payload of a Train Realtime Data Protocol (TRDP) message as defined in Clause A.7.

The CSTINFO size shall be restricted to the maximal DatasetLength parameter value of a MD-PDU as defined in Clause A.7.

TRDP MD Notification telegram parameters:

MsgType:	'Mn'
Protocol:	UDP
ComId:	2
SourceURI (user part):	void
DestinationURI (user part):	void
DestinationURI (host part):	grpECSP.anyVeh.aCst.aCITrn.ITrn
SourceURI (host part):	-
Dataset:	CSTINFO

The TRDP MD UserDataset is defined as (see Figure 10):

```

CSTINFO ::= RECORD
{
  cstInfo      CONSIST INFO  -- consist information as defined in
                             subclause 5.3.3.2.6
}

```

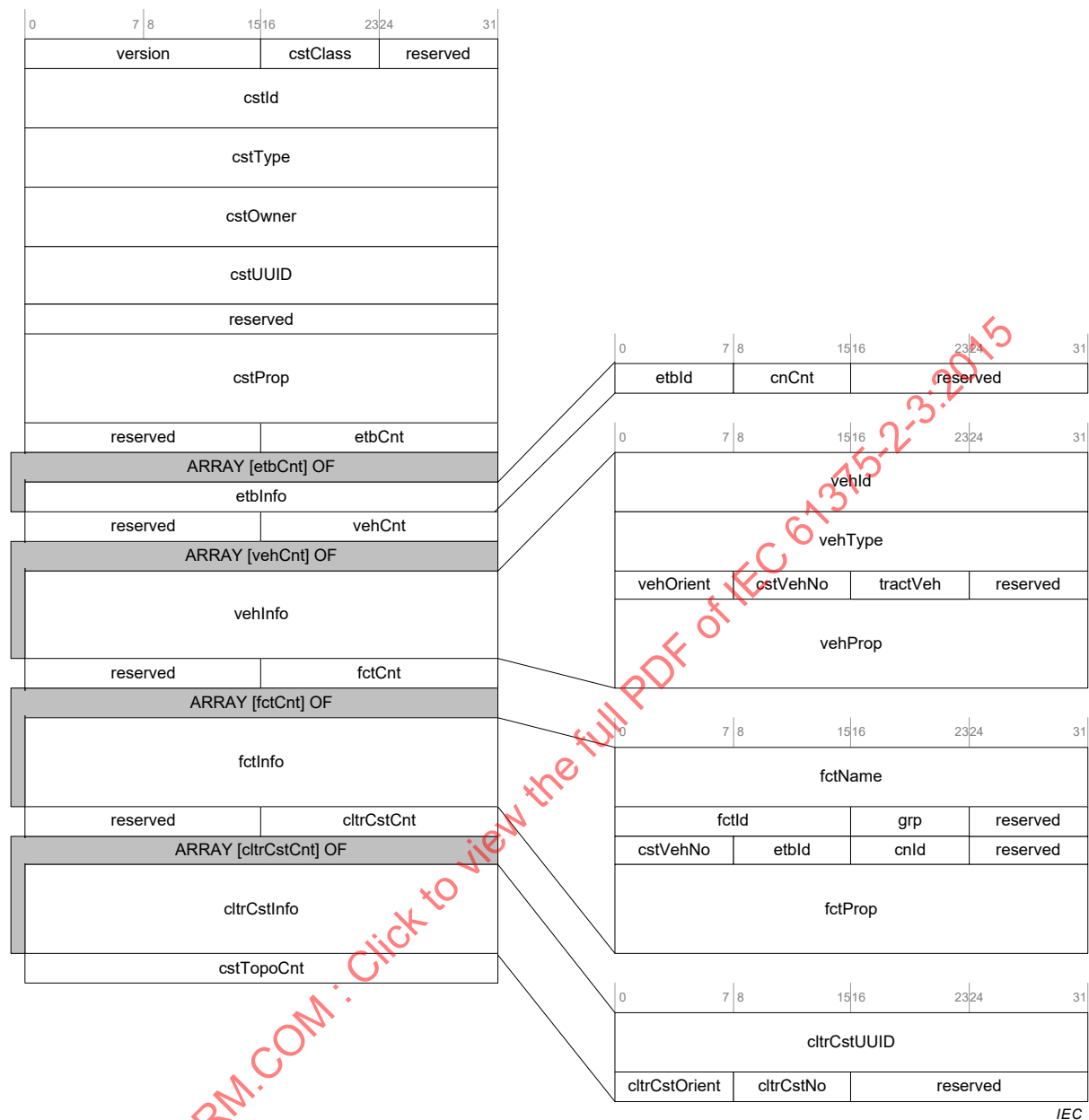


Figure 10 – CSTINFO notification data

5.2.6 CSTINFO Request

CSTINFO telegram sending can be controlled and requested by sending a CSTINFOCTRL notification message as depicted in Figure 11.

The protocol for sending CSTINFOCTRL notification messages is defined in 5.3.2.

CSTINFOCTRL notification message shall be sent within the payload of a Train Realtime Data Protocol (TRDP) message as defined in Clause A.7.

CSTINFOCTRL notification messages support two control modes (differentiated with parameter 'trnCstNo'):

trnCstNo > 0

Used to request the sending of CSTINFO telegrams. The receiving ECSP shall memorize the parameter value

'trnTopoCnt'. This mode shall be used by the ECSP responsible for the CSTINFO exchange within a consist (see 5.3.2.3)

trnCstNo = 0

Used to request the sending of CSTINFO telegrams, but without tracking the 'trnTopoCnt' parameter value. This mode shall be used by all devices interested in CSTINFO except for the ECSP responsible for the CSTINFO exchange.

TRDP MD Notification telegram parameters:

MsgType:	'Mn'
Protocol:	UDP
ComId:	3
SMI:	3 (only needed if SDTv2 is used)
UserDataVersion:	'0100'H (only needed if SDTv2 is used)
SourceURI (user part):	void
DestinationURI (user part):	void
DestinationURI (host part):	grpECSP.anyVeh.aCst.aCITrn.ITrn
SourceURI (host part):	-
Dataset:	CSTINFOCTRL

The TRDP MD UserDataset is defined as (see Figure 11):

```

CSTINFOCTRL ::= RECORD
{
    version      VERSION      -- version of the telegram
                                mainVersion = 1
                                subversion = 0

    trnCstNo     UINT8         -- train consist number
                                0 = without trnTopoCnt tracking
                                > 0 = with trnTopoCnt tracking

    cstCnt       UINT8         -- number of consists in list

    cstList      ARRAY[] OF CONSIST
                                -- consist list.
                                If trnCstNo > 0 this shall be an
                                ordered list starting with trnCstNo = 1
                                (exactly the same as in structure
                                TRAIN_DIRECTORY).
                                If trnCstNo = 0 it is not mandatory to
                                list all consists (only consists which
                                should send CSTINFO telegram).
                                parameters 'trnCstNo' and 'cstOrient'
                                are optional and can be set to 0.

    trnTopoCnt   UINT32        -- trnTopoCnt value

```

```

    trnCstNo > 0: actual value
    trnCstNo = 0: set to 0
safetyTrail ETBCTRL_VDP  -- ETB-VDP trailer
                           parameter 'safeTopoCount' = 0
                           completely set to 0 = SDTv2 not used
}

```

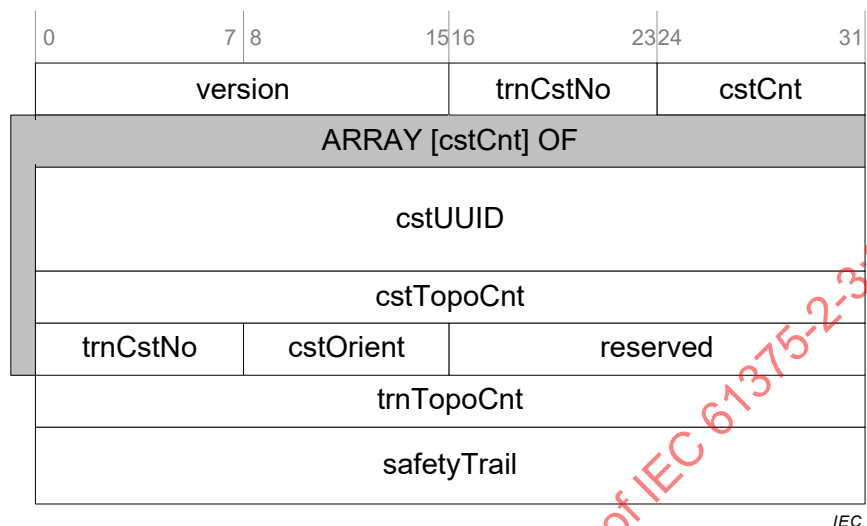


Figure 11 – CSTINFOCTRL telegram

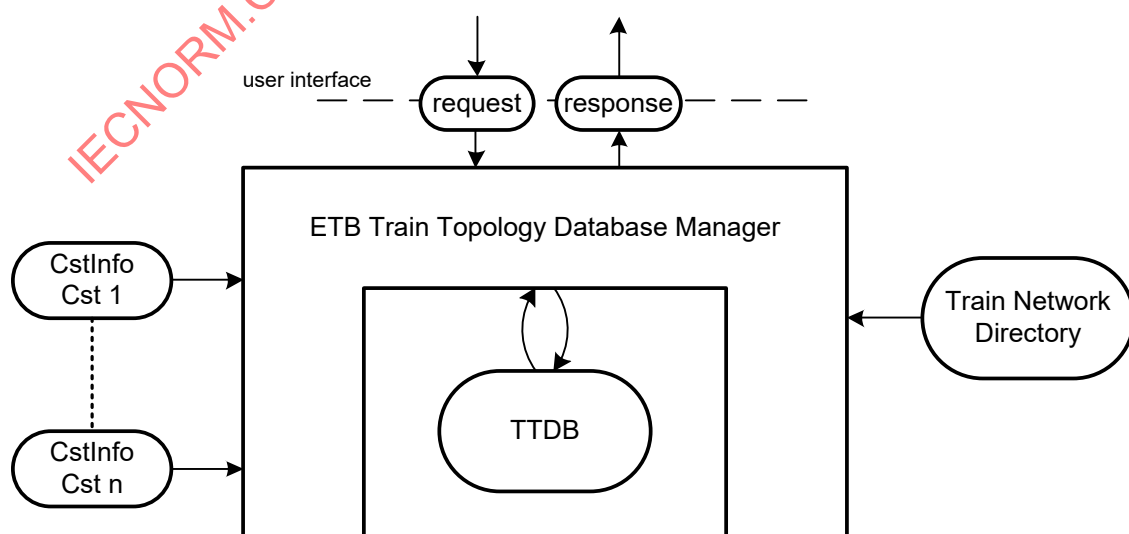
5.3 Train topology database

5.3.1 General

The train topology database (TTDB) is a repository for all the information related to the actual train composition and the actual ETB state.

An instance of the TTDB shall be available at least once in each consist.

NOTE In consists with multiple consist networks it might be an advantage to have multiple TTDB instances, one for each consist network



IEC

Figure 12 – TTDB management block diagram

The TTDB is maintained by a TTDB Manager function as shown in Figure 12, which reads in consist information from other consists (provided by the CSTINFO telegrams) and the result from ETB train inauguration obtained from the ETBN (Train network directory as defined in IEC 61375-2-5) and stores this information to the TTDB. Towards the application services, it provides a user interface, which allows retrieving parts or all the information from the TTDB (see Annex E and Annex G). The TTDB manager shall always keep the TTDB up-to-date.

The TTDB consists of four associated data blocks as it is depicted in Figure 13.

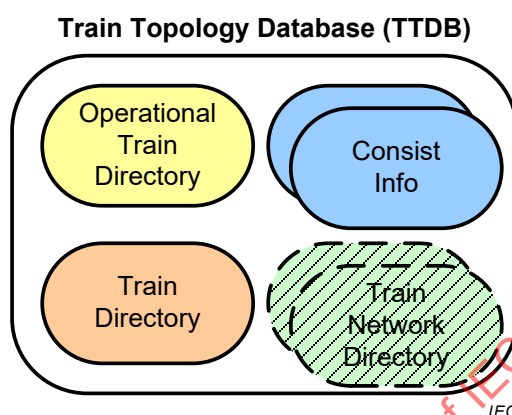


Figure 13 – TTDB Content

Consist Info

Contains the descriptions of each consist as they are distributed after ETB train inauguration within CSTINFO telegrams. This includes also a list of functions provided by the consist and all the consist related information which a TCN-DNS server needs for TCN-URI address resolution. This part is statically preconfigured for each consist. In trains with multiple ETBs, the consist info is available for each ETB and the content of the different consist info data blocks content can be different for each individual ETB (see 5.3.4 for details).

Closed trains are represented as consists in the consist info data block, and closed train consist infos may be added optionally. Those entries are identifiable by the consist info class parameter (see 5.2.4).

Train Network Directory

Contains the description of the network topology of one ETB. This is the result of the ETB train inauguration as defined in IEC 61375-2-5. The train network directory is hallmarked by unique signature called etbTopoCnt. The TTDB may contain up to four train network directories representing the maximal four ETBs of the train network.

Train Directory

is (re-) computed each time there is a change in the train composition indicated by a change of the etbTopoCnt as defined in IEC 61375-2-5. The train directory provides an ordered list of all consists which have been discovered over the ETB the train directory is associated to.

Operational Train Directory

is (re-) computed each time there is a change in the train directory or upon a user request for leading (see 6.5) or for correction (see 6.6). This data block adds information about leading and correction to the TTDB and represents the operational train view as defined in 4.2.4.1. In trains with multiple ETBs only one operational train directory for all ETBs will be generated.

Each of these data blocks is protected by a signature, which allows checking the integrity of the stored data, see 5.3.3 for a detailed definition of these data blocks.

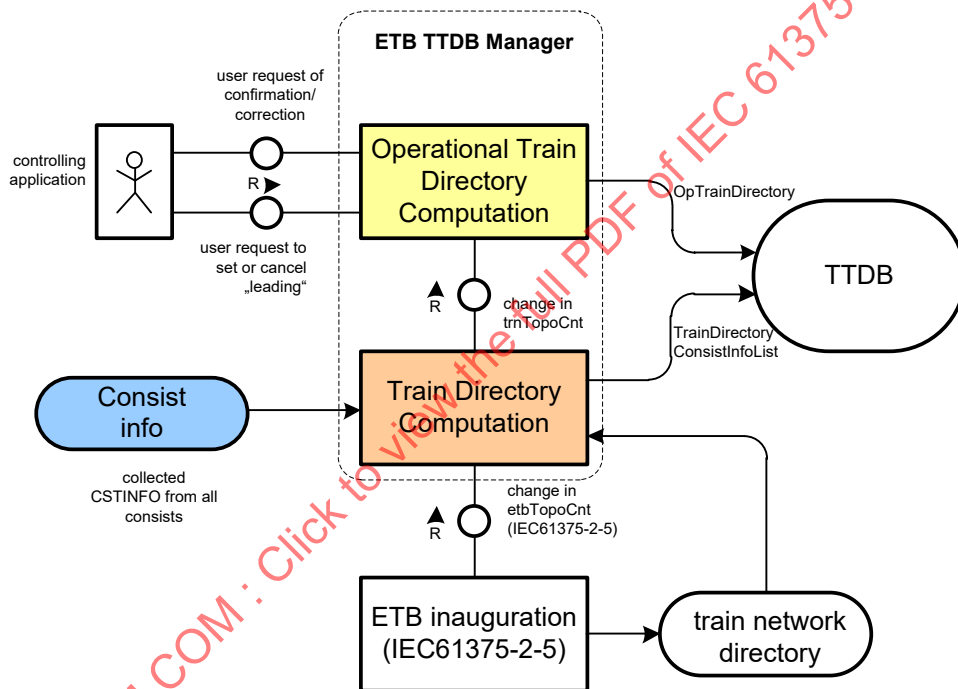
5.3.2 Computation of the TTDB

5.3.2.1 General

The TTDB is computed in several steps and by different functions as it is shown in Figure 14:

- ETB Inauguration, as defined in IEC 61375-2-5.
- Train directory computation on the base of the train network directory and the consist info.
- Operational train directory computation on the base of leading and correction information.

The different functions are defined in more detail in the following paragraphs.



IEC

Figure 14 – TTDB computation block diagram

5.3.2.2 ETB train inauguration

Basis of the TTDB is the train network directory, which is computed in accordance to the algorithm defined in IEC 61375-2-5, and which represents the topology of the ETB as a sequence of ETBN belonging to consists. A discovered ETB network topology will be uniquely identified by its topography counter (etbTopoCnt).

5.3.2.3 Computation of the train directory

A (re-)computation of the train directory shall be executed each time there is a change of the etbTopoCnt.

During train directory computation, the TTDB shall be enriched by consist information collected from the consists in the train (CSTINFO telegrams).

The process of train directory computation is defined by the state diagram shown in Figure 15 and further described in Table 3 to Table 5.

NOTE 1 A recomputation of the train directory will be prevented if train inaugurations are inhibited as defined in IEC 61375-2-5.

For the optimization of CSTINFO and CSTINFOCTRL message exchange (network load reduction) a set of flags controlling the sending of CSTINFO and CSTINFOCTRL messages shall be supported, which are defined as:

ciTxFlag	BOOLEAN	-- CSTINFO send control flag
		FALSE: CSTINFO not to send
		TRUE: CSTINFO to send
cicTxCnt1	UINT8	-- CSTINFOCTRL send control flag (counter)
		value range: 0..1
cicTxCnt2	UINT8	-- CSTINFOCTRL send control flag (counter)
		value range: 0..2

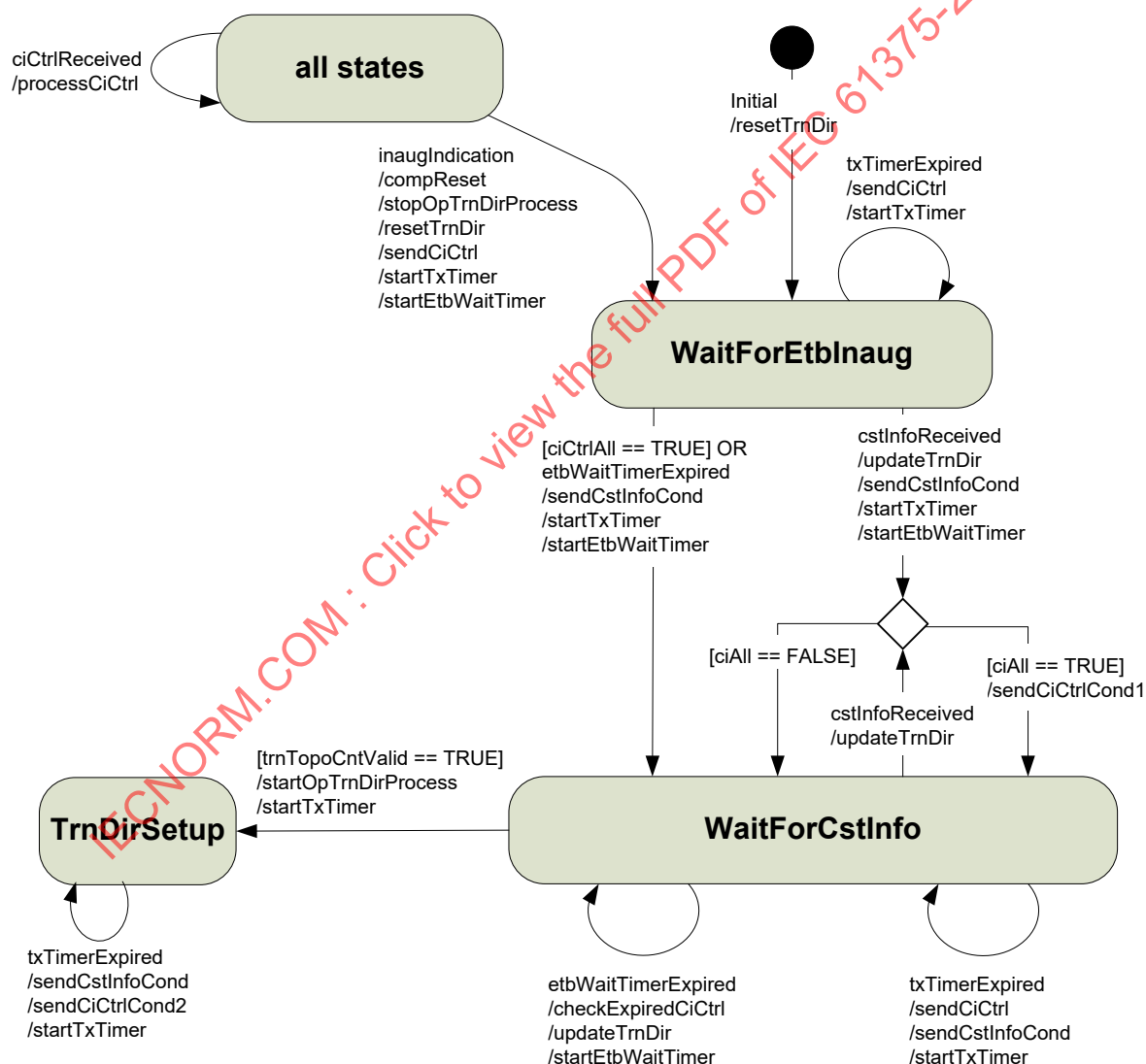


Figure 15 – Train directory computation state diagram

NOTE 2 For some transitions in the UML state machine on Figure 15 a trigger is the reception of a CSTINFOCTRL packet. In those cases only the guards are indicated in the UML state machine transition.

Table 3 – Train directory computation – triggers

Trigger	Description
Initial	Power-up or reboot of ECSP
inaugIndication	Indication of a new train inauguration provided by the local ETBN in accordance to IEC 61375-2-5.
ciCtrlReceived	CSTINFOCTRL telegram has been received.
cstInfoReceived	CSTINFO telegram has been received.
etbWaitTimerExpired	CSTINFOCTRL wait timer expiration – indicating that some consist listed in Train Network Directory is not sending CSTINFOCTRL packets, i.e. ECSP in this consist is not operating correctly.
txTimerExpired	Timer for periodic transmission of CSTINFO and CSTINFOCTRL telegram has expired. Expiration of this timer indicates that previously transmitted CSTINFO or CSTINFOCTRL has not been received by all ECSP.

Table 4 – Train directory computation – guards

Guard	Description
[trnTopoCntValid == TRUE]	CSTINFOCTRL has been received from all consists listed in the train directory with parameter 'cstTopoCnt' value != 0 and all consist from which CSTINFOCTRL has been received have identical trnTopoCnt value. NOTE 1 The trnTopoCnt value is known from the CSTINFOCTRL telegrams received with trnCstNo > 0. NOTE 2 This means that all active ECSP share the same input data for the calculation of Operational Train Directory.
[CiAll == TRUE]	CSTINFO message received from all consists listed in the train directory
[CiAll == FALSE]	CSTINFO message not (yet) received from all consists listed in the train directory
[CiCtrlAll == TRUE]	CSTINFOCTRL message received from all consists listed in the train directory
[CiCtrlAll == FALSE]	CSTINFOCTRL message not (yet) received from all consists listed in the train directory

Table 5 – Train directory computation – actions

Action	Description
startEtbWaitTimer	(Re-)Start one-shot timer with a time value of $(5,0 \pm 0,2)$ s. Within this time CSTINFOCTRL telegrams from all consists listed in Train Directory should be received.
startTxTimer	(Re-)Start one-shot timer with a time value $(1,0 \pm 0,1)$ s. This timer drives the periodic retransmission of CSTINFOCTRL and CSTINFO telegrams in case of some network failure. NOTE CSTINFO is retransmitted only if requested by some CSTINFOCTRL telegram.
resetTrnDir	Reset content of train directory: - train directory contains all consists identified by ETB inauguration (listed in Train Network Directory). - cstTopoCnt in all CONSIST structures are set to 0, except for own

Action	Description
	consist. compute the trnTopoCnt Clear the consist info list except for the entry of the own consist Reset send control flags: ciTxFlag = FALSE cicTxCnt1 = 1 cicTxCnt2 = 0
updateTrnDir	Store CSTINFO in ConsistInfoList. Update train directory: - Update CONSIST structures in train directory with cstTopoCnt from received CSTINFO - Calculate new value of trnTopoCnt.
startOpTrnDirProcess	Start the operational train directory processing by triggering the event 'startOpTrnDirProcess' in the ETBCTRL processing state machine, see 6.4.5
stopOpTrnDirProcess	Stop the operational train directory processing by triggering the event 'stopOpTrnDirProcess' in the ETBCTRL processing state machine, see 6.4.5
sendCstInfoCond	Conditional sending of CSTINFO telegram: IF (ciTxFlag == TRUE) THEN send CSTINFO telegram; ciTxFlag = FALSE; END; NOTE The conditional sending of CSTINFO aims to reduce the network load
sendCiCtrl	Send CSTINFOCTRL telegram
sendCiCtrlCond1	Conditional sending of CSTINFOCTRL telegram: IF (cicTxCnt1 > 0) THEN send CSTINFOCTRL telegram; cicTxCnt1 = cicTxCnt1 – 1; END; NOTE The conditional sending of CSTINFOCTRL aims to reduce the network load
sendCiCtrlCond2	Conditional sending of CSTINFOCTRL telegram: IF (cicTxCnt2 > 0) THEN send CSTINFOCTRL telegram; cicTxCnt2 = cicTxCnt2 – 1; END; NOTE The conditional sending of CSTINFOCTRL aims to reduce the network load
processCiCtrl	Process received CSTINFOCTRL telegram. IF (CSTINFOCTRL parameter 'trnCstNo' > 0) THEN memorize the CSTINFOCTRL parameter 'trnTopoCnt'; memorize the time when CSTINFOCTRL has been received; END; IF ((CSTINFOCTRL parameter 'trnCstNo' > 0) AND (state == TrainDirSetup) AND (CSTINFOCTRL parameter 'trnTopoCnt' != trnTopoCnt value in local train directory)) THEN cicTxCnt2 = 2;

Action	Description
	END; IF (CSTINFOCTRL message does contain an entry of parameter 'cstTopoCnt' with a value $\neq 0$ for the local consist, which is the consist the receiving ECSP belongs to) THEN ciTxFlag = TRUE; END; NOTE This means that the sending ECSP has not received the CSTINFO of the local consist and is requesting now to (re)send this CSTINFO message.
checkExpiredCiCtr	Check for expired CSTINFOCTRL messages. For each consist from which the last received CSTINFOCTRL message is older than the value of etbWaitTimer: • Discard the CSTINFOCTRL message together with the memorized trnTopoCnt value. • Remove the consist info from the consist info list. • Set consist parameter 'cstTopoCnt' in train directory to 0.
compReset	Reset the composition by triggering the event 'compReset' in the leading function state machine (see 6.5.3.2) and the confirmation/correction state machine (see 6.6.4).

NOTE The process of train directory computation is triggered by the ETBN after train inauguration. Subsequently, the ECSPs exchange CSTINFOCTRL and CSTINFO telegrams for the computation of the train directory. ETBN and ECSP need to be synchronized in a way that the ECSP can start the train directory computation process immediately after the train inauguration indication. If the time difference is too large (e.g. greater than EtbWaitTime) this consist will not be listed in the train directory with a cstTopoCnt $\neq 0$ and will subsequently not be listed in the operational train directory.

5.3.2.4 Computation of the operational train directory

A computation of the operational train directory shall be executed on user request and each time there is a change of the train directory associated to the operational network (ETB0).

If the user requests to set leading, the leading vehicle shall be determined and indicated in the database. This is further specified in 6.5.

If the user requests to cancel leading, the leading vehicle indication shall be removed from the database. This is further specified in 6.5.

As long as there is no user request for confirmation or correction, the TTDB shall be in state "unconfirmed".

If the user requests to correct the TTDB, the operational train directory shall be updated and the TTDB shall change to state "confirmed" if there is no conflict or error, otherwise it shall remain in state "unconfirmed". This is further specified in 6.6.

If the TTDB is in state "confirmed" and the train directory changes, it shall be checked whether the new train directory is in disagreement with the confirmed and/or corrected train composition. If there is a disagreement, the TTDB state shall be changed to "unconfirmed", this is further specified in 6.6.

NOTE This will for instance occur after the coupling of two trains.

Further specification on the computation of the operational train directory is given in 6.7.

5.3.3.1 Data model

The diagram illustrates the relationships between various entities in the Train Management Information System (TMIS). The entities are represented as classes, and their relationships are shown with directed associations and multiplicity values.

Entities and their relationships:

- OpTrainDirectoryState** (yellow) is associated with **OpTrainDirectory** (yellow) with a multiplicity of 1. It is also associated with **OpVehicle** (yellow) with a multiplicity of 1.63.
- OpTrainDirectory** (yellow) is associated with **OpConsist** (yellow) with a multiplicity of 1. It is also associated with **OpVehicle** (yellow) with a multiplicity of 1.63.
- OpConsist** (yellow) is associated with **ConsistInfoList** (blue) with a multiplicity of 1.63. It is also associated with **ConsistInfo** (blue) with a multiplicity of 1.63.
- ConsistInfoList** (blue) is associated with **ConsistInfo** (blue) with a multiplicity of 1.63.
- ConsistInfo** (blue) is associated with **ConsistProperties** (blue) with a multiplicity of 1. It is also associated with **FunctionInfo** (blue) with a multiplicity of 1.63.
- ConsistProperties** (blue) is associated with **VehicleInfo** (blue) with a multiplicity of 1. It is also associated with **VehicleProperties** (blue) with a multiplicity of 0.1.
- VehicleInfo** (blue) is associated with **FunctionInfo** (blue) with a multiplicity of 1. It is also associated with **VehicleProperties** (blue) with a multiplicity of 0.1.
- FunctionInfo** (blue) is associated with **FunctionProperties** (blue) with a multiplicity of 1. It is also associated with **VehicleInfo** (blue) with a multiplicity of 1.63.
- FunctionInfo** (blue) is associated with **ETB** (blue) with a multiplicity of 1.4. It is also associated with **CN** (blue) with a multiplicity of 1.32.
- ETB** (blue) is associated with **CN** (blue) with a multiplicity of 1.32.
- TrainDirectory** (orange) is associated with **Consist** (orange) with a multiplicity of 1.63.
- Consist** (orange) is associated with **ConsistInfoList** (blue) with a multiplicity of 1.63.
- ConsistInfoList** (blue) is associated with **ConsistInfo** (blue) with a multiplicity of 1.63.
- ConsistInfo** (blue) is associated with **ConsistProperties** (blue) with a multiplicity of 1. It is also associated with **FunctionInfo** (blue) with a multiplicity of 1.63.
- ConsistProperties** (blue) is associated with **VehicleInfo** (blue) with a multiplicity of 1. It is also associated with **VehicleProperties** (blue) with a multiplicity of 0.1.
- VehicleInfo** (blue) is associated with **FunctionInfo** (blue) with a multiplicity of 1. It is also associated with **VehicleProperties** (blue) with a multiplicity of 0.1.
- FunctionInfo** (blue) is associated with **FunctionProperties** (blue) with a multiplicity of 1. It is also associated with **VehicleInfo** (blue) with a multiplicity of 1.63.
- FunctionInfo** (blue) is associated with **ETB** (blue) with a multiplicity of 1.4. It is also associated with **CN** (blue) with a multiplicity of 1.32.
- ETB** (blue) is associated with **CN** (blue) with a multiplicity of 1.32.
- TrainNetworkDirectory** (green) is associated with **TrainNetworkDirectoryEntry** (green) with a multiplicity of 1. It is also associated with **Consist** (orange) with a multiplicity of 1.63.
- TrainNetworkDirectoryEntry** (green) is associated with **TrainNetworkDirectory** (green) with a multiplicity of 1. It is also associated with **Consist** (orange) with a multiplicity of 1.63.

References:

- OpTrainDirectoryState** is referenced by **opTrTopoCnt**.
- OpVehicle** is referenced by **opCatNo**.
- OpConsist** is referenced by **CatUID**.
- ConsistInfoList** is referenced by **CatUID**.
- ConsistInfo** is referenced by **CatUID**.
- ConsistInfoList** is referenced by **etbNo**.
- ETB** is referenced by **etbNo**.
- CN** is referenced by **cnNo**.
- FunctionInfo** is referenced by **cnNo**.
- VehicleInfo** is referenced by **tonVehNo**.
- VehicleInfo** is referenced by **tonVehNo**.

Notes:

- Maximal number of vehicles in train: 63

Figure 16 – TTDB class diagram (example)

NOTE An application interface for retrieving data from the TTDB is defined in Annex E.

5.3.3.2.1 Structure ETB_INFO – ETB information

```

ETB_INFO ::= RECORD
{
    etbId          UINT8          -- identification of the ETB
                                   0 = ETB0 (operational network)
                                   1 = ETB1 (multimedia network)
                                   2 = ETB2 (other network)
                                   3 = ETB3 (other network)

    cnCnt          UINT8          -- number of CN within
                                   consist connected to this ETB,
                                   value range 1..16,
                                   referring to cnId 0...15 acc. IEC 61375-2-5

    reserved01     UINT16         -- reserved for future use (= 0)
}

```

5.3.3.2.2 Structure CLTR_CST_INFO – info on closed train composition

The data structure CLTR_CST_INFO is defined as:


```

1 = ETB1 (multimedia network)
2 = ETB2 (other network)
3 = ETB3 (other network)
255 = related to all ETB (all-train group)
cnId      UINT8      -- identification of connected consist network
                        in the consist, related to the etbId
                        value range: 0..32
                        0 = not relevant (e.g. for groups)
reserved02  UINT8      -- reserved for future use (= 0)
}

```

The combination of parameters 'fctName', 'fctId', 'cnld' and 'etbld' shall be unique.

This part of IEC 61375 defines only that part of the function information, which is needed for function identification and function addressing. More detailed information about the function itself, and the way to retrieve this information over the network, shall be defined within the related application profile.

NOTE The application profile can define a TRDP message and related service primitives to retrieve function information. Good practise is to define a standard way which is common for all services.

5.3.3.2.5 Structure VEHICLE_INFO – static vehicle attributes

The data structure VEHICLE_INFO is defined as:

```

VEHICLE_INFO ::= RECORD
{
    vehId      LABEL      -- vehicle identifier label, application defined
                        (e.g. UIC vehicle identification number)
                        vehId of vehNo = 1 can be used also as
                        consist identifier
    vehType    LABEL      -- vehicle type, application defined
    vehOrient  UINT8      -- vehicle orientation
                        '01'B = same as consist direction
                        '10'B = inverse to consist direction
    cstVehNo   UINT8      -- sequence number of vehicle within the
                        consist(1..32)
    tractVeh   ANTIVALENT8 -- vehicle is a traction vehicle
                        '01'B = vehicle is not a traction vehicle
                        '10'B = vehicle is a traction vehicle
    reserved01  UINT8      -- for future use (= 0)
    vehProp    PROPERTIES -- static vehicle properties
}

```

5.3.3.2.6 Structure CONSIST_INFO– static consist attributes

This dataset is filled with the information, which has been received with the corresponding CSTINFO telegram.

The data structure CONSIST_INFO is defined as:

```

CONSIST_INFO ::= RECORD
{
    version    VERSION      -- CONSIST_INFO data structure version,
                        mainVersion = 1
                        subVersion = 0
    cstClass   UINT8      -- consist info classification
                        1 = (single) consist
}

```



```

2 = closed train
3 = closed train consist

reserved01  UINT8      -- reserved for future use (= 0)
cstId       LABEL      -- consist identifier, application defined
                        (e.g. UIC identifier)
cstType     LABEL      -- consist type, application defined
cstOwner    LABEL      -- consist owner, application defined
                        (e.g. "trenitalia.it")
cstUUID     UINT8[16]   -- UUID of the consist
reserved02  UINT32      -- reserved for future use (= 0)
cstProp     PROPERTIES  -- static consist properties
reserved03  UINT16      -- reserved for future use (= 0)
etbCnt      UINT16      -- number of ETB's, range: 1..4
etbInfoList ARRAY [] OF ETB_INFO
                        -- ETB information for ETBs in the consist
                        Ordered list starting with lowest etbId
reserved04  UINT16      -- reserved for future use (= 0)
vehCnt      UINT16      -- number of vehicles in consist
                        value range: 1..32
vehInfoList ARRAY [] OF VEHICLE_INFO
                        -- vehicle info for the vehicles in the consist
                        Ordered list starting with cstVehNo = 1
reserved05  UINT16      -- reserved for future use (= 0)
fctCnt      UINT16      -- number of consist functions
                        value range: 0..1024
fctInfoList ARRAY [] OF FUNCTION_INFO
                        -- function info for the functions in consist
                        lexicographical ordered by fctName
reserved06  UINT16      -- reserved for future use (= 0)
cltrCstCnt  UINT16      -- number of closed train consists
                        value range: 0..32
                        0 = consist is no closed train
cltrCstInfoList ARRAY [] OF CLTR_CST_INFO
                        -- info on closed train composition
                        Ordered list starting with cltrCstNo = 1
cstTopoCnt  UINT32      -- consist topography counter
                        from the beginning of the record until the
                        last element before the topography counter
                        computed as defined in 5.3.3.2.16
                        (seed value: 'FFFFFFFF'H)
}

```

5.3.3.2.7 Structure CONSIST_INFO_LIST – list of consist info

The consist info list provides a collection of CONSIST_INFO elements as defined in 5.3.3.2.7 which are filled with the content of received CSTINFO telegrams. It cannot be guaranteed that a consist info is available for all consists within a train, especially if there are consists inserted by correction or temporarily after a train inauguration. The consist info is identified by the cstUUID value of the consist.

Within the consist info list all data received from CSTINFO classes 1 and 2 telegrams shall be stored. Data received from CSTINFO class 3 telegrams need not be stored.

The data structure CONSIST_INFO_LIST is defined as:

```

CONSIST_INFO_LIST ::= RECORD
{

```

5.3.3.2.8 Structure CONSIST – dynamic consist attributes

Closed train consists are not referenced within this data structure. However, the CSTINFO class 3 datasets representing the close train consists are stored in the CONSIST_INFO_LIST as well.

CONSIST ::= RECORD

5.3.3.2.9 Structure TRAIN_DIRECTORY

The data structure TRAIN DIRECTORY is defined as:

TRAIN DIRECTORY ::= RECORD

[illegible]

```

                2 = ETB2 (other network)
                3 = ETB3 (other network)
cstCnt          UINT8          -- number of consists in train
                                range: 1..63
cstList ARRAY [] OF CONSIST
                                -- dynamic consist list
                                ordered list starting with trnCstNo = 1
trnTopoCnt      UINT32         -- Train topography counter
                                computed as defined in 5.3.3.2.16
                                from the beginning of the record until the
                                last element before the topography counter
                                (seed value: etbTopoCnt)
}

```

5.3.3.2.10 Structure OP_VEHICLE – operational vehicle attributes

The data structure OP_VEHICLE is defined as:

```

OP_VEHICLE ::= RECORD
{
    vehId          LABEL          -- unique vehicle identifier, application
                                defined (e.g. UIC identifier)
    opVehNo        UINT8          -- operational vehicle sequence number in train
                                value range: 1..63
    isLead         ANTIVALENT8    -- vehicle is leading
    leadDir        UINT8          -- vehicle leading direction
                                0 = not relevant
                                1 = leading direction 1
                                2 = leading direction 2
    trnVehNo       UINT8          -- vehicle sequence number within the train
                                with vehicle 01 being the first vehicle
                                in ETB reference direction 1 as defined in
                                IEC 61375-2-5
                                value range: 1..63
                                a value of 0 indicates that this vehicle
                                has been inserted by correction
    vehOrient      UINT8          -- vehicle orientation
                                '0'B = not known (corrected vehicle)
                                '01'B = same as operational train direction
                                '10'B = inverse to operational train
                                direction
    ownOpCstNo     UINT8          -- operational consist number the vehicle
                                belongs to
    reserved01     UINT8          -- reserved for future use (= 0)
    reserved02     UINT8          -- reserved for future use (= 0)
}

```

5.3.3.2.11 Structure OP_CONSIST – operational consist attributes

The data structure OP_CONSIST is defined as:

```

OP_CONSIST ::= RECORD
{
    cstUUID        UINT8[16]     -- Reference to static consist attributes
                                0 = not available (e.g. correction)
    opCstNo        UINT8          -- operational consist number in train (1..63)
    opCstOrient    UINT8          -- consist orientation
}

```

```

\0'B = not known (corrected vehicle)
\01'B = same as operational train direction
\10'B = inverse to operational train
direction
trnCstNo      UINT8      -- Sequence number of consist
                        in train with consist 01 being the first
                        consist in ETB reference direction 1
                        as defined in IEC 61375-2-5
                        value range 0..63
                        a value of 0 indicates that this consist
                        has been inserted by correction
reserved01    UINT8      -- reserved for future use (= 0)
}

```

5.3.3.2.12 Structure OP_TRAIN_DIRECTORY_STATE

The data structure OP_TRAIN_DIRECTORY_STATE is defined as:

```

OP_TRAIN_DIRECTORY_STATE ::= RECORD
{
    version          VERSION  -- TRAIN_DIRECTORY_STATE data structure version
                                mainVersion = 1
                                subVersion = 0
    reserved01       UINT8    -- reserved for future use (= 0)
    reserved02       UINT8    -- reserved for future use (= 0)
    etbId            UINT8    -- identification of the ETB the TTDB is
                                computed for
                                0 = ETB0 (operational network)
                                1 = ETB1 (multimedia network)
                                2 = ETB2 (other network)
                                3 = ETB3 (other network)
    trnDirState      UINT8    -- TTDB status
                                1 = UNCONFIRMED
                                2 = CONFIRMED
    opTrnDirState    UINT8    -- operational train directory status
                                1 = INVALID
                                2 = VALID
                                4 = SHARED
    reserved03       UINT8    -- reserved for future use (= 0)
    trnId            LABEL    -- train identifier, application defined
                                (e.g. "ICE75", "IC346"), informal
    trnOperator      LABEL    -- train operator, e.g. "trenitalia.it",
                                informal
    opTrnTopoCnt     UINT32    -- operational train topography counter
                                set to 0 if opTrnDirState = INVALID
    crc              UINT32    -- SC-32 computed over record
                                (seed value: 'FFFFFFFF'H)
}

```

5.3.3.2.13 Structure OP_TRAIN_DIRECTORY

The operational train directory represents the operational train view as defined in 4.2.4.1.

The data structure OP_TRAIN_DIRECTORY is defined as:

```

OP_TRAIN_DIRECTORY ::= RECORD
{

```


bit 24..29 = CN Id
bit 30..31 = 0

}

5.3.3.2.15 Structure TRAIN_NETWORK_DIRECTORY (informal)

The train network directory represents the network topology of one ETB as defined in IEC 61375-2-5.

The data structure TRAIN_NETWORK_DIRECTORY is defined as:

```
TRAIN_NETWORK_DIRECTORY ::= RECORD
{
    reserved01    UINT16          -- reserved for future use (=0)
    entryCnt      UINT16          -- number of entries in train net directory
    trnNetDir     ARRAY [entryCnt] OF TRAIN_NETWORK_DIRECTORY_ENTRY
    etbTopoCnt    UINT32          -- train network directory CRC
}
```

5.3.3.2.16 Computation of topography counter values

The TTDB defined three topography counters, which are used for different purposes:

cstTopoCnt:	unique signature of the consist info
trnTopoCnt:	unique signature of the train directory
opTrnTopoCnt:	unique signature of the operational train directory

For the computation of those topography counters, the following general rules shall apply:

- First, a SC-32 checksum (see Clause B.7) over the specified data range and with the specified seed value shall be calculated.
- If the calculation results in a value different to 0, the topography counter shall be set to the value of the SC-32 checksum.
- If the calculation results in a value of 0, the topography counter shall be set to a value of $(2^{32} - 1)$.
- Setting the topography counter to a value of 0 declares the topography counter to be invalid.
- The topography counter value shall be stored in big-endian format.

NOTE 1 Using a value of 0 to declare an invalid topography counter avoids the definition of an additional validity flag.

NOTE 2 The probability P_{crc} to not detect a modification of a data packet protected by a SC-32 CRC is about $P_{crc} \approx 2^{-32}$. The probability P_{tc} to not detect a modification of a data packet signed by a topography counter as specified in this part of IEC 61375 is about $P_{tc} \approx (2^{-32} + 2^{-63})$, a difference which is negligible.

5.3.4 Train Topology Database for multiple ETBs (Option)

5.3.4.1 General

The TTDB is built on the base of the train network directory, which represents the train backbone view of one particular ETB. This train backbone view can be different for the different ETBs within the train. If a TTDB were computed for each ETB, the consequence would be that a different train directory and operational train directory would be computed for each ETB, potentially leading to different train views dependent on which ETB is used for reference.

In order to avoid ambiguousness in the train views of trains with multiple ETBs (see 4.2.2.6.1), the train directory and the operational train directory shall be computed by ETB0 (operational ETB) only and shall be adopted by the other ETBs.

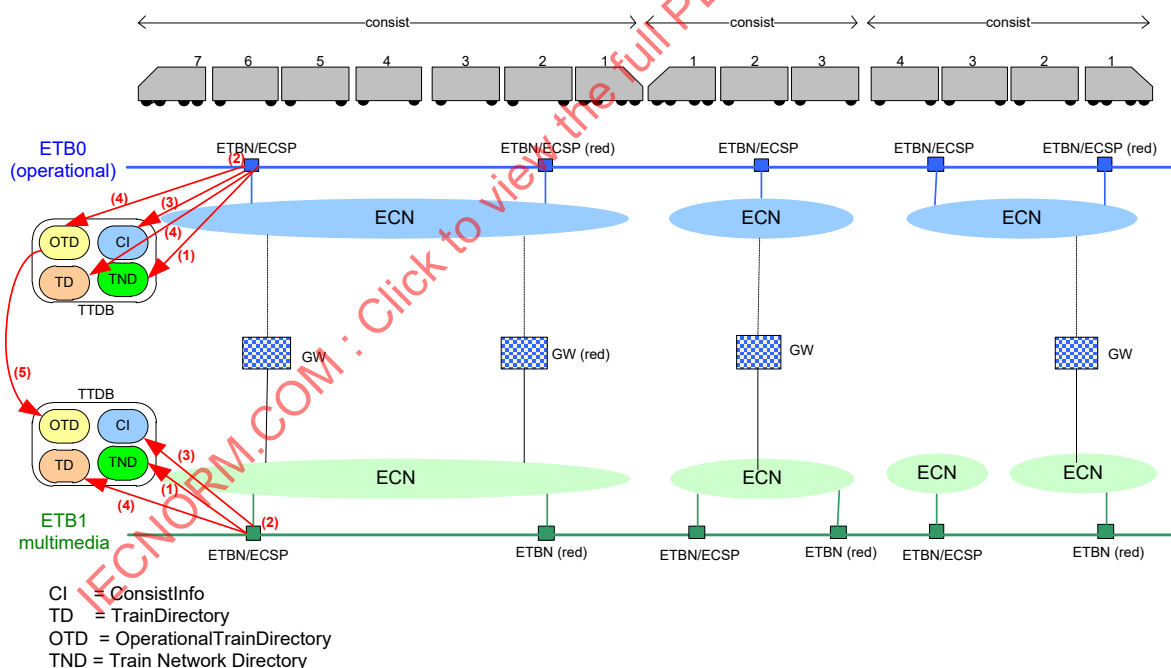
5.3.4.2 Adoption of the TTDB within dependent ETBs

Adopting the train directory and the operational train directory from ETB0 can be defined as a 5 step process (for illustration see Figure 17):

- Each ETB executes the train inauguration and generates the train network directory as defined in IEC 61375-2-5.
- Based on the inauguration result, IP addresses of ETBNs and the subnet addresses of the connected CNs are assigned as defined in IEC 61375-2-5.
- CSTINFO telegrams are exchanged on each ETB as defined in 5.2.3 and stored within the consist info list data block. The train directory is established and received CSTINFO telegrams are registered.
- ETB0 computes the operational train directory.
- The other ETBs copy or reference the operational train directory computed by ETB0.

NOTE 1 Instead of copying the operational train directory, it is also possible to provide access from other ETBs to the operational train directory of the ETB0 ('referencing').

All ETBs shall continuously monitor the opTrnTopoCnt of the ETB0, and in case of a change, shall update their TTDB.



IEC

Figure 17 – TTDB adoption (in this example shown for the first consist)

NOTE 2 A consequence will be that a consist which is not reachable on ETB0, but available on multimedia network, is not listed within the TTDB of the multimedia network.

5.3.4.3 Restrictions on network architecture

In order to ensure that CSTINFO telegrams are sent by all consists on all ETBs, each consist in the train should be connected to all the ETBs.

5.3.4.4 Restrictions on CSTINFO telegrams

The CSTINFO telegrams sent by one consist on the different ETBs shall indicate the same number, the same sequence, the same orientation, and the same identity of contained vehicles.

Functions which shall be accessible from multiple ETBs shall be listed in all the ETB related CSTINFO telegrams.

NOTE A typical use case is that a function of the operational network sends data to a remote (other consist) function on the multimedia network. Then it finds all information of the remote function in its local TTDB.

All other data within the CSTINFO telegram may be different as they are not relevant for interoperability.

5.4 Service Addressing

5.4.1 General

Service addressing is a substantial part of a communication system as it identifies the endpoints of a communication relationship and as such the communication partners. Communication, and related to that the addressing, can be defined between physical entities, like physical devices or human beings, or can be defined between more abstract entities like application or system functions.

A schema for the train wide addressing of physical entities is defined in IEC 61375-2-5.

This part of the standard defines an addressing scheme, which shall be used for the train wide addressing of onboard train functions.

On functional level, a more abstract addressing scheme based on TCN domain names (see 5.4.3) is defined which is equivalent to the domain name system defined for the Internet. The main reasons for introducing an additional addressing scheme on top of IP addresses are:

- Usable to address abstract functions instead of device related IP addresses. For example, the same function might be accessible via different IP addresses in different consists, but has the same name in all consists.
- IP addresses can be changed without impacting the domain name.
- Names can be better understood and interpreted by humans. This might become more important in future when IPv6 addresses are in use.

Functional addressing is related to logical functions.

Sources of information can be:

- a function within a vehicle or a consist

Destinations of information can be:

- a function within a vehicle or a consist
- a function within multiple or all consists

NOTE A function can be implemented by one or by several end devices in a consist.

5.4.2 TCN Domain Name System (TCN-DNS)

5.4.2.1 General

The TCN Domain Name System defines a hierarchical, domain based naming scheme and a distributed database system for implementing this naming scheme. Its primary use is to map host names to IP addresses.

NOTE The concept of TCN-DNS is strongly related to the Internet DNS as defined in RFC 1034 and RFC 1035.

5.4.2.2 TCN Domain Name Space and Zones

The TCN Domain name space is defined by a tree structure with the local train ('lTrn') being the (local) root. The next level underneath the root is defined by the closed train sub-domains (cltr01 ... cltrNN). The closed train sub-domains are further structured into the consist sub-domains (cst01 ... cstNN), the vehicle sub-domains (veh01 ... vehNN) finally ending in the hosts, which are the devices hosting the logical functions (fctDevXX ... fctDevYY).

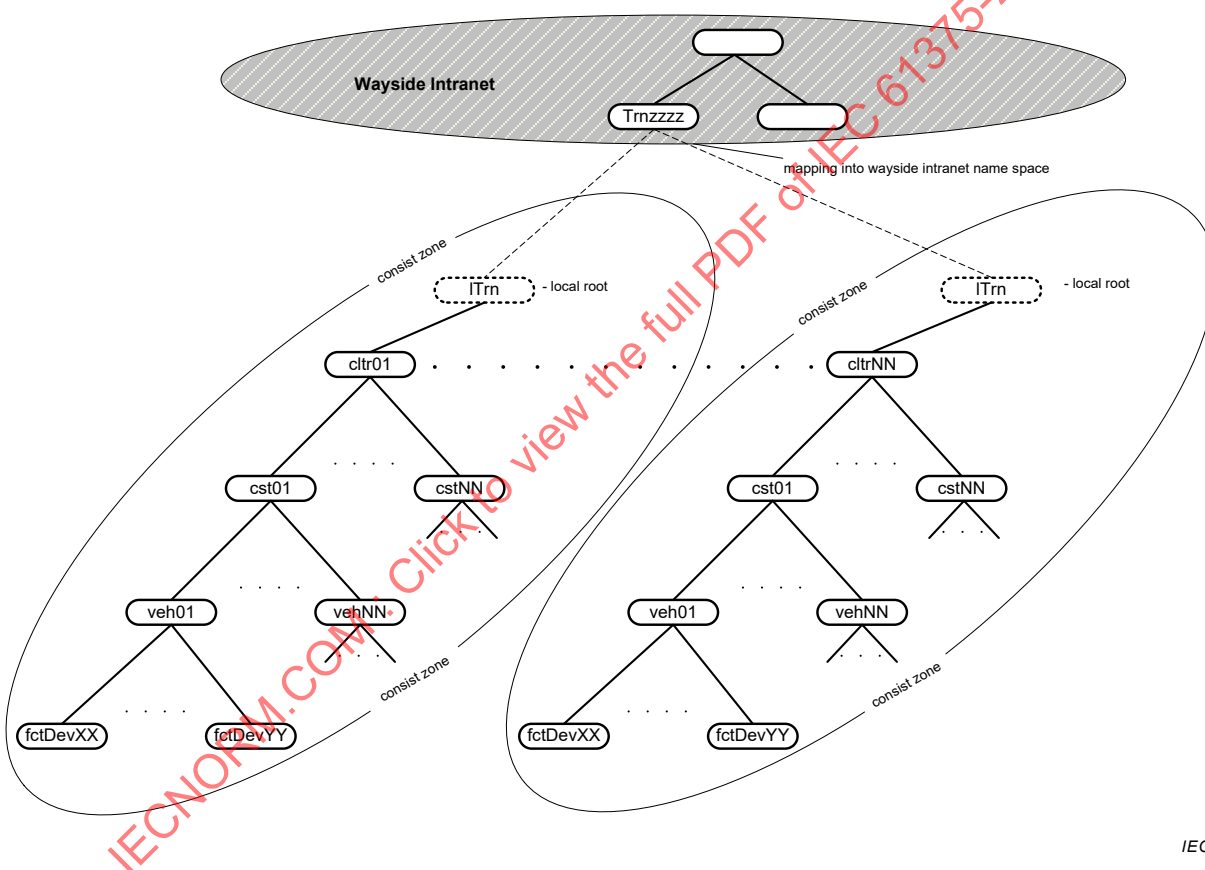


Figure 18 – TCN-DNS name space with division into zones

For onboard-to-ground communication, the whole TCN-DNS name space can be mapped into superior intranets owned for example by the operator or maintainer of the train (this however is not covered by this part of the standard). In this case, the mapped TCN-DNS name space may be identified by an operator's specific root name (e.g. 'Trnzzzz').

The TCN-DNS name space is divided into zones, where each zone is administrated by one authoritative DNS name server.

One possibility is to define consist sub-domains as one zone as it is shown in Figure 18, and one authoritative DNS name server is responsible for this zone. In consists with several networks it might be advantageous to subdivide the consist sub-domain in several zones, and to let the consist DNS name server delegate these zones to (subordinated) DNS name servers.

5.4.3 TCN Domain Names

Communicating entities (sources and destinations) shall be identified by a fully qualified TCN domain name which is derived from the TCN domain name space.

5.4.4 TCN-URI Scheme

5.4.4.1 General

The TCN domain name space syntax is defined by an URI scheme following the recommendations given in RFC 3986.

5.4.4.2 TCN URI schema syntax

The syntax of the TCN URI is defined in accordance to RFC 3986 using the Backus Naur Form (BNF) notation. The specific definition of the TCN URI is subject of the following subclauses.

The TCN URI uses some basic elements for syntax definition, which are defined as listed in Table 6.

Table 6 – TCN URI basic syntax

```
label = alpha 1{uchar}13 alphanum
uchar = alphanum | mark
alphanum = alpha | digit
alpha = "a" | "b" | "c" | "d" | "e" | "f" | "g" | "h" | "i" |
        "j" | "k" | "l" | "m" | "n" | "o" | "p" | "q" | "r" |
        "s" | "t" | "u" | "v" | "w" | "x" | "y" | "z" |
        "A" | "B" | "C" | "D" | "E" | "F" | "G" | "H" | "I" |
        "J" | "K" | "L" | "M" | "N" | "O" | "P" | "Q" | "R" |
        "S" | "T" | "U" | "V" | "W" | "X" | "Y" | "Z"
digit = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
mark = "-"
```

NOTE This label definition deviates from RFC 1035 with respect to:

- Restriction to maximally 15 characters (RFC 1035: 63 characters allowed)
- First character cannot be a digit (RFC 1035 allows a digit).

5.4.4.3 TCN URI general schema

The TCN URI Identifier shall be used to identify the source function/device and the destination function/device of information. The general schema of the TCN URI is defined in Table 7.

Table 7 – General schema syntax

```
TCN-URI = [scheme ":" "//"] user "@" host
scheme = "trn"
user = usr
host = fctdev "." vehicle "." consist "." [cltrain "."]train
["."]
```

According to this definition, the TCN-URI Identifier can be written as (Figure 19):



Figure 19 – TCN-URI Schema

NOTE 1 The host part of the TCN-URI is resolvable to an IP address by the TCN-DNS service.

NOTE 2 The label 'fctdev' identifies the logical device which is implementing the addressed service function. "Logical" device means that it is not known how this device is implemented, e.g. it could be a single physical device implementing the function or the function could be distributed over several physical devices.

For addressing a physical device, the physical addressing scheme defined in IEC 61375-2-5 shall be used.

5.4.4.4 TCN URI Scope

Within a communication relationship, both the source and the destination of information are identified by TCN-URI. Whether a specific TCN URI is related to a source or to a destination is called its scope. TCN-URI syntax rules may be different if the TCN-URI is used for a source or for a destination.

5.4.4.5 TCN URI User part (optional)

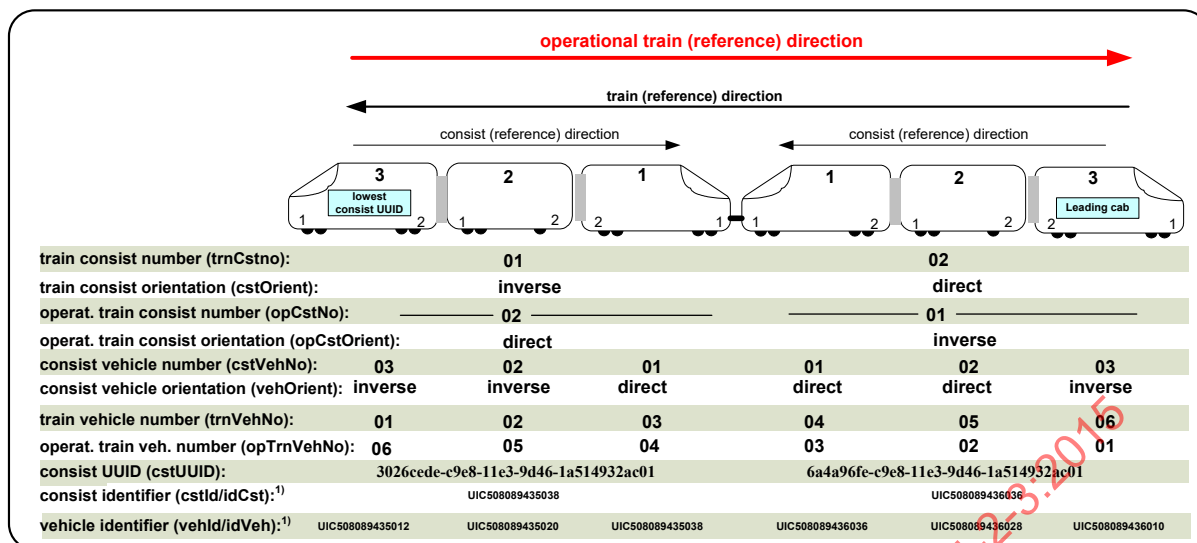
The user part can be used for the application profiles defined in IEC 61375-2-4 to distinguish different functions and function instances, which are located on the same physical device.

This part of IEC 61375 defines a character string of 32 octets which is reserved for the user part.

5.4.4.6 TCN URI Host part

5.4.4.6.1 Directions, orientations and numbers in a train (informative)

Navigation in a train network requires a precise definition of the directions and numbering schemes in a train. Directions and numberings are defined in different parts of IEC 61375.



¹⁾ Identifiers as defined by UIC are used in these examples, e.g. 'UIC508089435038' corresponds to UIC identifier '50 80 89 – 43 503 - 8'

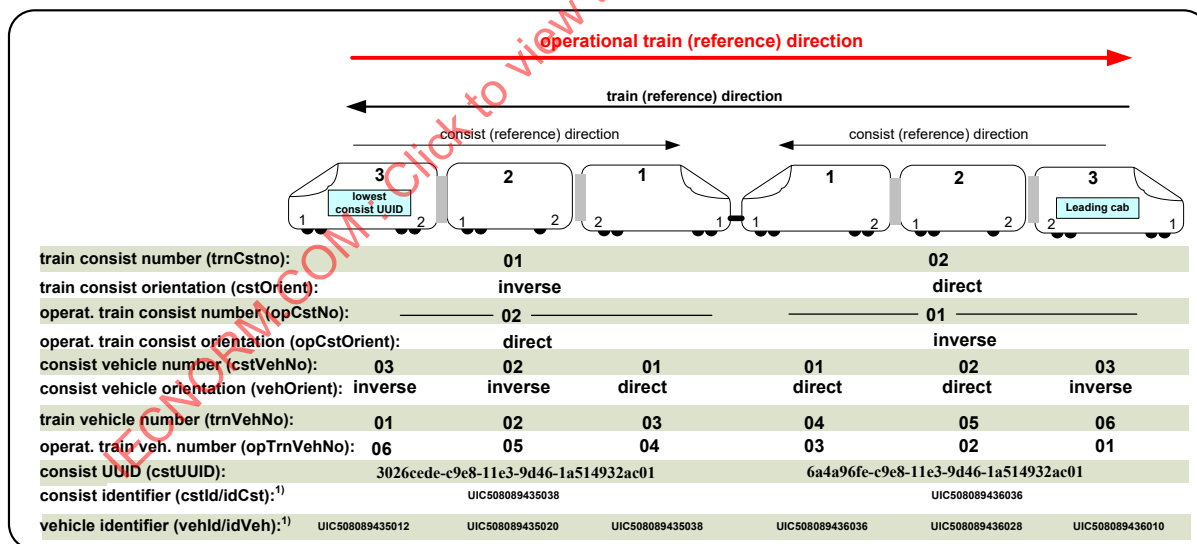
IEC

Figure 20 intends to show a summary of those directions and numberings.

NOTE 1 The consist reference direction is based on the construction of the consist. Vehicle 1/cab 1 is always end 1 (front) of the consist. The consist reference direction is used for the vehicle numbering within the consist.

NOTE 2 The TCN reference direction is based on the UUID of the consists as defined in IEC 61375-2-5. The consist with the lowest UUID is always the TCN consist 1.

NOTE 3 The operational direction is based on the leading vehicle/leading consist. The end 1 (front end) of the train is defined depending on the the viewing direction of the driving cab of the leading vehicle (see 4.2.4.3). The operational direction can be used for the management of operational direction depending signals.



¹⁾ Identifiers as defined by UIC are used in these examples, e.g. 'UIC508089435038' corresponds to UIC identifier '50 80 89 – 43 503 - 8'

IEC

Figure 20 – Directions, orientations and numbers in train

5.4.4.6.2 Function/Device Label

Function devices shall be identified by labels following the syntax given in Table 8:

Table 8 – Device label syntax

```

fctDev = "lDev" | deviceName | group
devicename = label
group = "grpAll" | grpDev
grpDev = "grp" 1{uchar}12

```

The meaning of the different device labels is defined in Table 9.

Table 9 – Device label definition

Label <Device>	Scope S=source D=destination	Manda- tory(M)/ Optiona(O)	Definition
"lDev"	D	O	The destination function is located on the local ED. To be used for addressing functions residing on the own device.
deviceName	S,D	M	Defines the function device (logical device) . EXAMPLE 'fctDoorCtrl', 'fctBrake', 'devHMI'. This will be resolved to an IP unicast address.
grpDev	D	M	Defines a group of function devices (logical device). EXAMPLE 'grpDoorCtrl', 'grpBrake', 'grpETBN', 'grpECSC'. This will be resolved to an IP multicast address.
"grpAll"	D	M	All function devices (logical device) or the physical device. This will be resolved to an IP multicast address.
anyDev	S	O	The location of the source function is unknown. This can be used for TRDP subscription based on ComId and source consist or source closed train. Shall be used in conjunction with 'anyVeh' and will resolve to an IP address range covering all CNs in a consist or a closed train. EXAMPLE 1 TCN-URI 'anyDev.anyVeh.cst05.anyCITrn.ITrn' can be used to subscribe to a specific ComId in consist 05 EXAMPLE 2 TCN-URI 'anyDev.anyVeh.leadCst.anyCITrn.ITrn' can be used to subscribe to a specific ComId in the leading consist

NOTE It is also possible to define alias names for devices, meaning that the same physical device appears with different device labels.

It is recommended to use defined prefixes for device names, e.g. 'fct' for a function device (logical device) and 'dev' for a physical device.

To support abstract functional addressing device labels or device group labels can be standardized. For instance, the function "brake" can be defined to be addressed via a device label "fctBrake", which is standardized for all TCN applications. It is however not in the scope of this standard to define this.

5.4.4.6.3 Vehicle Label

Vehicles shall be identified by labels following the syntax given in Table 10:

Table 10 – vehicle label syntax

```
veh = "lVeh" | cstVehNo | opVehNo | "leadVeh" | "aVeh" |
      idVeh | "anyVeh"
cstVehNo = "veh" digit digit
opVehNo = "opVeh" digit digit
idVeh = label
```

The meaning of the different vehicle labels is defined in Table 11.

Table 11 – Veh (vehicle) label definition

Label <veh>	Scope S=source D=destination	Manda- tory(M)/ Optiona(O)	Definition
lVeh	D	O	Function in the own vehicle.
aVeh	D	M	Functions located in some or all vehicles. The vehicle label "aVeh" can only be used in a combination with a group device label ("grpName"). End devices which are members of a device group can be spread over the Consist, or, for train wide groups, over the whole train.
opVehNo	S,D	M	Function in the vehicle with operational number $x \in \{01, \dots, 63\}$. This label can only be used in conjunction with the label "anyCst". The 'opVehno' is unique within the train and in reference to the leading vehicle (operational direction). EXAMPLE 'opVeh24'
cstVehNo	S,D	M	Function in the consist vehicle with cstVehNo $\in \{01, \dots, 32\}$ EXAMPLE 'veh05'
leadVeh	S,D	M	Function is in the leading vehicle. This label can only be used in conjunction with the label "leadCst"
idVeh	S,D	M	function in a vehicle with unique identifier (e.g. the 7/12 digit UIC vehicle number) EXAMPLE "UIC948002343044" // vehicle with UIC identifier 94800234304-4
anyVeh	D	M	function located in one of the vehicles. This label can be used when the location of the device in a consist is irrelevant. In this case, the device label shall be unique in the consist.

5.4.4.6.4 Consist Label

Consists shall be identified by labels following the syntax given in Table 12:

Table 12 – Consist label syntax

<code>consist = "lCst" "aCst" "anyCst" "leadCst" cstNo</code> <code> opCstNo idCst</code> <code>cstNo = "cst" digit digit</code> <code>opCstNo = "opCst" digit digit</code> <code>idCst = label</code>

The meaning of the different consist labels is defined in Table 13.

Table 13 – Consist label definition

Label <consist>	Scope S=source D=destination	Mandatory(M)/ Optional(O)	Definition
lCst	D	O	Function located in the own (local) consist
cstNo	S,D	M	Functions located in the consist with trnCstNo $\in \{01, \dots, 63\}$. EXAMPLE 'cst05' This is the consist sequence number obtained from the train directory which is not respecting corrections. For addressing functions in an operated train the label 'opCstNo' may be used instead which also respects correction.
opCstNo	S,D	O	Functions located in consist with operational number opCstNo $\in \{01, \dots, 63\}$. This label can only be used in conjunction with the label "anyVeh". This label is unique within the train and in reference to the leading Consist (operational direction). EXAMPLE 'opCst29'
aCst	D	M	functions located in some or all Consists. Can only be used in combination with a device group label.
anyCst	D	M	Function located in one of the Consists. This label can be used when the location of the device in the train is irrelevant, in which case, the device label shall be unique in the train, or this label shall be used if an operational vehicle is addressed. This label can only be used in combination with label "anyVeh" or in combination with label opVehNo.
idCst	S,D	M	unique identifier of a Consist EXAMPLE 'UIC948002343044' // Consist identified by its consist vehicle with UIC identifier 94 80 0234 304-4
leadCst	S,D	M	destination function is in the consist which contains the leading vehicle

5.4.4.6.5 Closed train Label (Optional)

Closed trains shall be identified by labels following the syntax given in Table 16:

Table 14 – Closed train label syntax

```
cltrain = "lClTrn" | "anyClTrn" | "aClTrn"
```

The meaning of the different closed train labels is defined in Table 17.

Table 15 – Closed train label definition

Label <cltrain>	Scope S=source D=destination	Mandatory(M)/ Optional(O)	Definition
lClTrn	S,D	O	destination/source function in the own closed train
anyClTrn	D	M	function located in one of the closed trains. This label may also be used if there is no closed train.
aClTrn	D	M	function located in all closed trains. This label may also be used if there is no closed train.

5.4.4.6.6 Train Label

Trains shall be identified by labels following the syntax given in Table 16:

Table 16 – Train label syntax

```
train = "lTrn"
```

The meaning of the different train labels is defined in Table 17.

Table 17 – Train label definition

Label <train>	Scope S=source D=destination	Mandatory(M)/ Optional(O)	Definition
lTrn	S,D	M	destination/source function in the own train

NOTE The URI scheme described herein is only defined for onboard train addressing, but it can be easily extended (and is prepared) for train-to-wayside and inter-train addressing. An extension can for instance look like:

```
trdp://instance.function@fctdev.vehicle.consist.cltrain.train.fleet.operator
```


5.4.5 Mapping TCN-URI to IP address

5.4.5.1 General

TCN-URI shall be mapped to IP addresses in accordance to the IP addressing scheme defined in IEC 61375-2-5.

The resolution of URIs to IP addresses is in general managed by the TCN-DNS service as specified in 5.4.2.

5.4.5.2 Mapping to IP MC group addresses

IEC 61375-2-5 assigns the IP address range 239.192.0.0/14 for train level IP MC group addressing. To use this address range in conjunction with the defined TCN-URI scheme, especially for supporting groups, which are limited to one consist, it is necessary to define a decomposition of the assigned IP MC group address range.

The general decomposition is defined as:

11101111.110000rr.zzzzzzzz.zzzzzzzz

with fields described in Table 18.

Table 18 – General decomposition of IP MC groups addresses

Subnet number part:	
[r]	Defines the range. '00'B = all-train groups '01'B = ETB-related groups '10'B = consist-limited groups '11'B = open for other use
[z]	For further decomposition

All-train groups are groups with group members related to different ETBs¹ in the train.

The decomposition for all-train groups is defined as:

11101111.11000000.gggggggg.gggggggg

with fields described in Table 19.

Table 19 – Decomposition of all-train groups

Host number part:	
[g]	All train group number (value range: 0 .. 65535) 0 .. 255 = as specified in IEC 61375-2-5 256 .. 65534 = usable for user defined all-train groups 65535 = reserved for future use

¹ For the definition of '... related to ETB' see 4.2.2.6.2.

EXAMPLE An IP MC group containing EDs from the operational network and the multimedia network is an all-train group.

An all-train group address shall be assigned to a function if FUNCTION_INFO parameter 'etbld' is set to 255 (see 5.3.3.2.4).

NOTE 1 Usage of TRDP for communication to all-train groups is problematic as all-train groups may contain EDs related to different ETBs, and etbTopoCnt values are different for the ETBs. The etbTopoCnt is contained in the TRDP header and used by destination EDs to verify the correctness of the received message, which will fail if an etbTopoCnt value related to another ETB is contained. Using a gateway in between ETB altering the etbTopoCnt value might be a solution, but is complex.

ETB-related groups are groups with group members related to one ETB.

The decomposition for ETB-related groups is defined as:

11101111.11000001.bbgggggg.gggggggg

with fields described in Table 20.

Table 20 – Decomposition of ETB-related groups

Subnet number part:	
[b]	ETB Identifier (etbld) value = 0..3
Host number part:	
[g]	Group number (value range: 0 .. 16383) 0 = all ED 1 = all ECSP (ETB0 only) 2..16382 = usable for user defined ETB-related groups 16383 = reserved for future use

EXAMPLE An IP MC group containing EDs from the operational network only is a ETB-related group. An IP MC group containing EDs from the multimedia network is another ETB-related group.

Consist-limited groups are ETB-related groups where all group members belong to one consist.

The decomposition for ETB-related groups is defined as:

11101111.11000010.bbgggggg.gggggggg

with fields described in Table 21.

Table 21 – Decomposition of consist-limited groups

Subnet number part:	
[b]	ETB Identifier (etbld) value = 0..3
[c]	trnCstNo (value range: 1 .. 63) 0 = local consist
Host number part:	
[g]	Group number (value range: 0 .. 254) 0 = all ED in all CN of the consist connected to the ETB 1 = ECSP within consist (ETB0 only) 2 .. 254 = usable for user defined consist-limited groups 255 = reserved for future use

Group identifiers need to be unique inside a consist. The same 8-bit group identifier value range is used for both vehicle-limited groups and consist-limited groups, so it must be shared. For example, the value of grpHMI is different for this group defined on consist level (“grpHMI.aVeh.IVeh”) and for this group defined on vehicle level (e.g. “grpHMI.veh02.ICst”).

Notice that “all ED” is coded by setting all grp-id bits to 0, and not to 1. This convention is used in order to avoid confusion, because grp-id bits set to 1 could also mean “all groups”.

5.4.5.3 Joining IP multicast groups

For joining IP MC groups some aspects should be considered.

- All-train and ETB-related groups have defined IP MC addresses which will not change when train composition changes, so it is sufficient for EDs to join the group when the system is booting.
- Consist-limited groups are dynamic IP Multicast groups possessing IP multicast group addresses which may change after each train inauguration. Due to their dynamic nature, all ED joining such a group need to leave the group with a new train inauguration, and join again with the new IP MC address. Another, recommended, solution is to translate IP MC addresses while routing IP telegrams from ETB to ECN (ETBN). During this network address translation (NAT, see RFC2663, RFC2766, RFC3022) the [c] bits representing the train consist number within the IP MC destination address can be substituted with 0 (local consist). In that case it is sufficient for EDs to join the group when the system is booting.
- EDs, which should be capable to receive TRDP telegrams addressed to both consist-limited as well as ETB-related or all-train groups, should join both IP MC groups.

EXAMPLE The function ‘fctDoor’ may be both train wide (ETB0-related) addressable (e.g. TCN-URI ‘fctDoor.anyVeh.aCst.aCItrn.ITrn’) as well as consist-limited (e.g. TCN-URI ‘fctDoor.anyVeh.cst01.anyCItrn.ITrn’). If we assume that NAT is used and that the fctId=50 for the fctDoor, the following two IP MC addresses need to be joined: 239.193.0.50 and 239.194.1.50.

5.4.5.4 Well known TCN-URI

Some TCN-URI mappings to IP addresses are predefined and do not require an explicit resolving of the URI via TCN-DNS. Those TCN-URI are listed in Table 22.

Table 22 – Well-known TCN-URI

TCN-URI (host part)	Scope	IP address	Description
grpAll.aVeh.aCst.aCItrn.ITrn	D	239.193.0.0	Broadcast to all end devices related to ETB0.
grpAll.aVeh.ICst.ICITrn.ITrn	D	239.194.0.0	Broadcast to all end devices related to ETB0 within the local consist
grpAll.aVeh.ICst.ICITrn.ITrn	D	239.255.0.0	Alternative broadcast to all end devices within the local consist if all ED are connected to the same CN (only one CN connected to the ETB) NOTE this address is defined in IEC 61375-3-4
IDev.IVeh.ICst.ICITrn.ITrn	S,D	127.0.0.1	Own device (local loop-back)
grpECSP.anyVeh.aCst.aCItrn.ITrn	D	239.193.0.1	Broadcast to all ECSP

EXAMPLE

The following example gives some hints for the implementation of grpAll.aVeh.ICst.ICITrn.ITrn in a consist with multiple CN.

Every ED in a consist subscribes to 239.194.0.0.

ETBNs use ETB inauguration result to add a rule to their MC routing table for address 239.194.X.0 where X represents their own consist (X = trnCstNo + (etbld × 64)). All other addresses from range 239.194.X.0 are not routed to ECN.

ETBNs translate MC address 239.194.X.0 to 239.194.0.0 when routing it from ETB to ECN.

In case of multiple CNs in one consist: packet with destination address 239.194.0.0 has to be handled by ETBNs in this consist as well in order to deliver it to all CNs. One solution is to route it from source CN to ETB and translate the destination address to 239.194.X.0. Another solution is that the ETBN in the source CN tunnels the packet to other ETBNs in the consist using IP unicast, and the destination ETBNs then retransmit the packet as multicast into their CN. The first approach is easier to implement but the packet is sent to all ETBNs in the train. Tunneling approach requires more support from ETBNs, on the other hand the packet never leaves consist boundary.

5.4.5.5 IP routing in networks with multiple ETBs

Basic rules for dynamic IP routing management are defined in IEC 61375-2-5.

The following additional rule shall be observed:

For transmitting IP packets to a destination ED, which is related to another ETB than the ETB the source ED is related to, the IP packet shall be transferred over the ETB the destination ED is related to.

NOTE 1 For a definition of 'ED related to ETB' see 4.2.2.6.2.

NOTE 2 The reason for this rule is that a sending ED cannot expect that an IP packet can be routed between the ETBs within the destination consist. It therefore needs to be routed within the sender's consist to the ETB the destination ED is related to.

EXAMPLE This example uses the train architecture shown in Figure 21. If the source ED in consist 'B' wants to send to the ED with fctDevNo = 10020 in consist 'E', it has to route the packet first to its local multimedia ECN and then via the ETB1 to consist 'E'.

5.4.6 Support of other URI schemas

Besides the defined TCN-URI schema, which is mainly targeting the functional addressing inside trains, also other URI schemas like the commonly used uniform resource locators (URL) with the syntax²

scheme://domain:port/path?....."

may be used. The 'domain' corresponds to the TCN-URI host part and may be resolved by the TCN-DNS service. This way, other URI schemas than TCN-URI can be supported, which might be requested for specific OMTS applications.

EXAMPLES

HTTP request: <http://devECSP.anyVeh.ICst.ITrn:80>

Mail service: <mailto:bob@devECSP.anyVeh.ICst.ITrn>

5.5 TCN-DNS Server

5.5.1 General

TCN-URIs host part are resolved to IP addresses with the aid of the TCN-DNS as defined in 5.4.2. The interaction between DNS clients (the "resolver") and the local DNS server is specified in RFC 1034 and RFC 1035.

A more efficient interface to the local DNS server especially suited to resolve multiple TCN-URIs is described in Annex E.

² A URL is technically a type of uniform resource identifier (URI) as defined in RFC3986.

5.5.2 Architecture

Each consist shall provide at least one consist DNS name server which is authoritative for this consist domain.

NOTE The consist domain may be subdivided into several zones where each zone has its own DNS name server. A special case is a consist with separated operational and multimedia network, in which case two dedicated DNS name servers, one for the operational network and one for the multimedia network, can be provided.

5.5.3 Functional address resolution

The consist DNS server shall be able to resolve TCN-URI host parts as defined in 5.4.4 to the corresponding IP address as defined in IEC 61375-2-5.

EXAMPLES

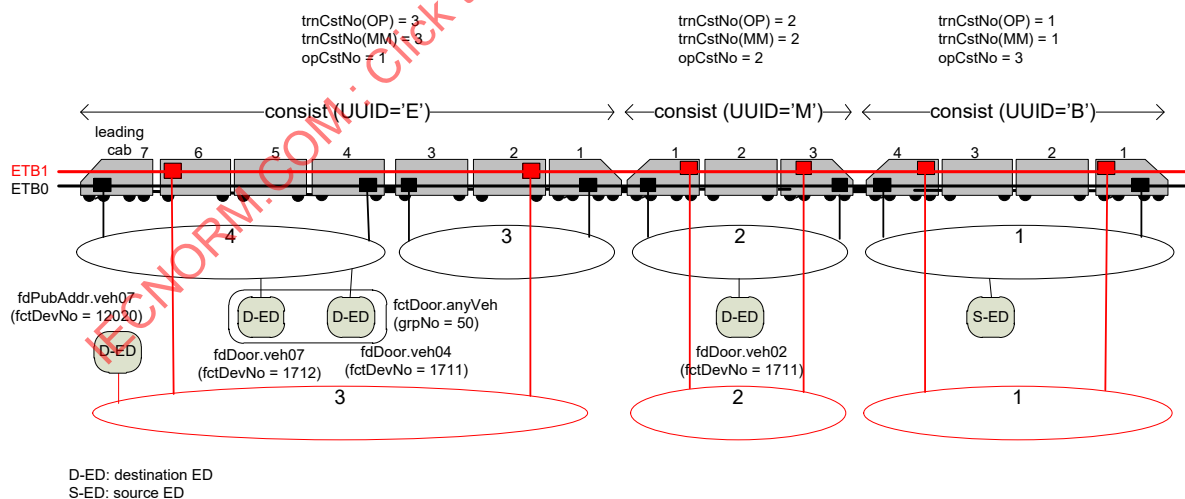
Given a train with two train backbones, ETB0 (operational network 'OP') and ETB1 (multimedia network 'MM'), composed of three consists, as shown in Figure 21. TCN reference directions are equal for ETB0 and ETB1. Due to the position of the leading cab the operational consist numbers are opposite to the TCN consist numbers of ETB0. The consist with UUID='E' has two ECN connected to ETB0 and one ECN connected to ETB1. The other consists have only one ECN for each ETB. Furthermore, it is assumed that there are two DNS name servers, one for the operational network and one for the multimedia network, and that end devices connected to either of the two networks use the related DNS name server for address resolution. There exists a TDB for each ETB with partly shared information following the rules defined in 5.3.4.

The task will be for a source ED located in consist with UUID = 'B' and connected to the operational network to resolve the TCN-URIs of the two shown destination end devices:

- ED [doorCTRL@fdDoor.veh02.cst02.anyCITrn.ITrn](#) in consist with UUID = 'M'
- ED [publicAnnounce@fdPubAddr.anyVeh.leadCst.anyCITrn.ITrn](#) in consist with UUID = 'E' (leading vehicle)

and in addition, to resolve the group addresses of the door controller group (only shown for consist 'E' in Figure 21), as a train wide group and as a consist-limited group:

- group [doorCTRL@fctDoor.anyVeh.aCst.aCITrn.ITrn](#) in the train
- group [doorCTRL@fctDoor.anyVeh.cst03.anyCITrn.ITrn](#) in consist 'E'



IEC

Figure 21 – TCN-URI resolving in a train

EXAMPLE 1

For resolving the TCN-URI 'doorCTRL@fdDoor.veh02.cst02.anyCITrn.ITrn' the ETB0 DNS server has to execute the steps as listed in Table 23.

Table 23 – TCN-URI resolving – Example 1

Step	Action	Result (example)
1	Isolate the TCN-URI host part and strip off parts not needed	= "fdDoor.veh02.cst02"
2	Lookup 'cstUUID' of label 'cstNo' = 2 in the train directory of the local TTDB	= "M"
3	Search for 'cstUUID' in the ConsistInfoList	Located CONSIST_INFO record for consist "M"
4	Search for ['fctName' = "fdDoor"] in conjunction with [cstVehNo = 02] in the function info array of the located CONSIST_INFO record and retrieve the value of parameters 'fctId', 'etbNo' and 'cnNo'.	fctId = 1711 etbNo = 0 cnNo = 1
5	The value of 'subnet id' can be derived from the train network directory associated to the ETB0	subnet id = 2
6	Compute the IP address from the parameter values 'etbNo', 'subnet id' and 'fctId' in accordance to IEC 61375-2-5	IP = 10.128.134.175

EXAMPLE 2

For resolving the TCN-URI 'publicAnnounce@fdPubAddr.anyVeh.leadCst.ITm' the ETB0 DNS server has to execute the steps as listed in Table 24:

Table 24 – TCN-URI resolving – Example 2

Step	Action	Result (example)
1	Isolate the TCN-URI host part and strip off parts not needed	= "fdPubAddr.anyVeh.leadCst"
2	Lookup the leading vehicle in the operational train directory and retrieve the values of parameters 'opVehNo' and 'ownOpCstNo'.	opVehNo = 1 ownOpCstNo = 1
3	Lookup 'cstUUID' of 'opCstNo' = 1 in the operational train directory	= "E"
4	Determine the consist orientation with respect to the operational train orientation by reading parameter 'opCstOrient' in the operational train directory	= inverse
5	Search for 'cstUUID' in the ConsistInfoList	Located CONSIST_INFO record for consist "E"
6	Determine the number of vehicles within that consist (parameter 'vehCnt') and compute the 'cstVehNo' of the leading vehicle.	Number of vehicles = 7. As the consist is inverse to the operational train direction, the value of 'cstVehNo' of the leading vehicle with opVehNo=1 is cstVehNo = 7
7	Search for ['fctdevName' = "fdPubAddr" AND 'cstVehNo' = 7] in the function info array of the located CONSIST_INFO record and retrieve the value of parameters 'fctId', 'etbNo' and 'cnNo'. NOTE The function fdPubAddr is associated to the multimedia network. To make it visible in the ETB0 TTDB this function is listed both in the CONSIST_INFO record of the ETB0 TTDB and in the CONSIST_INFO record of the ETB1 TTDB, see 5.3.4.4.	fctId = 12020 etbNo = 1 cnNo = 1
8	The value of 'subnet id' can be derived from the train network directory associated to the ETB1	subnet id = 3
9	Compute the IP address from the parameter values 'etbNo', 'subnet id' and 'fctId' in accordance to IEC 61375-2-5	IP = 10.160.238.244

EXAMPLE 3

The following example shows how a TCN-URI can be resolved to a train wide multicast group address. Here it is assumed that the IP MC group 'fctDoor' has been defined as a train wide group with a group Id = 50. For resolving the TCN-URI 'doorCTRL@fctDoor.anyVeh.aCst.aCITrn.ITrn' the ETB0 DNS server has to execute the steps as listed in Table 25.

Note that train wide multicast group addresses must be unique train wide.

Table 25 – TCN-URI resolving – Example 3

Step	Action	Result (example)
1	Isolate the TCN-URI host part and strip off parts not needed	= "fctDoor"
2	Search for the first match of ['fctName' = "fctDoor"] in the consist info list by iterating through the CONSIST_INFO records and retrieve the value of parameters 'fctId'.	fctId = 50
3	Compute the IP address from the parameter value 'fctId' in accordance to the scheme defined above	IP = 239.193.0.50

EXAMPLE 4

The following example shows how a TCN-URI can be resolved to a consist-limited multicast group address. Here it is assumed that the two door controllers in consist E form a multicast group with group Id = 50. For resolving the TCN-URI 'doorCTRL@fctDoor.anyVeh.cst03.anyCITrn.ITrn' the ETB0 DNS server has to execute the steps as listed in Table 26:

Table 26 – TCN-URI resolving – Example 4

Step	Action	Result (example)
1	Isolate the TCN-URI host part and strip off parts not needed	= "fctDoor.anyVeh.cst03"
2	Lookup 'cstUUID' of label 'cstNo' = 3 in the train directory of the local TTDB	= "E"
3	Search for 'cstUUID' in the ConsistInfoList	Located CONSIST_INFO record for consist "E"
4	Search for ['fctName' = "fctDoor"] in the function info array of the located CONSIST_INFO record and retrieve the value of parameter 'fctId'	fctId = 50
5	Compute the IP address from the parameter values 'trnCstNo' and 'fctId' in accordance to the scheme defined above	IP = 239.194.1.50

5.5.4 Protocol

The consist DNS name server shall implement the DNS protocols as specified in RFC 1034 and RFC 1035.

The consist DNS name server shall accept recursive client queries as shown in the example of Figure 22.

TCN-URIs addressing destinations within the local or remote consists shall be resolved within a time of 1,0 s.

NOTE The time is measured from the query reception until sending the answer to the client is completed.

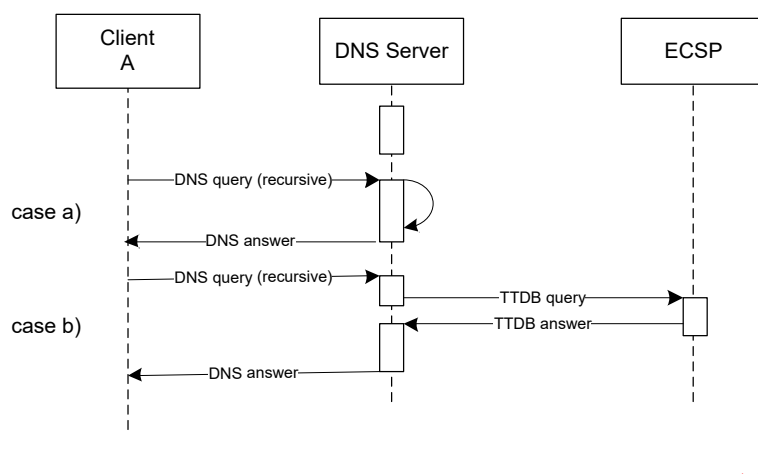


Figure 22 – DNS protocol (case a without, case b with TTDB interrogation)

5.5.5 Multiple ETBs

The mapping of TCN-URI to IP addresses is ETB specific for inter-consist communication as it depends on the train network directory of the specific ETB. Therefore, a TCN-DNS server shall be provided for each ETB.

A function shall use the TCN-DNS server for TCN-URI address resolution, which is related to the ETB the functions wants to communicate.

NOTE Addressing a function in a different consist may result in different IP addresses depending on the ETB which is used for communication.

Take care that the return path of a inter-consist communication will use the same ETB if for returned telegrams the IP address of the original sender is used as destination address.

5.6 Data exchange

5.6.1 General

This subclause defines the data exchange between functions and devices over ETB. Dependant on the involved functions, different communication protocols might be applied. In this standard, only communication protocols are defined which are used by the services specified herein.

5.6.2 Operational network communication

All services of the operational network shall use the TRDP protocol as specified in Annex A (with supplementary information in Annex C and Annex D) for the transfer of process data and message data between operational network functions.

For the transfer of streaming data (audio/video), appropriate streaming data protocols may be used, as for instance (list not exhaustive):

RTSP	Real Time Streaming Protocol	RFC 2326
RTCP	Real Time Control Protocol	RFC 3605
RTP	Real Time Transport Protocol	RFC 3550

SRTP	Secure Real Time Transport Protocol	RFC 3261
SIP	Session Initiation Protocol	RFC 3711
TLS	Transport Layer Security Protocol	RFC 5246

For non-operational services like file transfer or service access, appropriate data protocols may be used, as for instance (list not exhaustive):

FTP	File Transfer Protocol	RFC 956
SFTP	SSH (Secure) File Transfer Protocol	RFC diverse
SNMP	Simple Network Management Protocol	RFC diverse
SSH	Secure Shell	RFC diverse
SCP	Secure Copy Protocol	-

5.6.3 OMTS network communication

In principle, the same protocols as used within the operational network can also be used for the OMTS network. Due to the special nature of the OMTS network as a platform for multimedia and telematics services, additional protocols may be used, as for instance (list not exhaustive):

HTTP	Hypertext Transfer Protocol	RFC 2616
SOAP	Simple Object Access Protocol	World Wide Web Consortium (W3C)

5.6.4 Quality of Service (QoS)

IEC 61375-1 defines a set of data classes, which shall be supported by ETB, and IEC 61375-2-5 defines a minimum of 4 priorities for the data transfer over ETB. The assignment of priorities to data classes shall be made, if not otherwise defined, in accordance to Table 27.

Table 27 – Data class priorities

Data Class	Priority	Remarks
Process Data	3	
Message Data	0 .. 1	
Stream Data	1 .. 2	The choice of priorities may depend on the type of the stream, e.g. an audio stream might require a higher priority than a video stream
Best Effort Data	0	
Supervisory Data	3	This data class is only defined in IEC 61375-2-5
3 = highest priority, 0 = lowest priority		

5.7 Service discovery

Each consist shall announce the functions it supports within its CSTINFO telegram. All supported functions can then be retrieved from the TTDB.

5.8 Train Info Service

The trainInfo service defines a consist internal service which allows to retrieve data from the TTDB. It is not within the scope of this standard to define consist internal services, but a useful set of function primitives is given in Annex E for information.

6 Services of the communication profile – ETB Control Service

6.1 General

This subclause specifies the service functions which are basically needed for enabling user data transfer over ETB. These functions are:

- Function Leading
- Function Sleep Mode
- Function Confirmation/Correction

In case of multiple ETBs, e.g. one operational network and one multimedia network, those functions shall only be activated on the operational network.

NOTE This restriction avoids contradicting comma.

6.2 Communication model

ETB control functions follow the client-server model as depicted in Figure 23. The ETB control service client (ECSC) is requesting some function primitive from the ETB control service provider (ECSP), and the ECSP is responding the function execution result.

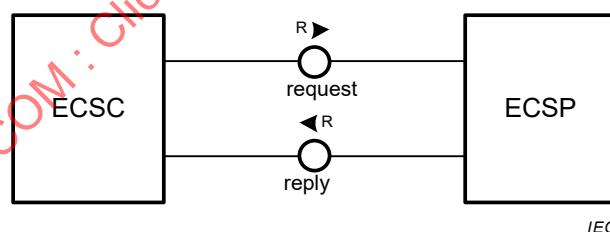


Figure 23 – ETB control service model

Only one ECSP shall be active at one time within a consist.

It is recommended to implement multiple ECSP, e.g. combined with an ETBN, within a consist and to select one of the ECSP within the consist to become the active ECSP.

This part of the standard makes no assumption about the client of this service. For the specification of the service, an abstract client (ECSC) is modeled which is invoking the service functions and also retrieving the function results.

In general it is recommended to have only one device within each consist dedicated as ETB control service client in order to avoid conflicting server operations, for example one device requesting a certain correction of the train composition and another device requesting another correction.

6.3 ECSP Supervision

The ECSP shall supervise the availability of the ECSC.

A failure of the ECSC shall be detected latest after a time of $T_{ECSC\_fail} = 5,0$ s. If a failure is detected, the ECSP shall react as defined for the individual ETB control service functions.

NOTE Annex E defines an interface between ECSC and ECSP for the services specified herein.

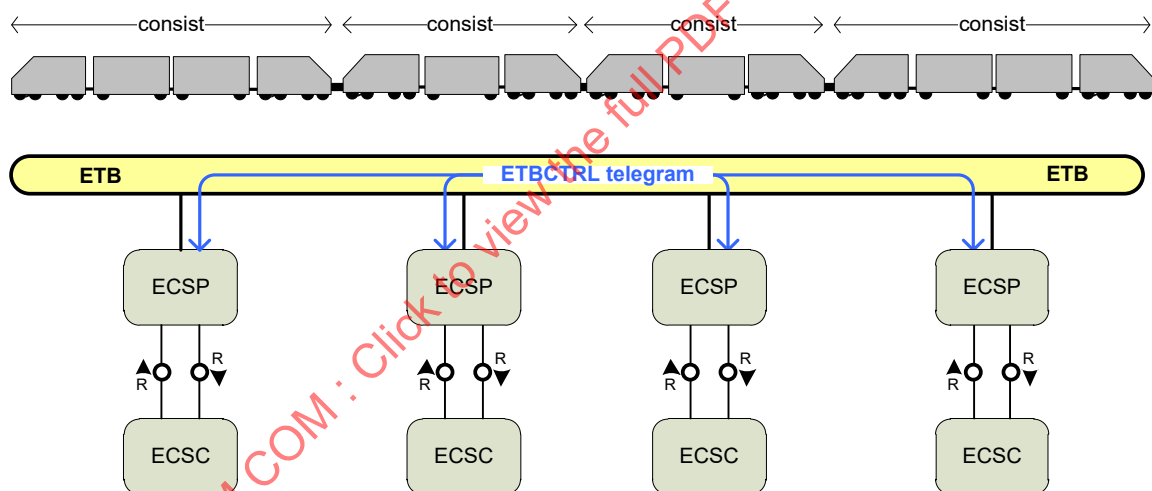
6.4 ECSP Interconnection

6.4.1 General

All ETB control functions require data exchange between all the ECSPs within a train. These are all the ECSPs which are related to a consist listed in the train directory with a valid *cstTopoCnt*.

For data exchange between the ECSPs, each ECSP sends cyclically a ETBCTRL telegram to all ECSPs within the train as shown in Figure 24. The data exchange can be 'regular' for non-safety relevant application or 'safe' for safety related applications.

In case of multiple ETBs, e.g. one operational network and one multimedia network, ETBCTRL telegrams shall only be exchanged on the operational network (ETB0).



IEC

Figure 24 – ETBCTRL telegram exchange

6.4.2 ETBCTRL telegram exchange selection

An ECSP shall support either the regular ETBCTRL telegram transmission or the safe ETBCTRL telegram transmission using SDTv2 as defined in Annex B.

An ECSP implementing the safe ETBCTRL telegram transmission may also process received regular ETBCTRL telegrams.

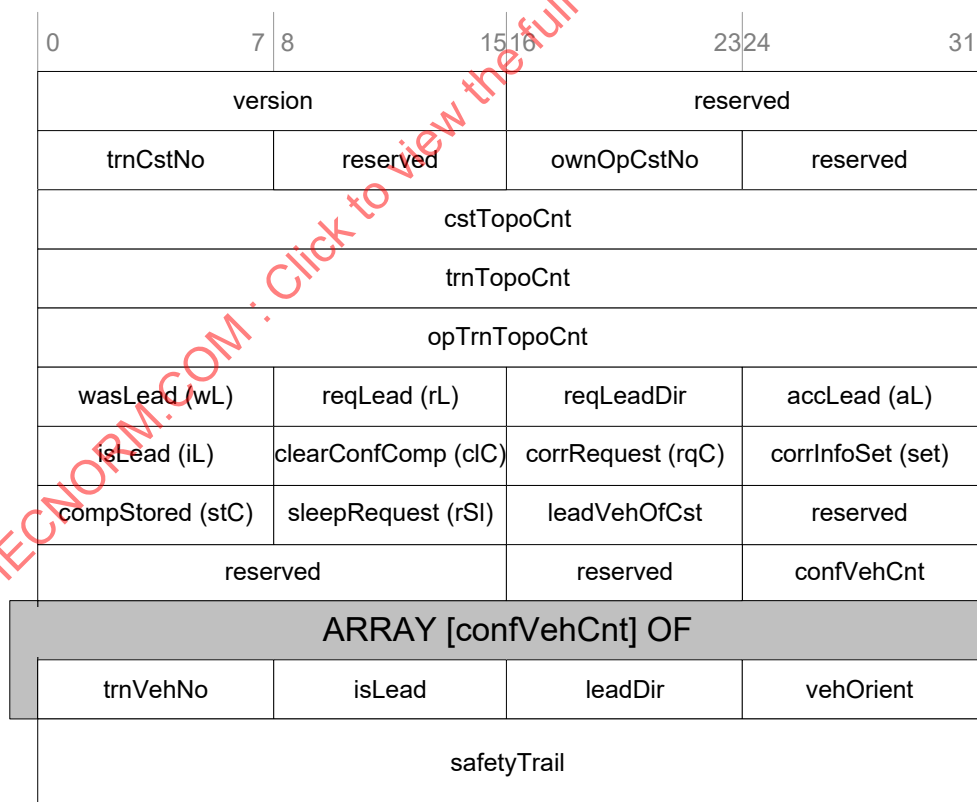
6.4.3 ETBCTRL telegram transmission

TRDP PD telegram parameters:

Message type:	'Pd'
ComId:	1
SMI:	1 (only needed if SDTv2 is used)
UserDataVersion:	'0100'H (only needed if SDTv2 is used)
DestinationURI (host part):	grpECSP.anyVeh.aCst.aCITrn.ITrn
Dataset:	ETBCTRL_TELEGRAM
Cycle time:	(0,5 ± 0,1) s
Timeout:	3,0 s

6.4.4 Structure of the ETBCTRL telegram

The structure of the ETBCTRL telegram is shown in Figure 25.



IEC

Figure 25 – ETBCTRL telegram

```

CONFIRMED_VEHICLE_ ::= RECORD
{
    trnVehNo      UINT8      -- vehicle sequence number within the train
                             with vehicle 01 being the first vehicle
                             in ETB reference direction 1 as defined in
                             IEC 61375-2-5
                             value range: 0..63
                             a value of 0 indicates that this vehicle
                             has been inserted by correction
    isLead        ANTIVALENT8 -- vehicle is leading, informal
                             may be used for plausibility checks
    leadDir       UINT8      -- vehicle leading direction, informal
                             0 = not relevant
                             1 = leading direction 1
                             2 = leading direction 2
                             may be used for plausibility checks
    vehOrient     UINT8      -- vehicle orientation, informal
                             0 = not known (corrected vehicle)
                             1 = same as operational train direction
                             2 = inverse to operational train
                             direction
                             may be used for plausibility checks
}

```

Informal parameters in the telegram may be used for a comparison between the own vehicle list and the indicated vehicle list as a plausibility check in addition to the opTopoCnt value check.

```

ETBCTRL_VDP ::= RECORD
{
    reserved01     UINT32      -- reserved, shall be set to 0
    reserved02     UINT16      -- reserved, shall be set to 0
    userDataVersion VERSION    -- version of the vital ETBCTRL telegram
                             mainVersion = 1
                             subVersion = 0
    safeSequCount  UINT32      -- safe sequence counter, as defined in B.9
    safetyCode     UINT32      -- checksum, as defined in B.9
}

```

```

ETBCTRL_TELEGRAM ::= RECORD
{
    version        VERSION    -- data structure version
                             parameter 'mainVersion'
                             shall be set to 1.
    reserved01     UINT16      -- reserved (=0)
    trnCstNo       UINT8      -- own train consist number
                             value range: 1..63
    reserved02     UINT8      -- rsv: reserved (=0)
    ownOpCstNo     UINT8      -- own operational number
                             value range: 1..63
                             0 = unknown (e.g. after inauguration)
    reserved03     UINT8      -- rsv: reserved (=0)
    cstTopoCnt     UINT32      -- consist topography counter
    trnTopoCnt     UINT32      -- train directory topography counter
    opTrnTopoCnt   UINT32      -- operational Train directory topography
                             counter
    wasLead        ANTIVALENT8 -- wL: consist was leading control flag
}

```

```

                                '01'B = FALSE
                                '10'B = TRUE
reqLead      ANTIVALENT8      -- rL: leading request control flag
                                '01'B = FALSE
                                '10'B = TRUE
reqLeadDir   UINT8           -- rLD: (request) leading direction
                                0 = not relevant
                                1 = consist direction 1
                                2 = consist direction 2
accLead      ANTIVALENT8      -- aL: accept remote leading request
                                control flag
                                '01'B = FALSE, not accepted
                                '10'B = TRUE, accepted
isLead       ANTIVALENT8      -- iL: consist contains leading vehicle
                                control flag
                                '01'B = FALSE
                                '10'B = TRUE
clearConfComp ANTIVALENT8      -- clC: clear confirmed composition
                                control flag
                                '01'B = 0:FALSE
                                '10'B = 1:TRUE
corrRequest  ANTIVALENT8      -- rqC: request confirmation control flag
                                '01'B = 0:FALSE
                                '10'B = 1:TRUE
corrInfoSet  ANTIVALENT8      -- set: correction info set control flag
                                '01'B = 0:FALSE
                                '10'B = 1:TRUE
compStored   ANTIVALENT8      -- stC: corrected composition stored
                                control flag
                                '01'B = 0:FALSE
                                '10'B = 1:TRUE
sleepRequest ANTIVALENT8      -- rSl: request sleep mode control flag
                                '01'B = FALSE
                                '10'B = TRUE
leadVehOfCst UINT8           -- position of leading vehicle in consist
                                value range 0..32
                                0 = not defined
                                (1 = first vehicle in consist in
                                Direction 1, 2 = second vehicle, etc.)
reserved04   UINT8           -- rsv: reserved (=0)
reserved05   UINT16          -- rsv: reserved (=0)
reserved06   UINT8           -- rsv: reserved (=0)
confVehCnt   UINT8           -- number of confirmed vehicles in train
                                value range: 0..63,
                                0 = no correction requested
confVehList  ARRAY [confVehCnt] OF CONFIRMED_VEHICLE
                                -- dynamic ordered list
                                of confirmed vehicles in train,
                                starting with the vehicle at
                                train extremity 1 (see 4.2.4.2)
safetyTrail  ETBCTRL_VDP     -- ETB-VDP trailer
                                SafeTopoCount = 0
                                completely set to 0 = SDTv2 not used
}

```

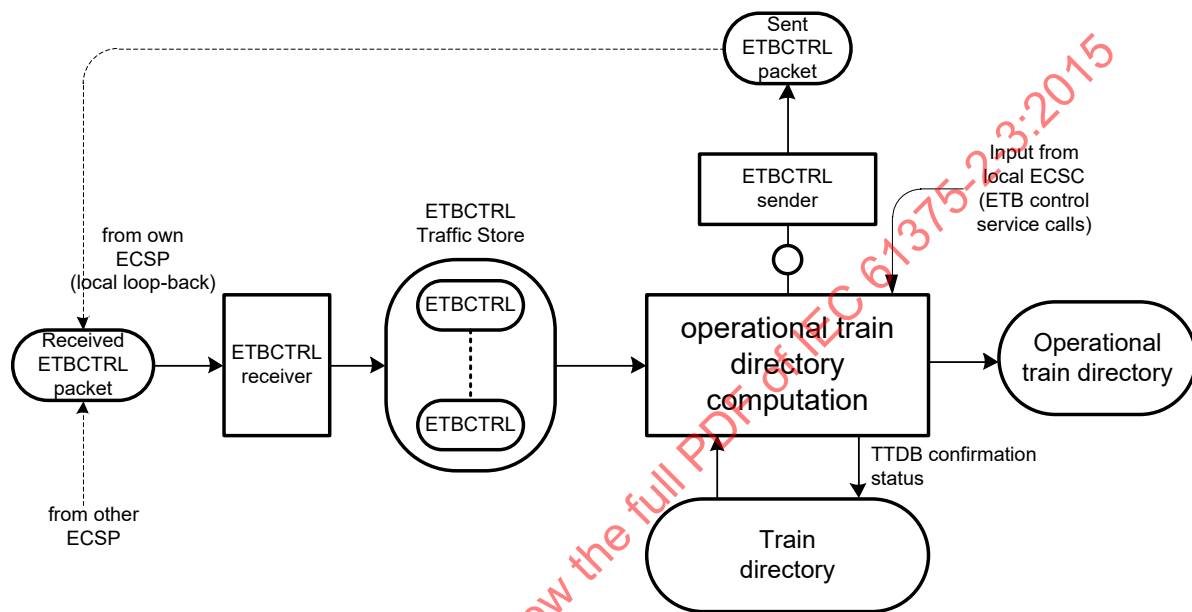
NOTE The parameter 'reqLeadDir' is only meaningful if one of the parameters 'reqLead', 'wasLead' or 'isLead' is set to TRUE. This parameter is relevant for operational train direction determination.

6.4.5 Operational train directory computation process

6.4.5.1 Overview

The ETB control service functions “leading” and “confirmation/correction” determine the content of the operational train directory. The computation of the operational train directory is a function on each ECSP connected to ETB0, which is responsible to create and to maintain the operational train directory as defined in 5.3.3.

The principal operation of the operational train directory computation is shown in Figure 26.



IEC

Figure 26 – Operational train directory computation block diagram

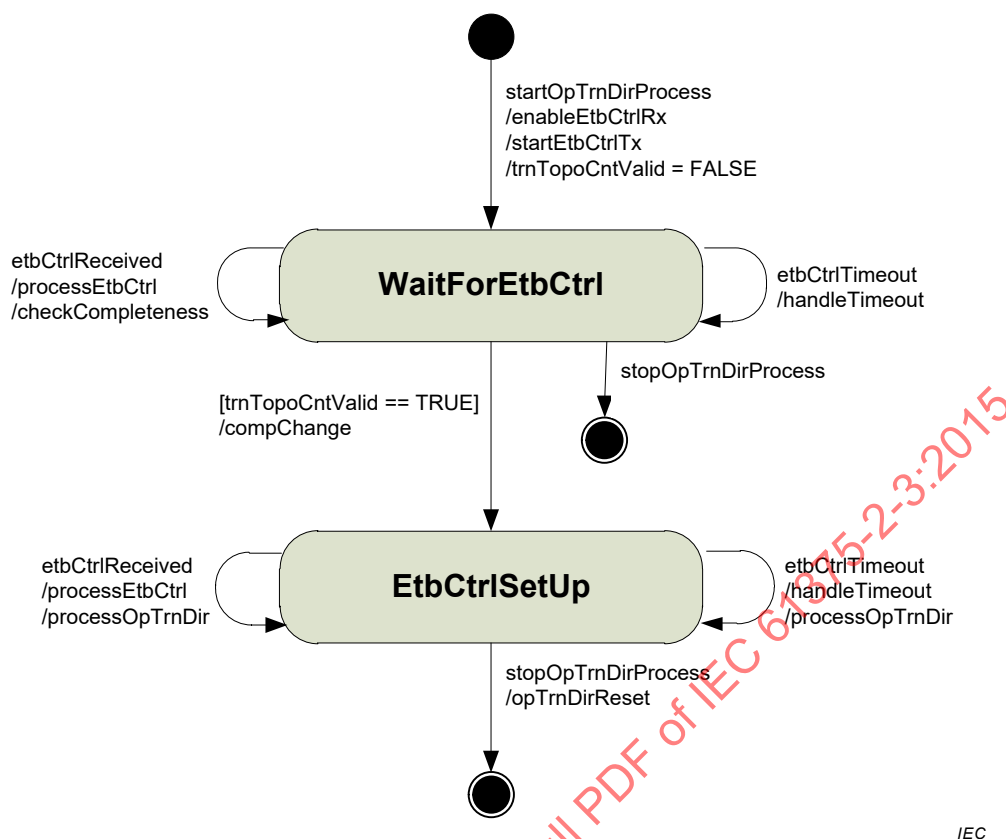
Input to the operational train directory computation is the collection of ETBCTRL telegrams received from all the ECSPs. This includes the own (local) ECSP as well as the remote ECSPs (= all the others except the own ECSP).

Those ETBCTRL telegrams might for instance be stored to an ETBCTRL traffic store for further processing. The operational train directory computation checks the content of the ETBCTRL traffic store for changes with respect to leading vehicle information, correction information and trnTopoCnt.

If a change is detected, a new operational train directory shall be computed in accordance to the rules given later in this subclause and the confirmation status of the TTDB shall be updated (if needed). In addition, actualized input to the ETBCTRL telegram sent by the local ECSP shall be provided.

6.4.5.2 State diagram

The operational train directory computation process is defined by the state diagram shown in Figure 27 and further described in Table 28 to Table 30.



IEC

Figure 27 – ETBCTRL processing state diagram

Table 28 – ETBCTRL processing – triggers

Trigger	Description
startOpTrnDirProcess	Start processing of the operational train directory, triggered by the train directory computation state machine, see .5.3.2.
stopOpTrnDirProcess	Stop processing of the operational train directory, triggered by the train directory computation state machine, see .5.3.2. If the transmission of ETBCTRL telegram continues, telegram parameter 'trnTopoCnt' shall be set to 0 until processing of the operational train directory is restarted (trigger 'startOpTrnDirProcess')
etbCtrlReceived	ETBCTRL telegram received
etbCtrlTimeOut	ETBCTRL telegram timeout

Table 29 – ETBCTRL processing – guards

Guard	Description
[trnTopoCntValid == TRUE]	ETBCTRL telegrams have been received from all active ECSP listed in the local train directory with cstTopoCnt != 0 and all received valid ETBCTRL telegrams (including own ETBCTRL telegram) have a value of trnTopoCnt identical to the trnTopoCnt value stored in the local train directory.

Table 30 – ETBCTRL processing – actions

Action	Description
opTrnDirReset	The operational train directory shall be reset as defined in 6.7.3 (trigger event 'opTrnDirReset')
enableEtbCtrlRx	Prepare to receive ETBCTRL telegrams from all consist ECSPs listed in the train directory with a valid cstTopoCnt and establish individual

Action	Description
	timeout supervision for each of those consist ECSPs. Set receive timeout to a value of $(3,0 \pm 0,1)$ s. The timers shall be activated after the reception of the first related ETBCTRL telegram.
startEtbCtrlTx	Start to cyclically send ETBCTRL telegrams Cycle time: $(0,5 \pm 0,1)$ s Once ETBCTRL telegram transmission is started it can run forever, meaning that it needs not to be stopped if trigger 'stopOpTrnDirProcess' activates.
processEtbCtrl	Received ETBCTRL telegrams shall be processed as follows: 1) ETBCTRL telegrams originating from an ECSP which is not listed in the train directory (unknown cstTopoCnt value) or listed with cstTopoCnt = 0 shall be discarded. 2) ETBCTRL telegrams with a parameter 'trnTopoCnt' value different to the value of parameter 'trnTopoCnt' stored in the local train directory shall be discarded. 3) Control flags and other relevant parameters from non-discarded ETBCTRL telegrams shall be memorized, e.g. stored in the traffic store
checkCompleteness	trnTopoCntValid shall be set to TRUE if (non-discarded) ETBCTRL telegrams have been received from all ECSPs listed in the train directory with cstTopoCnt != 0 except from those ECSPs which are marked as being timed out.
processOpTrnDir	Start operational directory computation process: • The leading vehicle function state machine as defined in 6.5. • The confirmation/correction function state machine as defined in 6.6 • Supporting operations as defined in 6.7 This computation process can be executed event driven (as indicated in the state diagram) or cyclic.
handleTimeout	Mark the related source consist as being timed out. Set the (locally stored) control flags of the related source ECSP to deferred values (e.g. in traffic store). Deferred values are defined: • in 6.5 for the leading function • in 6.6 for the confirmation/correction function • in 6.8 for the sleep mode function
compChange	Trigger a 'compChange' event: • in 6.5 for the leading function • in 6.6 for the confirmation/correction function

6.5 Function "Leading"

6.5.1 General

This function allows to elect one of the vehicles within the train to become the leading vehicle. The leading vehicle is typically the vehicle with the activated drivers desk. The function shall ensure, that at any time maximally one vehicle receives the property "leading". In some special cases, as for instance after coupling of two trains, two vehicles may have this property, which leads to a leading conflict. Such a leading conflict needs to be detected and indicated to the user. It is the users responsibility to resolve the leading conflict, e.g. by deactivating one of the active driver's cabs.

6.5.2 Function primitives

The function primitives defined in this section (Table 31 to Table 33) and shown in Figure 28 shall be supported.

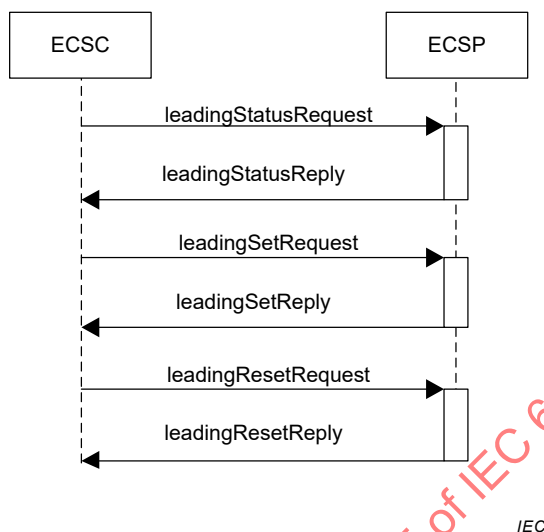


Figure 28 – Leading sequence diagram

Table 31 – Leading function primitives – F_leadingStatusRequest

Function primitive:	F_leadingStatusRequest
Description:	Read the leading status of the own consist
Request parameter:	-
Response parameter:	isLeading = yes/no wasLeading = yes/no leadingDir = Dir1/Dir2 pendingLeadingRequest = yes/no leadingConflict = yes/no leadingVehicle = n th vehicle of leading consist

Table 32 – Leading function primitives – F_leadingSetRequest

Function primitive:	F_leadingSetRequest
Description:	Request to become leading vehicle. Parameter leadingDir indicates the main viewing direction of the driver's cab. NOTE In vehicles with two driver's cabs it also defines at which vehicle end the leading driver's cab is located
Request parameter:	leadingDir = Dir1/Dir2
Response parameter:	-

Table 33 – Leading function primitives – F_leadingResetRequest

Function primitive:	F_leadingResetRequest
Description:	Remove request or remove property “leading”
Request parameter:	-
Response parameter:	-

6.5.3 ECSP to ECSP protocol

6.5.3.1 General

This defines the protocol for service related data exchange between the ECSP peers.

The leading function uses four parameters (control flags) of the ETBCTRL telegram for control as listed in Table 34.

Table 34 – Leading function control flags

control flag	short form	description	deferred value
reqLead	rL	client request or abandon leading property	FALSE
accLead	aL	accept a remote leading request	Don't care
isLead	iL	consist is leading	FALSE
wasLead	wL	consist was leading previously	unmodified NOTE Under certain conditions this value can change to 'FALSE', see note below
opTrnTopoCnt	-	Computed operational train topography counter value	Don't care

Key

Unmodified = previously stored value to be kept

Don't care = this value shall not be used in computation (to be ignored)

NOTE In case the leading consists stops sending ECSP telegrams the leading property gets lost and a new operational train directory is computed without a leading consist. However, because the 'wasLead' control flag remains unmodified no change of the operational train direction will occur. If thereafter a new leading vehicle is elected, this control flag is set to 'FALSE', see 6.5.3.2.

The state of the leading vehicle function is determined by the set of flags received from the other consists (remote leading control flags, sent by the consists listed in the train directory with an cstTopoCnt != 0) and the own set of flags (own leading control flags) as it is shown in Figure 29.

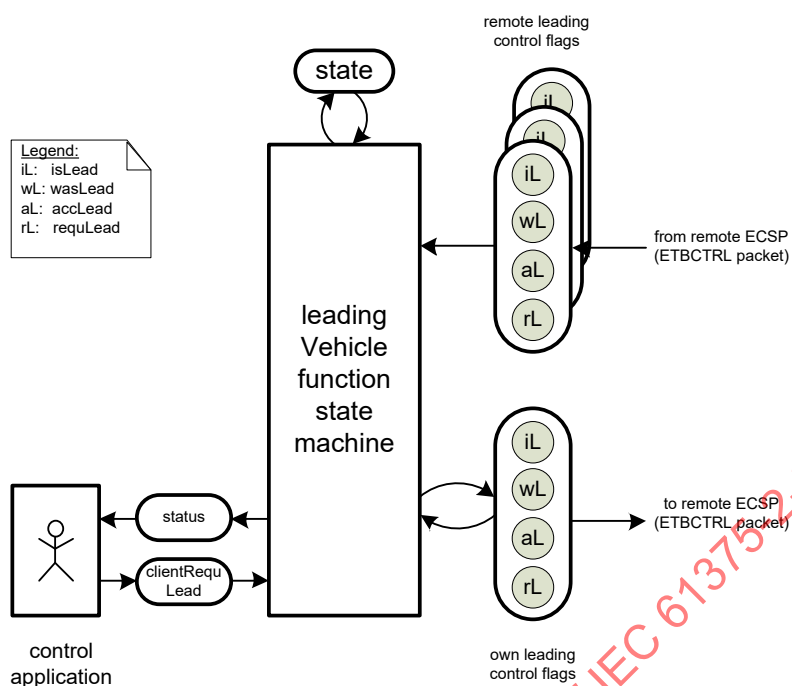


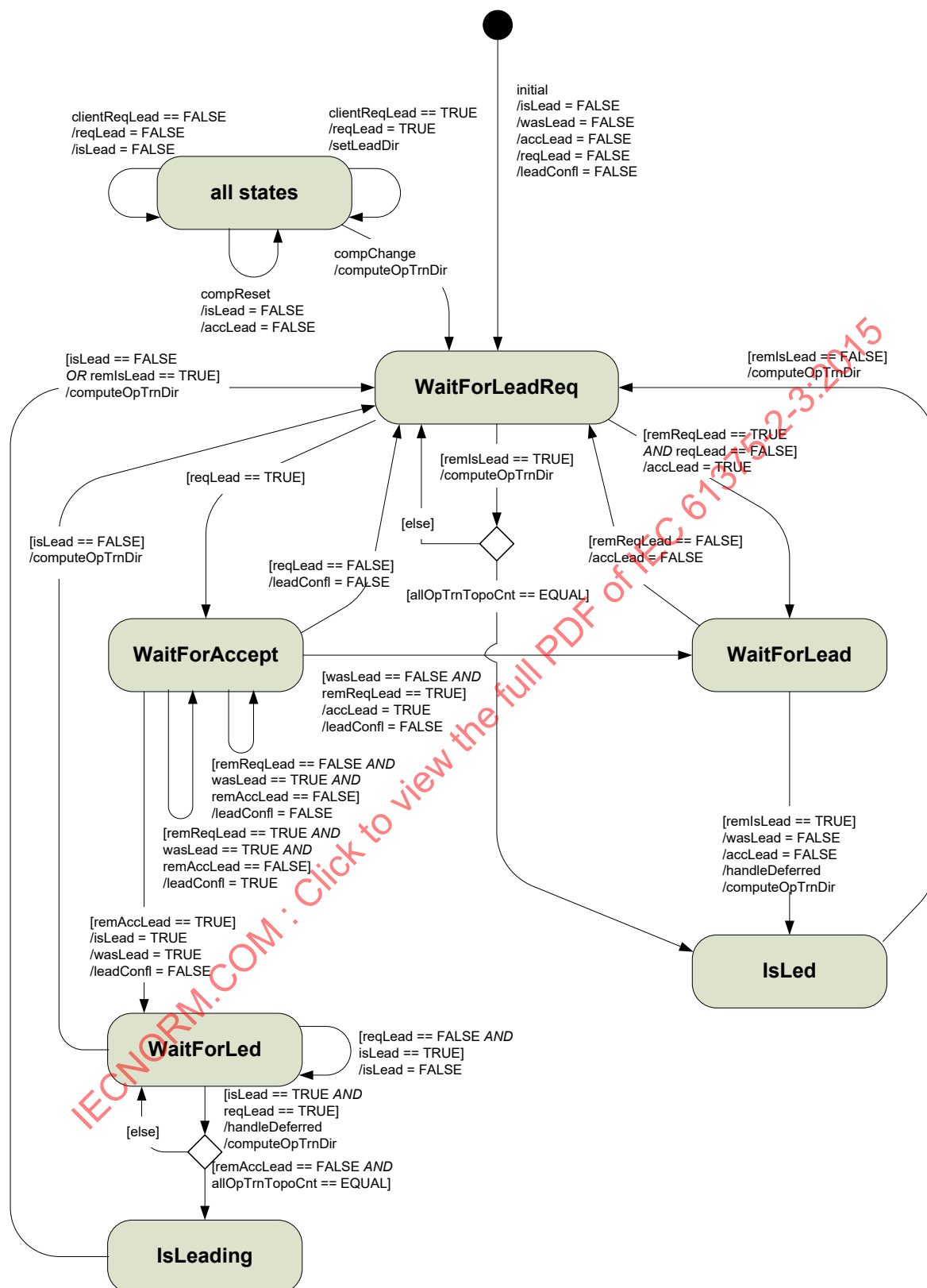
Figure 29 – Leading vehicle function state machine block diagram

All these flags are cyclically exchanged with the ETBCTRL telegram, but it has to be considered that, due to the asynchronous exchange of the ETBCTRL telegrams, the complete set of control flags might not be consistent at one time instance, leading to transient effects.

Locally, the leading vehicle function state machine is triggered by the ECSP control application, which forwards requests for leading to the state machine. The state machine, in return, forwards information about the actual status to the control application. One special information is the indication of a leading conflict which may occur if more than one consist request leading at the same time or if, e.g. after coupling, two leading vehicles exist.

6.5.3.2 State Diagram

The state diagram shown in Figure 30 defines the behavior of an ECSP with respect to the leading function. The related triggers, guards and actions are defined in Table 35 to Table 37.



IEC

Figure 30 – State diagram of leading function

NOTE 1 For transitions which are triggered by ETBCTRL telegram reception only the guards are shown to improve readability.

NOTE 2 The guard "remIsLead == TRUE" in state IsLeading should never happen, because if another train with property leading is coupled, then the trigger compChange will fire, because of the inauguration. But it is safer to handle this guard even though.

NOTE 3 A re-inserted ECSP in a composition with a leading vehicle will directly transit through state WaitForLead (guard remReqLead) to state IsLed (guard remIsLead).

NOTE 4 If a late inserted ECSP cannot integrate in a composition, this will lead to a composition change (trigger compChange).

Table 35 – Leading function – triggers

Trigger	Description
Initial	Power-up or re-boot of ECSP
clientReqLead	Local ECSC requests leadership (= TRUE) (service primitive F_leadingSetrequest) or requests to abandon leadership (= FALSE) (service primitive F_leadingResetRequest)
compChange	Train composition has changed. This event is triggered by the ETBCTRL processing state machine, see 6.4.5.
compReset	composition reset

Table 36 – Leading function – guards

Guards	Description
reqLead	Control flag in ETBCTRL indicating, if set, that own consist request leading property NOTE This control flag is set by trigger clientReqLead.
wasLead	Control flag in ETBCTRL indicating, if set, that the own consist previously owned leading property
isLead	Control flag in ETBCTRL indicating, if set, that the own consist is leading
remReqLead	At least one remote consist requests leading, meaning that in at least one received ETBCTRL telegram except the own ETBCTRL telegram the control flag reqLead ('rL') is set. NOTE If more than one consist requests leading this might end in a leading conflict
remAccLead	All remote consists accept own leading request. Logical AND over control flags accLead of all ECSPs except the own ECSP and except ECSPs marked as being timed out (see 6.4.5.2)
remIsLead	One remote consist is leading;. Control flag 'isLead' of one ECSP except the own ECSP is set to TRUE
allOpTrnTopoCnt	Comparing the parameter 'opTrnTopoCnt' in all (non-discarded and not timed out) ETBCTRL telegrams. EQUAL = identical value in all telegrams

Table 37 – Leading function – actions

Action	Description
computeOpTrnDir	Compute the operational train directory taking into account the correction information obtained with the ETBCTRL telegram and update ETBCTRL telegram parameter 'opTrnTopoCnt'..., see 6.7.3 for details
setLeadDir	Set ETBCTRL telegram parameter 'reqLeadDir' to the leading direction requested by the local ECSC.
leadConfl	Local flag which, if set, indicates to the application that a leading conflict has been detected.
handleDeferred	In case there are ECSPs marked as being timed out, reset (= set to FALSE) the deferred value of control flag 'wasLead' of the corresponding ECSP. NOTE This means that the deferred value of control flag 'wasLead' changes from 'unmodified' to 'FALSE'.

The parameter 'wasLead' is used to prevent a change of the operational direction after a composition change when this consist had the leadership before the composition change.

6.5.3.3 Preconditions

Before issuing an F_leadingSetRequest or F_leadingResetRequest to the ECSP, an ECSC should inhibit ETB train inauguration.

ETB train inauguration inhibit can be cancelled once the leading request transaction is finished.

6.5.3.4 Leading conflict

A leading conflict may occur if two trains are coupled both having a leading consist.

A leading conflict shall be indicated to the ECSC latest 1,0 s after detection.

NOTE 1 A leading conflict is indicated with the F_leadingStatusRequest function primitive.

NOTE 2 This part of IEC 61375 does not specify how a leading conflict can be resolved because this depends highly on the application model. One possibility is to indicate this to the driver so that the driver can deactivate one of the leading consists.

NOTE 3 In case of a leading conflict operation may be restricted until the leading conflict is resolved.

6.5.3.5 ECSC failure

In case of a ECSC failure ((see 6.3) the ECSP shall set telegram parameter 'reqLead' to value FALSE.

6.6 Function Confirmation/Correction

6.6.1 General

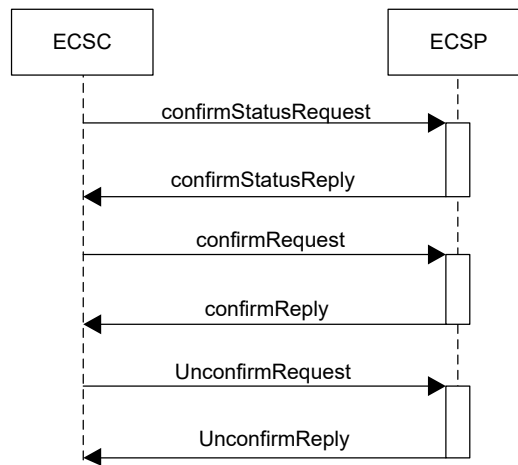
This function allows confirming the train composition as it has been discovered during train inauguration (see IEC 61375-2-5) or to correct the discovered train composition by adding/inserting 'gaps'. A gap is one or a sequence of vehicles which are physically present, but which have not been discovered during train inauguration (e.g. if the vehicles are powerless). A confirmation or correction is train-wide valid, meaning that this information will first be coordinated with all the other ECSPs before becoming valid. This also implies that all other ECSPs will add those confirmation and correction information to their local TTDB. A confirmed or corrected train composition will receive a unique signature (opTrnTopoCnt) which may be used in safety related functions to ensure that all train wide communication is based on the same view of the train composition.

NOTE A correction of the discovered train composition will only affect the TTDB operational train directory. The TTDB train directory and the train network directory as defined in IEC 61375-2-5 will not be affected.

6.6.2 Function primitives

Function primitives are provided by the ECSP and may be invoked by the local ECSC for controlling confirmation and correction.

The functions defined in (Table 38 to Table 40) and shown in Figure 31 shall be supported.



IEC

Figure 31 – Confirmation sequence diagram

Table 38 – Confirmation function primitives – F_confirmStatusRequest

Function primitive:	F_confirmStatusRequest
Description:	Read status of the train directory.
Request parameter:	-
Response parameter:	TTDB parameter values: - trnDirState as defined in 5.3.3.2.12 - opTrnDirState as defined in 5.3.3.2.12 - opTrnTopoCnt as defined in 5.3.3.2.12

Table 39 – Confirmation function primitives – F_confirmRequest

Function primitive:	F_confirmRequest
Description:	Correct and confirm a train composition. Correction information shall be entered in the ETBCTRL telegram (parameter ConfVehCnt and ConfVehList). If only confirmation is requested, no confirmed vehicle list shall be filled in (ETBCTRL parameter ConfVehCnt = 0)
Request parameter:	set of confirmation and correction rules (see 6.7.2.2)
Response parameter:	-

Table 40 – Confirmation function primitives – F_unconfirmRequest

Function primitive:	F_unconfirmRequest
Description:	Unconfirm a train composition. This includes also to remove any correction information from the ETBCTRL telegram.
Request parameter:	-
Response parameter:	-

6.6.3 ECSP to ECSP protocol

6.6.3.1 General

This subclause defines the protocol for service related data exchange between the ECSP peers.

The confirmation/correction function uses four parameters ('control flags') of the ETBCTRL telegram for control as listed in Table 41:

Table 41 – Confirmation function control flags

control flag / parameter	short form	description	deferred value
corrRequest	rqC	Confirmation/correction requested	FALSE
clearConfComp	clC	Clear confirmed composition	Don't care
corrInfoSet	set	Correction info set	FALSE
compStored	stC	Composition stored	Don't care
opTrnTopoCnt	-	Computed operational train topography counter value	Don't care
Don't care = this value shall not be used in computation.			

The state of the confirmation/correction function is determined by the set of flags received from the other consists (remote confirmation/correction control flags, sent by the consists listed in the train directory with an $cstTopoCnt \neq 0$) and the own set of flags (own confirmation/correction control flags) as it is shown in Figure 32.

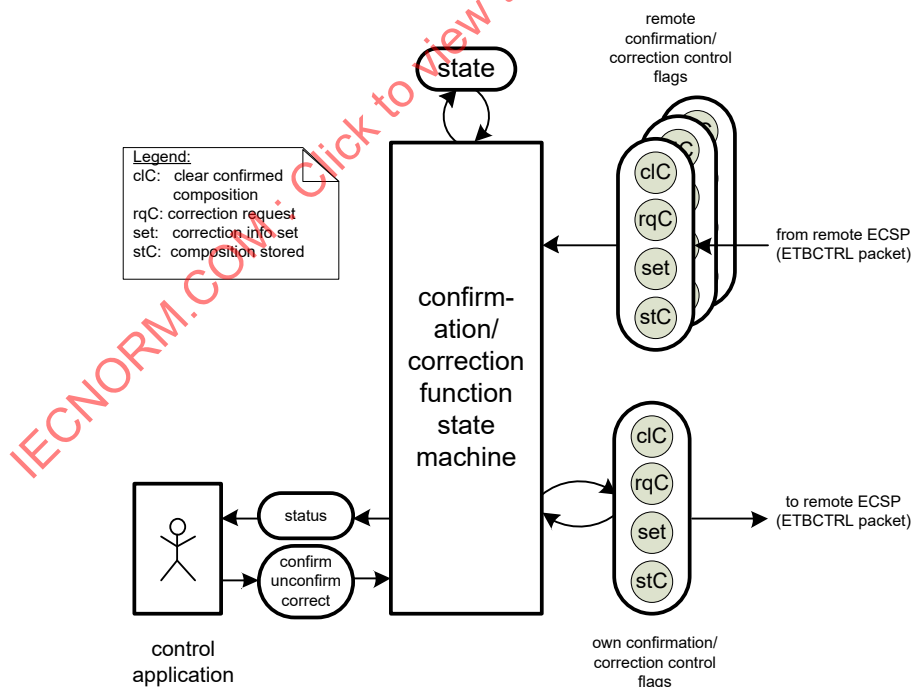


Figure 32 – Confirmation/correction function state machine block diagram

All these flags are cyclically exchanged with the ETBCTRL telegram, but it has to be considered that, due to the asynchronous exchange of the ETBCTRL telegrams, the complete set of control flags might not be consistent at one time instance, leading to transient effects.

Locally, the confirmation/correction function state machine is triggered by the ECSP control application, which forwards a request for confirmation, a request to abandon confirmation ('unconfirm') or a request for correction to the state machine. The state machine, in return, forwards information about the actual status to the control application.

6.6.3.2 Correction sequence

The interaction for correction between the ECSP of the different consists in the train is depicted in Figure 33 as an example of a train with two consists A and B.

Initially, the four control flags (stC,set,rqC,clC) are set to logical (0000)³.

Confirmation with or without correction is initiated by an ECSC (ECSC A in the example) by invoking the F_confirmRequest function primitive. ECSP A will then add the confirmation/correction information to the ETBCTRL telegram, which informs all the ECSPs (including ECSP A) about the required correction or confirmation (control flags set to (0010)).

Correction information comprises:

- The number of confirmed vehicles in the train (parameter 'confVehCnt' in ETBCTRL telegram)
- An ordered list of confirmed vehicles in the train (parameter 'ConfVehList' in ETBCTRL telegram)

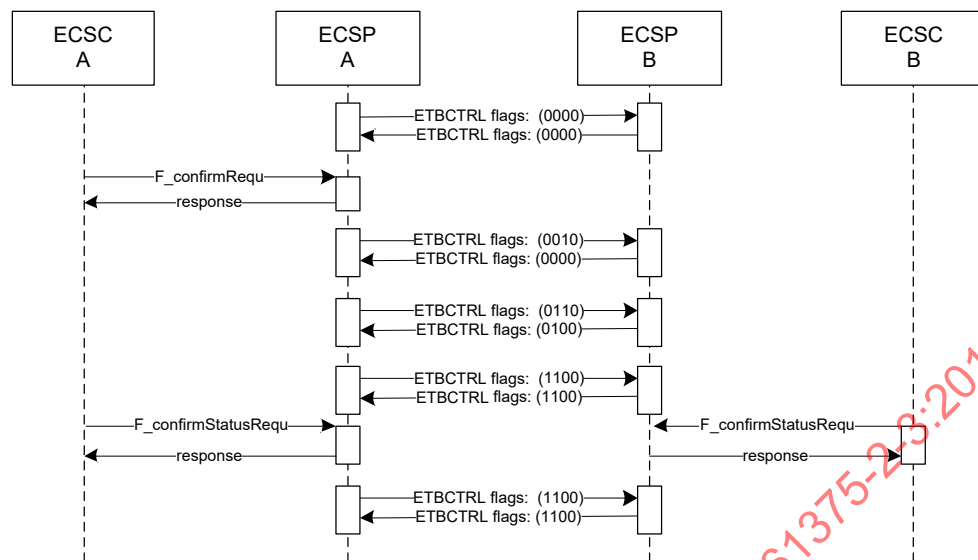
ECSP A will then compute a confirmed/corrected operational train directory and will thereafter set the control flags to (0110).

ECSP B will as well compute a confirmed/corrected operational train directory and will thereafter set the control flags to (0100).

If all ECSPs have computed a confirmed/corrected operational train directory, the ECSPs change to state 'CompStored', which is reflected in the control flags set to (1100).

The ECSC can ask for the confirmation status by invoking the F_confirmStatusRequest function primitive.

³ '0' stands for FALSE, '1' stands for TRUE. Here: (0000) stands for (FALSE,FALSE,FALSE,FALSE).



IEC

Figure 33 – Correction/confirmation protocol sequence chart (example)

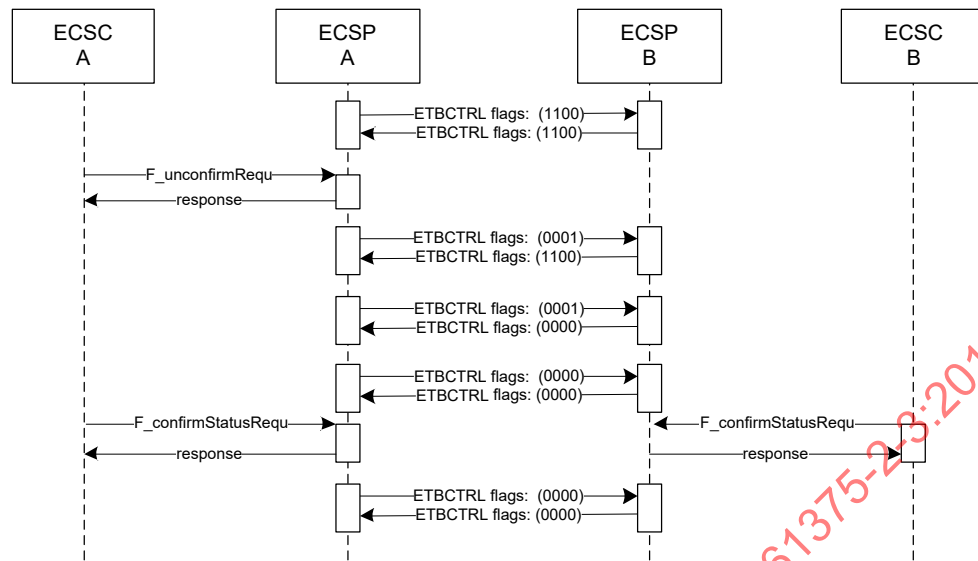
6.6.3.3 Sequence for confirmation of an existing composition

The sequence for confirmation of an existing composition is identical to the correction sequence. The only difference is that the list of confirmed vehicles is empty.

6.6.3.4 Sequence for confirmed composition cancellation

A confirmed composition can be cancelled by invoking the F_unconfirmRequest function primitive, which shall be responded by the ECSP with F_unconfirmReply. Upon reception of a cancellation request, the ECSP shall set the 'clC' control flag for three consecutive process data cycles, see Figure 34.

A ECSP receiving an ETBCTRL telegram with the 'clC' control flag set will acknowledge this by setting its control flags to (0000).

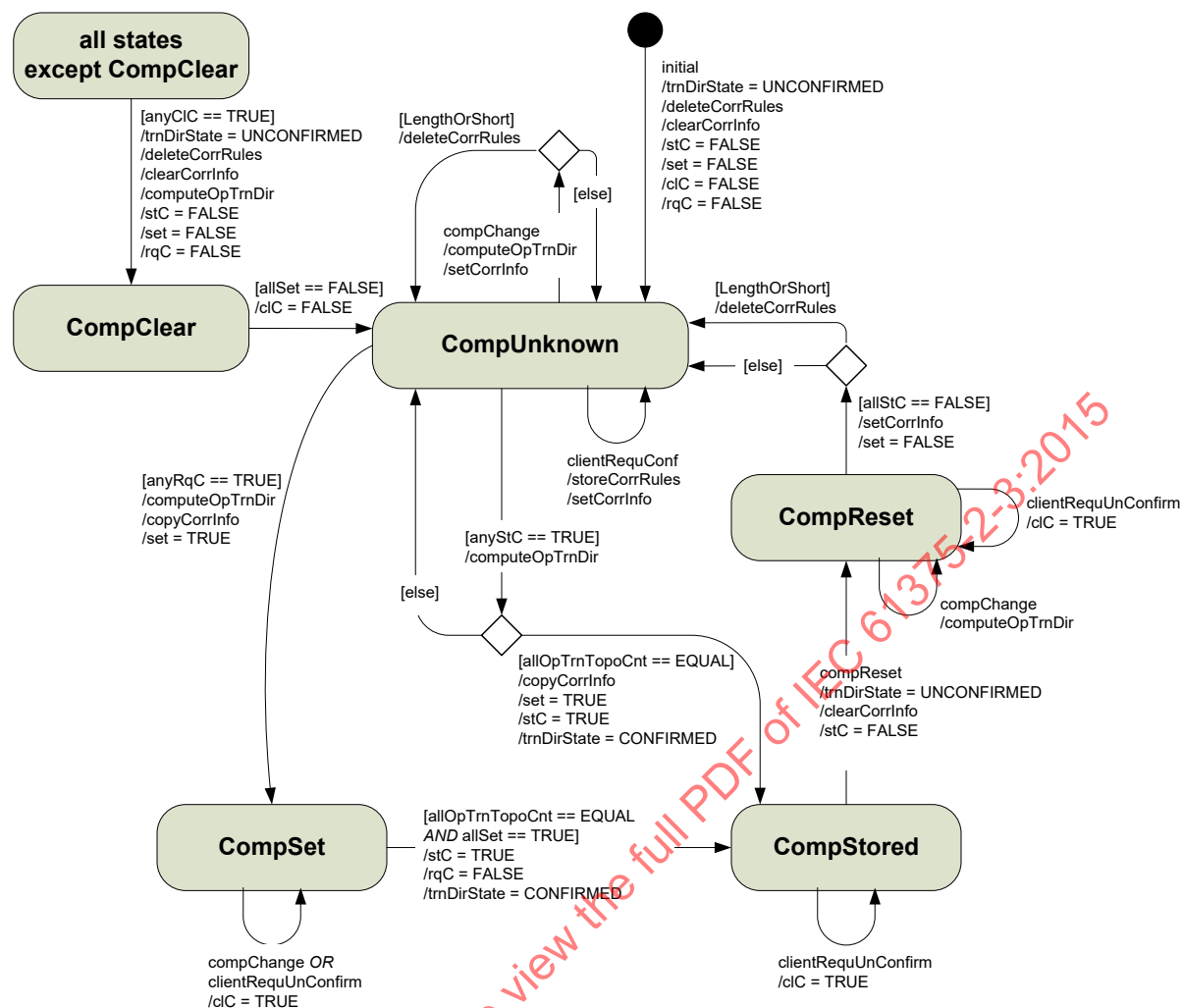


IEC

Figure 34 – Unconfirm protocol sequence chart (example)

6.6.4 State diagram

The confirmation/correction function is defined by the state diagram shown in Figure 35 with further explanations given in Table 42 to Table 44.



IEC

Figure 35 – Confirmation/correction state diagram

NOTE For transitions which are triggered by ETBCTRL telegram reception only the guards are shown to improve readability.

Table 42 – Confirmation/correction state diagram – Trigger

Trigger	Description
Initial	Power-up or re-boot of ECSP
clientRequConfirm	Confirmation/correction command received from ECSC
clientRequUnConfirm	Command received from ECSC to unconfirm (reset)
compChange	Train composition has changed. This event is triggered by the ETBCTRL processing state machine, see 6.4.5.
compReset	Confirmed composition reset

Table 43 – Confirmation/correction state diagram – Guard

Guard	Description
anyRqC == TRUE	Logical OR over the 'rqC' control flags in all ETBCTRL telegrams equals TRUE
anyCIC == TRUE	Logical OR over the 'clC' control flags in all ETBCTRL telegrams equals TRUE

Guard	Description
allStC == FALSE	Logical OR over the 'stC' control flags in all ETBCTRL telegrams equals FALSE
allSet == FALSE	Logical OR over the 'set' control flags in all ETBCTRL telegrams equals FALSE for a time of at least 3 ETBCTRL cycles. NOTE This time condition shall ensure that also reinserted ECSP properly enter state CompUnknown.
anyStC	Logical OR over the 'stC' control flags in all ETBCTRL telegrams
allOpTrnTopoCnt	Comparing the parameter 'opTrnTopoCnt' in all ETBCTRL telegrams EQUAL = identical value in all telegrams
LenghtOrShort	The discovered end vehicles of the new composition are not identical to the discovered end vehicles of the last confirmed composition (= train lengthening or shortening)

Table 44 – Confirmation/correction state diagram – Action

Action	Description
computeOpTrnDir	Compute the operational train directory taking into account the correction information obtained with the ETBCTRL telegram and update ETBCTRL telegram parameter 'opTrnTopoCnt'..., see 6.7.3 for details
trnDirState	Set the TTDB parameter 'trnDirState' to defined value.
storeCorrRules	Store the correction rules obtained from user locally
deleteCorrRules	Delete the locally stored correction rules
setCorrInfo	Check consistency of the locally stored correction rules with the operational train directory and add correction information to the ETBCTRL telegram as defined in 6.7.2.
copyCorrInfo	This action is optional. Copy the confirmed vehicle list (ETBCTRL parameters 'confVehList' and 'confVehCnt') from the ECSP distributing it to the own ETBCTRL telegram (only if this ECSP is not already distributing the confirmed vehicle list). ETBCTRL parameter 'rqC' shall not be changed. NOTE By copying this, the correction information is not lost when the ECSP which originally distributed the informations is out of order.
clearCorrInfo	Remove correction information from the ETBCTRL telegram (remove confirmed vehicle list) and set ETBCTRL parameter 'confVehCnt' to 0.

6.6.5 ECSC Failure

In case of a ECSC failure (see 6.3) no special action is requested for the ECSP.

6.7 Computation of the operational train directory

6.7.1 General

The computation of the operational train directory is supported by two actions:

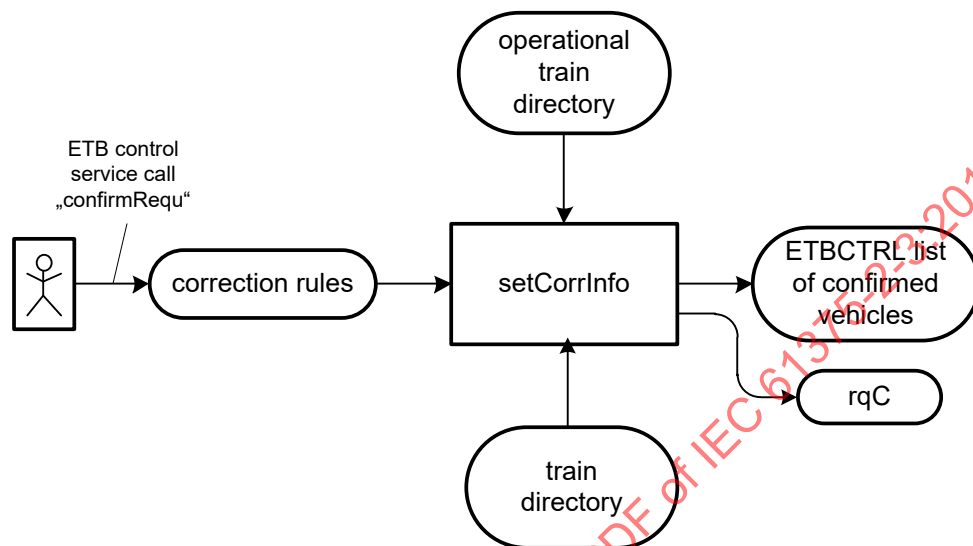
- The action “setCorrInfo”
- The action “computeOpTrnDir”

Those two actions are controlled by the leading function (computeOpTrnDir only) and the confirmation/correction function.

6.7.2 Action setCorrInfo

6.7.2.1 General

The purpose of this action is to transfer the correction rules given by the user to a list of confirmed vehicles (confVehList) as it will be appended to the ETBCTRL telegram. This process is shown in Figure 36.



IEC

Figure 36 – Action “setCorrInfo” block diagram

For transfer, information from the train directory (e.g. TTDB parameter ‘vehId’) and from the operational train directory (e.g. TTDB parameter ‘trnVehNo’) is needed.

This transfer includes a consistency check, which checks whether the rules can be correctly applied to the given train directory following the correction policy defined in 6.7.2.3.

6.7.2.2 Confirmation/Correction rules

Confirmation/Correction rules are rules which describe which confirmation or correction actions shall be performed on a given train configuration as defined by the train directory.

The rules defined in Table 45 shall be supported.

Table 45 – Confirmation/Correction rules

Rule	Description
Confirm the composition without correction	The train composition as discovered during train inauguration and represented by the train directory is confirmed (no correction necessary)
Insert N vehicles between vehicle with VehId X and vehicle with VehId Y	Correct the discovered train composition by inserting a gap with N vehicles between discovered vehicles X and Y.
Insert N vehicles in front of extremity 1 of the vehicle with VehId X	Correct the discovered train composition by inserting a gap with N vehicles in front of the vehicle.
Insert N vehicles after extremity 2 of the vehicle with VehId X	Correct the discovered train composition by inserting a gap with N vehicles after the vehicle.
Confirm the composition with correction	The train composition including the vehicle gaps shall be confirmed.
Unconfirm the composition	Discard all corrections made and unconfirm the composition

NOTE It is not within the scope of this standard to define a specific coding of those rules. A possible implementation is given in Annex E.

6.7.2.3 Correction Policy

The correction policy defines the conditions under which a correction is accepted and under which a correction is kept following a train inauguration.

- a) A correction, meaning the insertion of a gap with N vehicles into a discovered train composition, shall be accepted if the following conditions are fulfilled:
 - 1) The gap is not inserted within the boundaries of a consist (only insertion between consists is allowed).
 - 2) The total number of 63 vehicles is not exceeded.

In all other cases shall the user correction request be refused.

- b) If after a correction a new train inauguration occurs, the correction shall only persist and the confirmation shall be maintained if the following conditions are fulfilled:
 - 1) If at the position of a gap discovered vehicles (not present after last train inauguration) appear: the number N of gap vehicles shall equal the number of newly discovered vehicles.
 - 2) The discovered end vehicles of the new composition are identical to the discovered end vehicles of the old composition (no lengthening or shortening).

In all other cases shall the composition become unconfirmed.

- c) If after a new train inauguration previously existing middle vehicles are not anymore discovered, but the discovered end vehicles are identical to the discovered end vehicles before (no train lengthening or shortening), a gap shall be automatically inserted replacing the missing middle vehicles. This however shall only be done if the gap is not inserted within the boundaries of a consist (only insertion between consists is allowed).

6.7.2.4 Function execution

The function setCorrInfo shall behave as follows:

If there is only a rule for composition confirmation without correction, and there was no automatic gap insertion (see 6.7.2.3 item 3), then no confirmed vehicle list (confVehList) shall be appended to the ETBCTRL telegram (ETBCTRL parameter 'confVehCnt' = 0) and the ETBCTRL parameter 'rqC' shall be set to TRUE.

If there is only a rule for composition confirmation without correction, and there was an automatic gap insertion, then the confirmed vehicle list (confVehList) shall be appended to the ETBCTRL telegram (ETBCTRL parameter 'confVehCnt' != 0) and the ETBCTRL parameter 'rqC' shall be set to TRUE.

If there are neither correction rules nor a rule to confirm the composition stored, then no confirmed vehicle list (confVehList) shall be appended to the ETBCTRL telegram (ETBCTRL parameter 'confVehCnt' = 0) and ETBCTRL parameter 'rqC' shall be set to FALSE.

If the correction rules are inconsistent with the correction policy, then no confirmed vehicle list (confVehList) shall be appended to the ETBCTRL telegram (ETBCTRL parameter 'confVehCnt' = 0) and ETBCTRL parameter 'rqC' shall be set to FALSE.

If there are correction rules stored AND the correction rules are consistent with the correction policy, then the confirmed vehicle list (confVehList) shall be appended to the ETBCTRL telegram (ETBCTRL parameter 'confVehCnt' != 0) and the ETBCTRL parameter 'rqC' shall be set to TRUE.

NOTE 1 The consistency check presumes that after the successful execution of a confirmation/correction, the confirmed train composition equals the real train composition and that a change of the real train composition is only possible by decoupling or coupling of consists (e.g. the coupling of two trains). This means that if a train composition was wrongly confirmed or corrected (e.g. a non-operable vehicle in the train has been forgotten during confirmation/correction), the operational train directory may also show false results.

NOTE 2 During correction, train composition gaps are announced which indicate locations of unreachable vehicles. At that time it often is only known how many vehicles are in the gap (because they are visible to the driver for example), but not how those vehicles are associated to consists. For example, a gap with 3 vehicles might be filled with a 3-vehicle consist, a 1-vehicle and a 2-vehicle consist or three 1-vehicle consist.

NOTE 3 There exists a related action 'copyCorrInfo' which copies correction information obtained from another ETBCTRL telegram to the own ETBCTRL telegram, see 6.6.4. In this case ETBCTRL parameter 'rqC' is set to FALSE, but ETBCTRL parameter 'confVehCnt' is set to a value != 0. The combination of these two parameters indicates the nature of the confirmation/correction as can be seen from the following truth table:

rqC	confVehCnt	description
FALSE	= 0	Composition unconfirmed
FALSE	!= 0	Correction info copied
TRUE	= 0	Confirmation requested
TRUE	!= 0	Correction requested

EXAMPLES

Some typical examples of consistency check results are shown in Figure 37.

These examples base on a corrected train composition as shown in the beginning of Figure 37. Each vehicle is also a consist. The list of confirmed vehicles (confVehList) which is appended to the ETBCTRL telegram will look like this:

confVehCnt = 9

trnVehNo	isLead	leadDir	vehOrient
0	1	0	0
1	1	0	2
0	1	0	0
2	1	0	1
0	1	0	0
0	1	0	0
3	1	0	1
4	2	1	1
0	1	0	0

Case 1 exactly fits to the confirmed composition, in cases 2 and 4 the number of new vehicles fits to the number of vehicle gaps at that location, so this is consistent. In case three, it is not clear to which location the new vehicle shall be mapped to, it can be E or F, so this is considered inconsistent.

Case 5a depicts a case where a previously present vehicle disappeared after inauguration (creating a gap of three vehicles at that location: two corrected vehicle gaps plus one missing vehicle) and, after a next train inauguration, two new vehicles with unknown vehId are reported at these locations (case 5b). As the gap is not fully filled this is considered inconsistent

Cases 6 and 8 show a gap filling at the train's end, which is inconsistent. It cannot be determined whether the vehicle representing the gap became active or a new vehicle has been appended, in that case increasing the number of vehicles to a total of 10.

Case 7 shows a real train lengthening, but also here it cannot be determined whether one of the new vehicles fills the vehicle gap and only one vehicle has been appended, or the vehicle gap is still present and two vehicles have been appended.



IEC

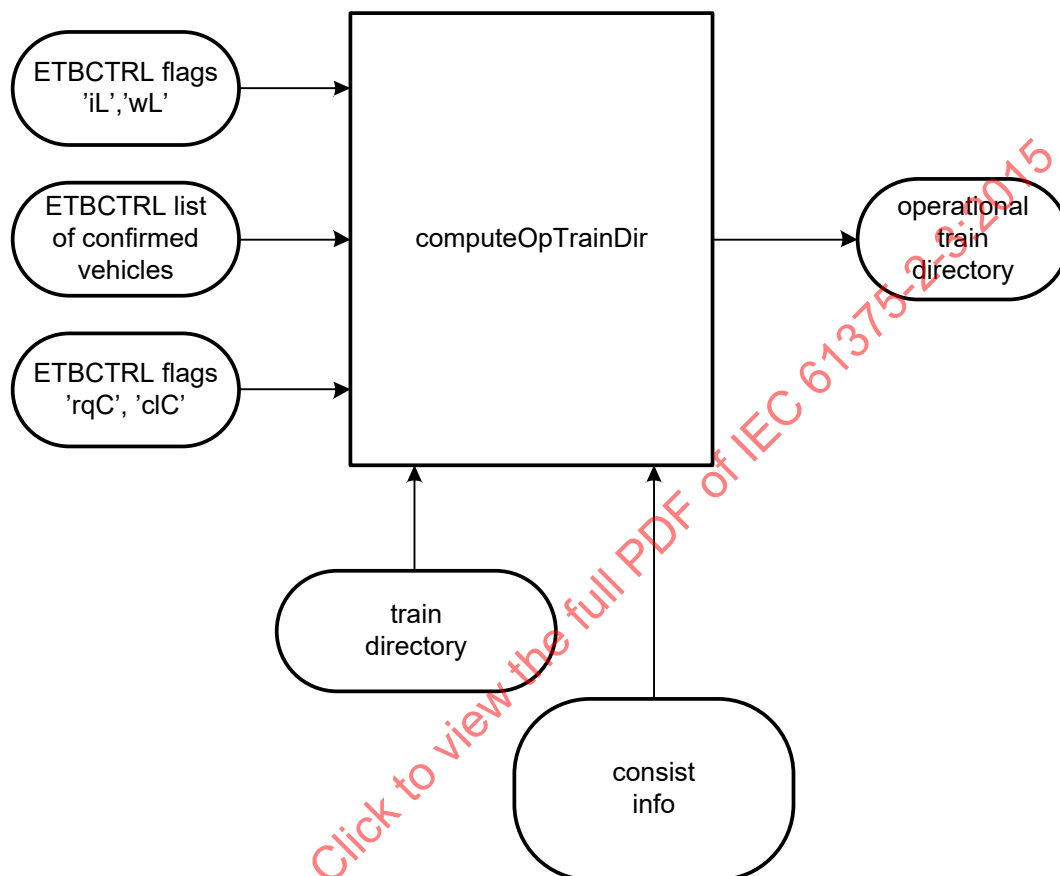
Figure 37 – Train composition consistency check examples

6.7.3 Action computeOpTrnDir

6.7.3.1 General

The action computeOpTrnDir has the task to generate the operational train directory by reading and interpreting information from four input data sources (see Figure 38):

- correction information, taken from the ETBCTRL telegram and provided by the confirmation/correction function , especially the action 'setCorrInfo'
- the information on a leading vehicle within the train, taken from the ETBCTRL telegram and provided by the leading vehicle function (see 6.5)
- the actual sequence of consists in the train, taken from the train directory
- the number and sequence of vehicles within a consist, taken from the consist info list



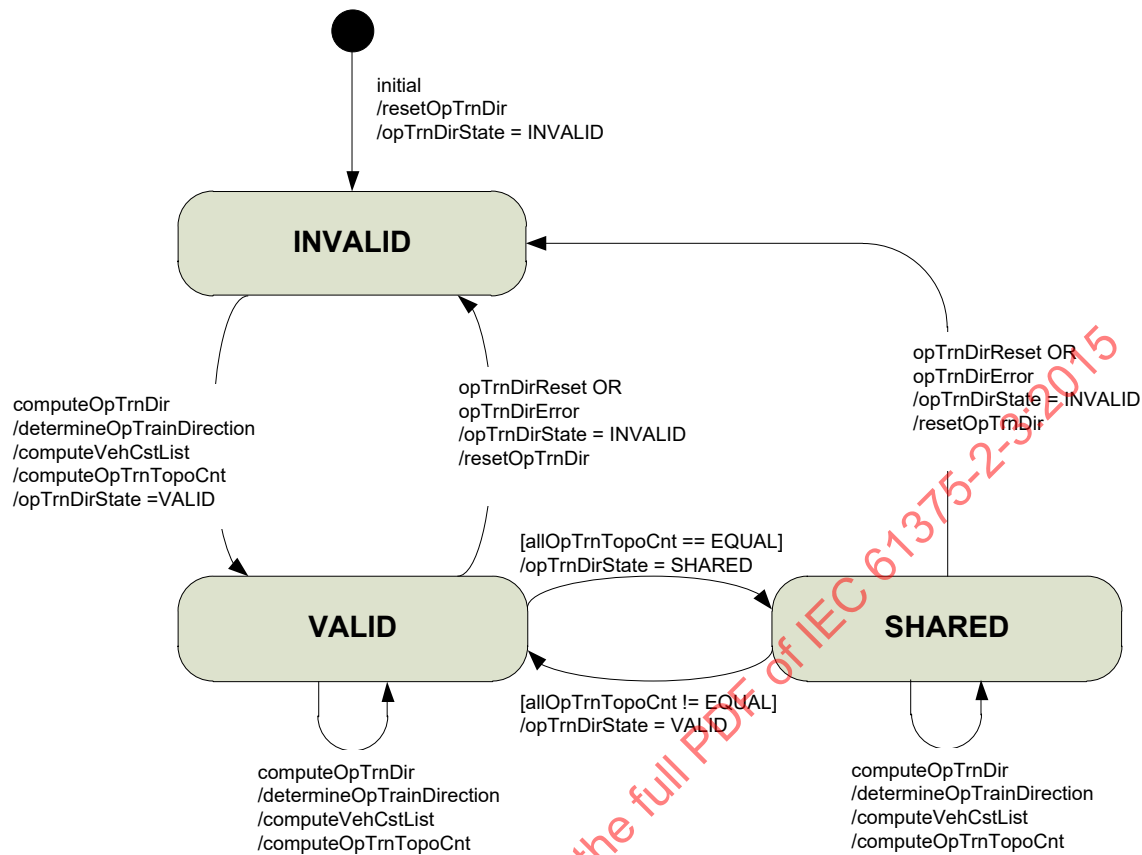
IEC

Figure 38 – Computation of the operational train directory

6.7.3.2 Operational train directory computation state diagram

The state diagram shown in Figure 39 and further explained in Table 46 to

Table 48 defines the computation of the operational train directory.



IEC

Figure 39 – computeOpTrnDir state chart

Table 46 – Operation Train Directory computation state diagram – Trigger

Trigger	Description
initial	Power-up or reboot of ECSP
computeOpTrnDir	A new computation of the operational train directory has been requested
opTrnDirReset	A reset of the operational train directory has been requested
opTrnDirError	An error has occurred preventing the computation of a valid operational train directory

Table 47 – Operation Train Directory computation state diagram – Guards

Guard	Description
allOpTrnTopoCnt == EQUAL	ETBCTRL telegrams have been received from all active ECSP listed in the local train directory with cstTopoCnt != 0 and all received valid ETBCTRL telegrams (including own ETBCTRL telegram) have an identical value of parameter 'opTrnTopoCnt'

Table 48 – Operation Train Directory computation state diagram – Action

Action	Description
determineOpTrainDirection	The operational train direction shall be determined as specified in 4.2.4.3.
computeVehCstList	An ordered list of consists (TTDB parameter 'opCstList') and vehicles (TTDB parameter 'opVehList') shall be generated using information from the train directory and the consist info list. If parameter 'confVehCnt' value != 0 then vehicle gaps and missing vehicles, provided with correction information in the ETBCTRL telegram, shall be considered as defined in 6.7.2. As a rule, a vehicle gap shall be counted as a consist containing one vehicle.
computeOpTrnTopoCnt	The operational train topography counter value shall be calculated as defined in 5.3.3.2.13.
resetOpTrnDir	The operational train directory and the operational train directory status shall be invalidated by setting the following TTDB parameter values: opTrnTopoCnt = 0 opTrnOrient = unknown
opTrnDirState	Set TTDB parameter 'opTrnDirState' to defined value

NOTE In state 'SHARED' all ECSPs are synchronized using the same OpTrnTopoCnt value. EDs can profit because EDs can send TRDP telegrams train wide without risking that the telegram will be discarded by the receiver due to a mismatch in the opTrnTopoCnt value.

EXAMPLE

With the corrected train composition obtained from Figure 37 the operational train directory vehicle list will look as shown in Table 49.

Table 49 – Example of operational train directory

vehId	opVehNo	isLead	leadDir	trnVehNo	VehOrient	ownOpCstNo
-	1	1	0	0	0	1
'H'	2	2	1	4	2	2
'G'	3	1	0	3	2	3
-	4	1	0	0	0	4
-	5	1	0	0	0	5
'D'	6	1	0	2	2	6
-	7	1	0	0	0	7
'B'	8	1	0	1	1	8
-	9	1	0	0	0	9

6.8 Function Sleep Mode (Option)

6.8.1 General

This function allows to set a complete train to a low power mode (= sleep mode) and to return to regular operation on demand. Sleep mode will be entered if there is a request from all consists, and will be left if there are demands from at least one consist.

6.8.2 Sleep Mode Use Case (informal)

The following use case for the function sleep mode is taken from the concepts defined in UIC Leaflet 556.

The ETBNs in an individual vehicle should be able to take up the conditions listed in Table 50.

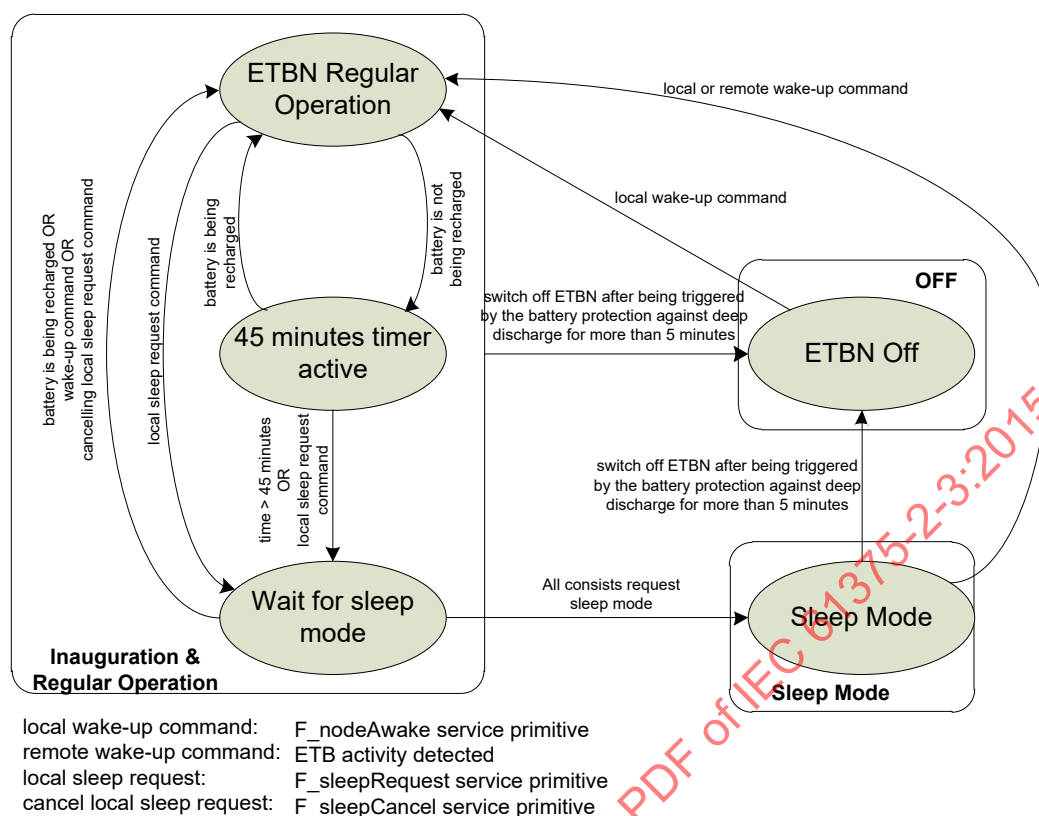
Table 50 – ETBN operating conditions

No	Operating condition	Description
0	OFF	An ETBN is not able to work when the voltage of the DC system of the vehicle concerned falls below 70 % of the nominal voltage.
1	SLEEP MODE	The ETBN is in a low power mode, but can receive signals on the train backbone cable, e.g. Ethernet frames.
2	INAUGURATION	The ETBN determines the train directory
3	REGULAR OPERATION	Information exchange over ETB

The following requirements determine the transitions of the operating and internal conditions of the ETBN:

- With battery charge available, the ETBN goes into the inauguration and regular operation.
- If the ETBN is in the sleep mode (operating condition 1), a dedicated action initiates the transition into the inauguration and regular operation (operating conditions 2 and 3). For example, UIC Leaflet 550-1 defines the action as pushing button "coach lighting on" on the operating switch desk.
- If ETB activity is recognized during sleep mode, the ETBN goes into the inauguration and regular operation (operating conditions 2 and 3).
- 45 min after the termination of battery charging throughout the train (e.g. switching off of the battery charging train line from the traction unit), the ETBN goes into the sleep mode (operating condition 1) corresponding to the sleep mode concept specified for the ETBN in this subclause.
- If the protection of the battery against deep discharge is activated (see also UIC Leaflet 550), the ETBN is switched off (operating condition 0), since all remote-controlled loads are likewise switched off.
- For emergency operation the ETBN can also be switched on below the lowest switch off stage of the minimal voltage, e.g. with the push button "coach lighting on" on the operating desk, provided that the minimum voltage of the DC system (70 % of the nominal voltage, IEC 60571) is available.
- The ETBCTRL telegram control bit "sleepRequest" (request to change to sleep mode) controls the transition between node operation and waiting in sleep mode.

The corresponding operational states of the ETBN are shown in Figure 40.



IEC

Figure 40 – Use case “sleep mode” state diagram

The handling of state ‘ETBN Off’ is a consist local matter not affecting interoperability and therefore not further specified in this part of IEC 61375.

Albeit the sleep mode use case described above is related to one ETBN, this part of the standard defines a more generic concept, which shares the sleep mode functionality between ECSP and ETBN:

- ECSP coordinates sleep mode requests between the different consists, handles the timing and provides the control interface to the ECSC
- ETBN supports by stopping transmission, by activating a line sensing function in sleep mode which detects line activity and by awaking the consist in case line activity has been detected

6.8.3 Exclusivity

All consists in a train have to support the sleep mode function in order to make the sleep mode function work. A mixture of sleep-mode aware and sleep-mode unaware consists is not supported.

6.8.4 Function primitives

Function primitives are provided by the ECSP and may be invoked by the local ECSC for controlling sleep mode.

The function primitives defined in this subclause (Table 51 to Table 54) and shown in Figure 41 shall be supported.

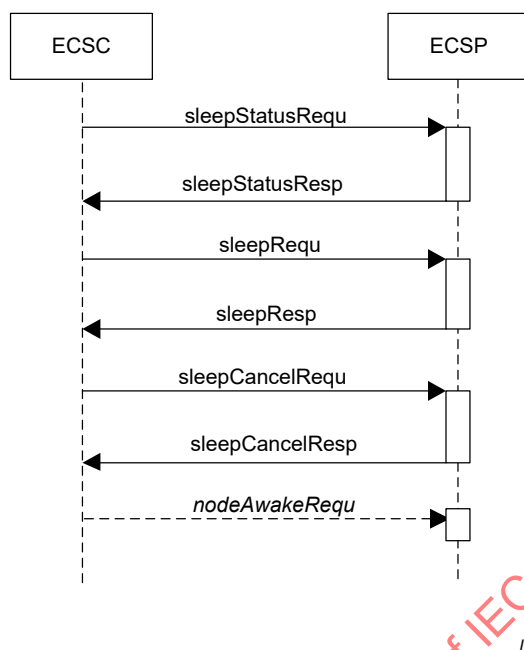


Figure 41 – Sleep control sequence diagram

Table 51 – Sleep mode function primitives – F_sleepStatus

Function primitive:	F_sleepStatus
Description:	Read status of sleep requests
Request parameter:	-
Response parameter:	number of consists requesting sleep own sleep control state

Table 52 – Sleep mode function primitives – F_sleepRequest

Function primitive:	F_sleepRequest
Description:	(Local) request to enter sleep mode (set sleepRequest bit in ETBCTRL-VDP)
Request parameter:	-
Response parameter:	sleepRequest = SET/UNSET

Table 53 – Sleep mode function primitives – F_sleepCancel

Function primitive:	F_sleepCancel
Description:	(Local) cancel sleep mode request (reset sleepRequest bit in ETBCTRL-VDP)
Request parameter:	-
Response parameter:	sleepRequest = SET/UNSET

Table 54 – Sleep mode function primitives – F_nodeAwake

Function primitive:	F_nodeAwake
Description:	Local request to return to regular operation (awake train). This function can be triggered by a local action, as for instance pushing a button "coach lighting on" on the operating switch desk or remotely via wireless ground to train communication. Dependant on the sleep mode implementation this command can be directed to the ECSP or to one or all of the ETBN.
Request parameter:	-
Response parameter:	-

6.8.5 ECSP to ECSP protocol

6.8.5.1 General

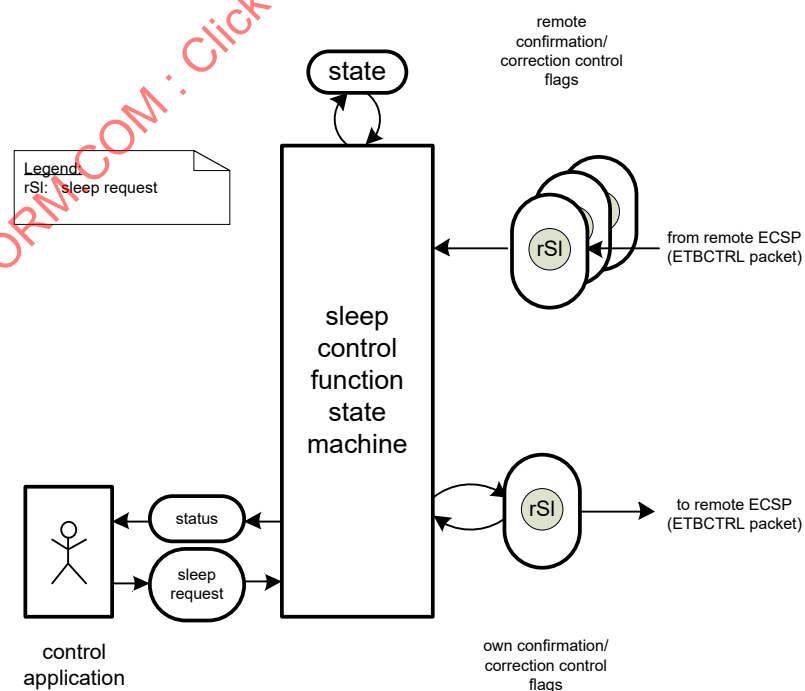
This subclause defines the protocol for service related data exchange between the ECSP peers.

The sleep control function uses one parameter ('flag') of the ETBCTRL telegram for control as listed in Table 55:

Table 55 – Sleep control function control flags

control flag	short form	description	default value
sleepRequest'	rSI'	Sleep request	FALSE

The state of the sleep control function is determined by the flags received from the other consists (remote sleep control flags, sent by the consists listed in the train directory with an cstTopoCnt != 0) and the own flag (own sleep control flag) as it is shown in Figure 42.



IEC

Figure 42 – Sleep control function state machine block diagram

6.8.5.2 Sleep sequence

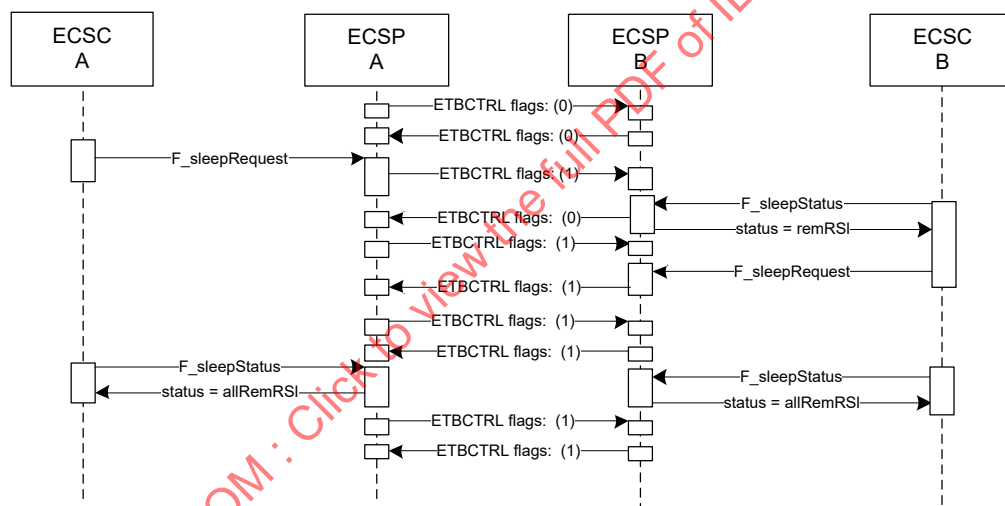
The interaction for sleep control between the ECSPs of the different consists in the train is depicted in Figure 43 as an example of a train with two consists A and B.

Initially, the control flag for sleep control (rSI) is set to logical (0)⁴.

Sleep request is initiated by an ECSC (ECSC A in the example) by invoking the F_sleepRequest function primitive. The local ECSP will then set its 'rSI' control flag, signaling that it is prepared for sleep mode, which informs all the ECSPs (including own ECSP) about the request. The remote ECSC will get this information with its status poll (function primitive F_sleepStatus) and then decides whether to prepare for sleep or not (this is an application decision). If prepared for sleep, the remote ECSC has to demand sleep mode from its local ECSP, which then will set the 'rSI' control flag accordingly. If all ECSPs have the 'rSI' control flag set, sleep mode can be entered.

NOTE 1 This part of IEC 61375 does not define how the ECSP drives a consist to a low power mode (sleep mode). This is task of the consist design.

NOTE 2 A possible sleep control interface between ECSP and ETBN is defined in Annex E.



IEC

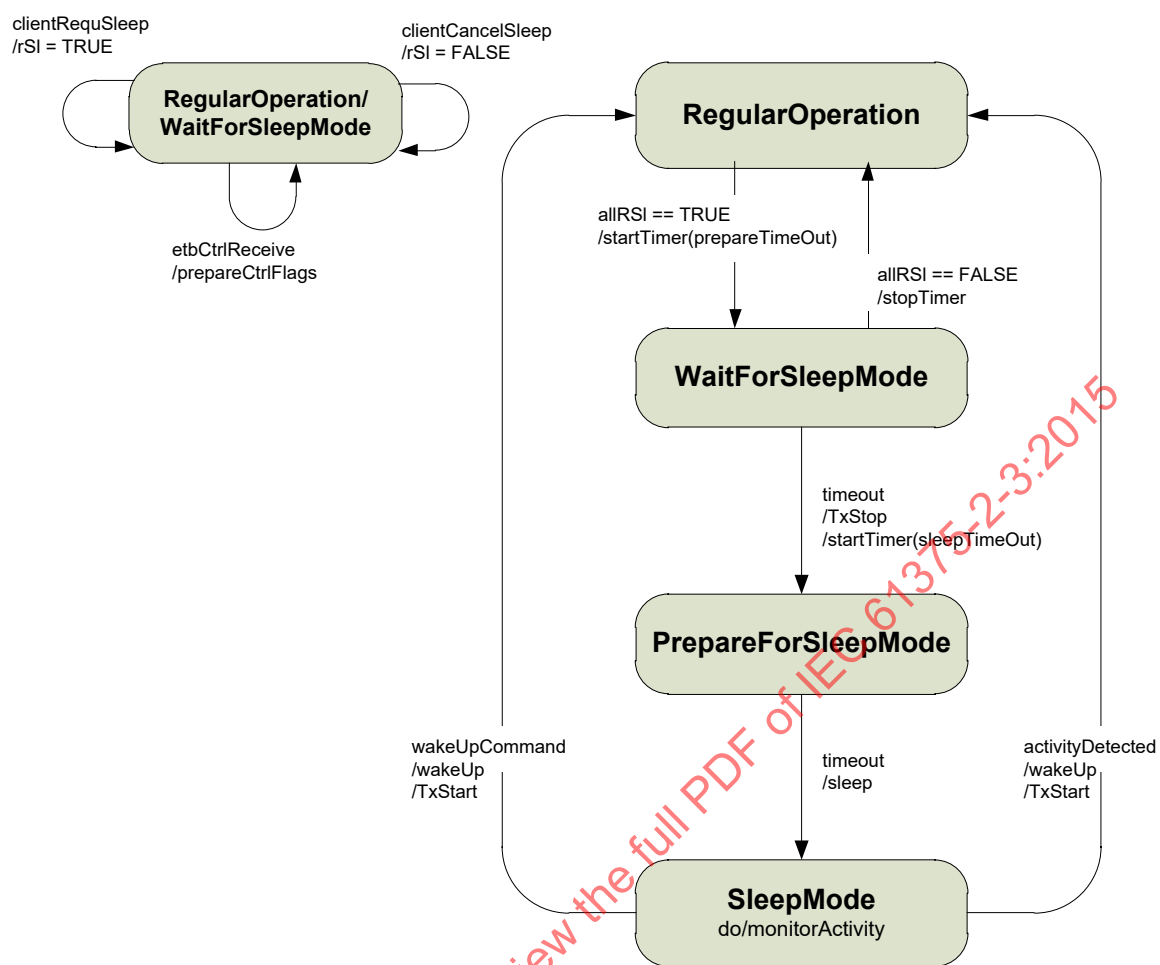
Figure 43 – Sleep request protocol sequence chart (example)

6.8.5.3 State diagram

The ECSP state diagram for sleep control is shown in Figure 44 and Tables 56 to 58.

NOTE An ECSP, which detects all 'sleepRequest' bits set, continues to send ETBCTRL telegram for a while in order to ensure that all other ETBNs have also detected this (state WaitForSleepMode). After this, the ETBN stops sending by changing to active bypass mode (state PrepareForSleepMode). Here it also stays for a while in order to ensure that all other ETBNs have also reached this state. Thereafter it passes to sleep.

⁴ (0) stands for (FALSE).



IEC

Figure 44 – Sleep control state diagram

Table 56 – Sleep control state diagram – trigger

Trigger	Description
clientRequSleep	Local sleep request command received
clientCancelSleep	Local sleep request cancel command received
activityDetected	Activity on ETB detected (HELLO frames)
wakeUpCommand	Local command to wake up
etbCtrlReceive	New ETBCTRL telegram received
timeout	Timeout

Table 57 – Sleep control state diagram – guards

Guards	Description
allRSI	Logical AND over rSI control flags in all ETBCTRL telegrams

Table 58 – Sleep control state diagram – action

Action	Description
prepareCtrlFlags	Retrieve rSI control flag from received ETBCTRL telegrams and store locally
startTimer	Start a timer with the timeout values: prepareTimeOut = (5,0 ± 1,0) s sleepTimeOut = (10,0 ± 1,0) s
stopTimer	Stop the timer
TxStop	ETBN shall stop sending on ETB, but ETBN in intermediate node setting shall still forward Ethernet frames between direction 1 and direction 2. ETBN shall be able to receive Ethernet frames for detecting line activity. NOTE One implementation possibility is to switch the ETBN in bypass
Sleep	Switch the consist to a low power mode (consist design dependant)
TxStart	ETBN resumes (enables) sending on ETB
monitorActivity	ETBN shall sense activity on ETB (reception of HELLO frames). Upon reception, the ETBN shall trigger 'activityDetected'
wakeup	Return from low power mode ETBN start train inauguration (sending HELLO and TOPOGRAPHY frames)

6.8.5.4 ECSC Failure

In case of a ECSC failure (see 6.3) no special action is requested for the ECSP.

Annex A (normative)

Train Real-Time Data Protocol (TRDP)

A.1 General

This annex defines the train real-time data protocol (TRDP) which shall be used for the exchange of TCN process data and TCN message data over ETB. The TRDP is executed by a TRDP layer which is placed on top of the TCP/UDP transport layer (Figure A.1). This layer may optionally be extended by a safety layer which provides safe end-to-end data transmission of safety critical process data and message data between any two end devices in the network.

All TRDP data on the wire is defined as big endian and byte alignment. Possible marshalling of user data needs to be done by a specific service primitive called from application layer before calling the TRDP service primitives to send data. As well a possible unmarshalling of user data needs to be done at application level after receiving data from the TRDP layer.

NOTE Also the optional safety layer uses the data in wire format (big endian and byte alignment).

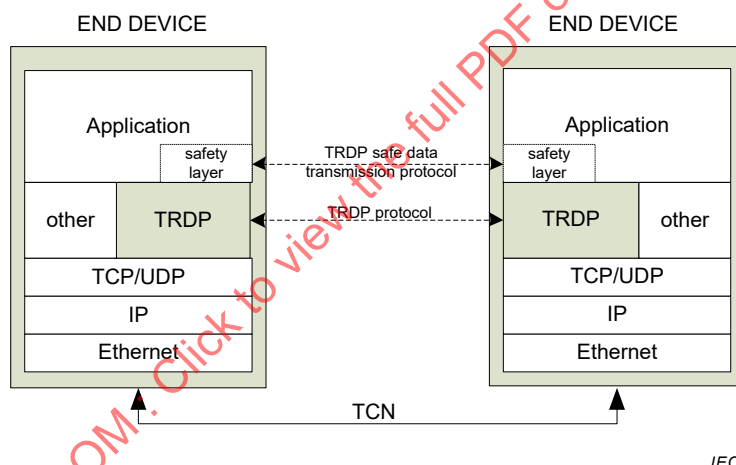


Figure A.1 – Overview of the protocol stack

A.2 Lower Layers

A.2.1 Data link layer

TRDP uses the ETB as defined in IEC 61375-2-5 for data communication within trains.

A.2.2 Network Layer

TRDP is based on IP (RFC 791) for data exchange between functions located in different consists of a train.

IP Addressing of functions and devices is defined in IEC 61375-2-5 and in this part of IEC 61375. IP address assignments may change after train inaugurations.

Data transfer between ETB and CN is provided by a gateway functionality of the ETBN as defined in IEC 61375-2-5. As a recommendation, this gateway should only transfer data between ETB and CN if the data are consistent with the actual train backbone view (defined by

the 'etbTopoCnt' value) and (optionally) operational train view (defined by the 'opTrnTopoCnt' value). This consistency can be checked by comparing the values of 'etbTopoCnt' and 'opTrnTopoCnt' contained in TRDP telegrams to the values stored in the ETBN. In case of a discrepancy the telegrams should be discarded.

NOTE 1 The filtering of inconsistent data within the ETBN reduces the probability that inconsistent data are transferred to the destination. But it does not release the destination from the responsibility to check the consistency by itself.

NOTE 2 A possible implementation will be to define corresponding firewall rules for the IP router within the ETBN (if IP router is available)

A.2.3 Transport Layer

TRDP uses the services of the UDP (RFC 768) and TCP (RFC 793) transport layer protocols for data communication.

TRDP process data shall be sent with UDP. TRDP process data packet size shall be restricted to the size of one Ethernet frame.

TRDP message data may be sent with UDP or with TCP. The choice shall be done by the user when the message data transfer is invoked.

NOTE Message data packets exceeding the Ethernet MTU in size (about 1 500 bytes) can be transmitted by TCP in order to avoid problems with packet fragmentation. However, TCP restricts to point-to-point communication.

For interoperable data communication, the destination port assignments (well-known ports) defined in Table A.1 shall be used.

Table A.1 – UDP/TCP port assignments

Protocol	Destination Port
Process Data	17224
Message Data (UDP)	17225
Message Data (TCP)	17225

For receiving any process data telegrams and for receiving UDP message data notification, request and confirm telegrams the well-known port shall be used.

For receiving UDP message data reply telegrams the port the related request was sent from shall be used.

For sending any process data telegrams and for sending UDP message data notification, request and confirm telegrams a private source port different from the well-known port shall be used.

For sending UDP message data reply telegrams any source port different from the one the request was received from can be used.

TCP connections shall be established between a source port different from the well-known port and the well-known port as destination.

Using different well-known port numbers may be allowed for project specific purposes.

The used well-known port numbers should be given as a configuration parameter to the communication stack.

A.3 TRDP FCS Computation

For TRDP FCS CRC computation the CRC32 according to IEEE802.3 is used. The CRC calculation shall be done on the data prepared for transmission – big endian format and byte alignment. The CRC itself shall be appended in little endian format (least significant byte first).

Based on the table below the following algorithm in C programming language notation with start value 'FFFFFFFF'H is used (see Figure A.2 and Figure A.3):

```

/*****
 * GLOBAL FUNCTION DEFINITIONS
 */

/**
 * @internal
 * Computes and returns a 32-bit FCS.
 *
 * @param buf    Input buffer
 * @param len    Length of input buffer
 * @param fcs    Initial (seed) value for the FCS calculation
 *
 * @return Computed fcs value
 */
static UNSIGNED32 fcs32(const UNSIGNED8 buf[], UNSIGNED32 len, UNSIGNED32 fcs)
{
    UNSIGNED32 i;

    for (i=0; i < len; i++)
    {
        fcs = (fcs >> 8)^fcstab[(fcs ^ buf[i]) & 0xff];
    }
    return (fcs ^ 0xffffffff);
}

```

Figure A.2 – FCS Computation

```

/* The FCS-32 generator polynomial:
 * x**0 + x**1 + x**2 + x**4 + x**5 + x**7 + x**8 + x**10 + x**11 +
 * x**12 + x**16 + x**22 + x**23 + x**26 + x**32.
 */
static const UINSIGNED32 fcstab[256] =
{
    0x00000000, 0x77073096, 0xee0e612c, 0x990951ba,
    0x076dc419, 0x706af48f, 0xe963a535, 0x9e6495a3,
    0x0edb8832, 0x79dcb8a4, 0xe0d5e91e, 0x97d2d988,
    0x09b64c2b, 0x7eb17cbd, 0xe7b82d07, 0x90bf1d91,
    0x1db71064, 0x6ab020f2, 0xf3b97148, 0x84be41de,
    0x1dad47d, 0x6ddde4eb, 0xf4d4b551, 0x83d385c7,
    0x136c9856, 0x646ba8c0, 0xfd62f97a, 0x8a65c9ec,
    0x14015c4f, 0x63066cd9, 0xfa0f3d63, 0x8d080df5,
    0x3b6e20c8, 0x4c69105e, 0xd56041e4, 0xa2677172,
    0x3c03e4d1, 0x4b04d447, 0xd20d85fd, 0xa50ab56b,
    0x35b5a8fa, 0x42b2986c, 0xdbbbc9d6, 0xacbcf940,
    0x32d86ce3, 0x45df5c75, 0xdcd60dcf, 0xabd13d59,
    0x26d930ac, 0x51de003a, 0xc8d75180, 0xbfd06116,
    0x21b4f4b5, 0x56b3c423, 0xcfba9599, 0xb8bda50f,
    0x2802b89e, 0x5f058808, 0xc60cd9b2, 0xb10be924,
    0x2f6f7c87, 0x58684c11, 0xc1611dab, 0xb6662d3d,
    0x76dc4190, 0x01db7106, 0x98d220bc, 0xefd5102a,
    0x71b18589, 0x06b6b51f, 0x9fbfe4a5, 0xe8b8d433,
    0x7807c9a2, 0x0f00f934, 0x9609a88e, 0xe10e9818,
    0x7fa0dbbb, 0x086d3d2d, 0x91646c97, 0xe6635c01,
    0x6b6b51f4, 0x1c6c6162, 0x856530d8, 0xf262004e,
    0x6c0695ed, 0x1b01a57b, 0x8208f4c1, 0xf50fc457,
    0x65b0d9c6, 0x12b7e950, 0x8bbeb8ea, 0xfcb9887c,
    0x62dd1dd, 0x15da2d49, 0x8cd37cf3, 0xfbd44c65,
    0x4db26158, 0x3ab551ce, 0xa3bc0074, 0xd4bb30e2,
    0x4adfa541, 0x3dd895d7, 0xa4d1c46d, 0xd3d6f4fb,
    0x4369e96a, 0x346ed9fc, 0xad678846, 0xda60b8d0,
    0x44042d73, 0x33031de5, 0xaa0a4c5f, 0xdd0d7cc9,
    0x5005713c, 0x270241aa, 0xbe0b1010, 0xc90c2086,
    0x5768b525, 0x206f85b3, 0xb966d409, 0xce61e49f,
    0x5edef90e, 0x29d9c998, 0xb0d09822, 0xc7d7a8b4,
    0x59b33d17, 0x2eb40d81, 0xb7bd5c3b, 0xc0ba6cad,
    0xedb88320, 0x9abfb3b6, 0x03b6e20c, 0x74b1d29a,
    0xead54739, 0x9dd277af, 0x04db2615, 0x73dc1683,
    0xe3630b12, 0x94643b84, 0x0d6d6a3e, 0x7a6a5aa8,
    0xe40ecf0b, 0x9309ff9d, 0x0a00ae27, 0x7d079eb1,
    0xf00f9344, 0x8708a3d2, 0x1e01f268, 0x6906c2fe,
    0xf762575d, 0x806567cb, 0x196c3671, 0x6e6b06e7,
    0xfed41b76, 0x89d32be0, 0x10da7a5a, 0x67dd4acc,
    0xf9b9df6f, 0x8ebeeef9, 0x17b7be43, 0x60b08ed5,
    0xd6d6a3e8, 0xa1d1937e, 0x38d8c2c4, 0x4fdff252,
    0xd1bb67f1, 0xa6bc5767, 0x3fb506dd, 0x48b2364b,
    0xd80d2bda, 0xaf0a1b4c, 0x36034af6, 0x41047a60,
    0xdf60efc3, 0xa867df55, 0x316e8ee, 0x4669be79,
    0xcb61b38c, 0x5c66831a, 0x256fd2a0, 0x5268e236,
    0xcc0c7795, 0xbb0b4703, 0x220216b9, 0x5505262f,
    0xc5ba3bbe, 0xb2bd0b28, 0x2bb45a92, 0x5cb36a04,
    0xc2d7ffa7, 0xb5d0cf31, 0x2cd99e8b, 0x5bdeae1d,
    0x9b64c2b0, 0xec63f226, 0x756aa39c, 0x026d930a,
    0x9c0906a9, 0xeb0e363f, 0x72076785, 0x05005713,
    0x95bf14a8, 0xe2b87a14, 0x7bb12bae, 0x0cb61b38,
    0x2d28e9b, 0xe5d5be0d, 0x7cdcefb7, 0x0bdbdf21,
    0x86d3d2d4, 0xf1d4e242, 0x68ddb3f8, 0x1fda836e,
    0x81be16cd, 0xf6b9265b, 0x6fb077e1, 0x18b74777,
    0x88085ae6, 0xff0f6a70, 0x66063bca, 0x11010b5c,
    0x8f659eff, 0xf862ae69, 0x616bffd3, 0x166ccf45,
    0xa0a0278, 0xd70dd2ee, 0x4e048354, 0x3903b3c2,
    0xa7672661, 0xd06016f7, 0x4969474d, 0x3e6e77db,
    0xae16a4a, 0xd9d65adc, 0x40df0b66, 0x37d83bf0,
    0xa9bcae53, 0xdeb9ec5, 0x47b2cf7f, 0x30b5ffe9,
    0xbdbdf21c, 0xcabac28a, 0x53b39330, 0x24b4a3a6,
    0xbad03605, 0xcdd70693, 0x54de5729, 0x23d967bf,
    0xb3667a2e, 0xc4614ab8, 0x5d681b02, 0x2a6f2b94,
    0xb40bbe37, 0xc30c8ea1, 0x5a05dflb, 0x2d02ef8d
};

```

Figure A.3 – FCS Table

A.4 Interaction between TRDP user and TRDP Layer

The TRDP layer shall provide services for process data and message data communication to the TRDP user at upper layer (application) as shown in Figure A.4. This service interface will be defined in an abstract way in the subsequent clauses for process data and message data communication.

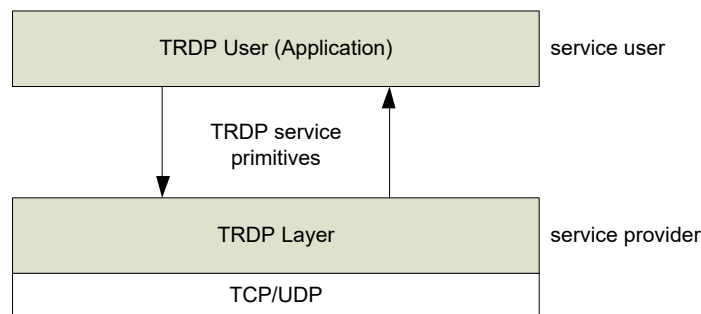


Figure A.4 – TRDP service model

A.5 Communication Identifier (ComId)

All TRDP communication is using a so called ComId transmitted in the header of each PDU. The ComId is a unique identifier for the user data structure of the PDU.

The ComId forms, together with source IP address and destination IP address, a unique identifier of the PDU within the train.

The ComId is application dependent but should be standardized in the application profiles. It should be the reference towards the following parameters of the PD telegram:

- structure (data set) (one-to-one relation)
- source URI
- sink URI
- communication parameters (TTL, QoS)
- timeout
- validity behavior (behavior in case of invalid data / timeout) (only for PD_PDU)
- cycle time (only for PD-PDU)
- redundancy (source is redundant) (only for PD-PDU)
- safe data transmission parameters (if used, only for PD-PDU)

The ComId shall be set to zero (ComId = 0) in the following cases:

- for PDUs with unspecified user dataset.
- in message data confirmation messages
- in message data error messages

The ComIds 1..999 shall not be used by the application since they are reserved for specific use as defined in Table A.2.

Table A.2 – Reserved Comlds

comld	Description
0	unspecified PDU
1	ETBCTRL telegram
2	CSTINFO notification message
3	CSTINFOCTRL notification message
4 – 9	Additional Comlds reserved for communication framework and ETB control service
10	TRDP Echo
11 – 29	Additional Comlds reserved for communication framework and ETB control service
30	Additional Comlds reserved for TRDP statistics data
31	TRDP – statistics request command
32-34	Additional Comlds reserved for TRDP statistics data
35	TRDP – global statistics data
36	TRDP – subscription statistics data
37	TRDP – publishing statistics data
38	TRDP – redundancy statistics data
39	TRDP – join statistics data
40	TRDP- UDP listener statistics data
41	TRDP – TCP listener statistics data
50 – 79	Additional Comlds reserved for communication framework and ETB control service
80	Conformance test – control telegram
81	Conformance test – status telegram
82	Conformance test – confirmation request telegram
83	Conformance test – confirmation reply telegram
84	Conformance test – opTrnDir request telegram
85	Conformance test – opTrnDir reply telegram
86	Conformance test – echo request telegram
87	Conformance test – echo reply telegram
88	Conformance test – echo notification telegram
89 – 99	Additional Comlds reserved for conformance test
100	TTDB – operational train directory status telegram
101	TTDB – operational train directory notification
102	TTDB – train directory information request
103	TTDB – train directory information reply
104	TTDB – consist information request
105	TTDB – consist information reply
106	TTDB – train network directory information request
107	TTDB – train network directory information reply
108	TTDB – operational train directory information request
109	TTDB – operational train directory information reply
110	TTDB – train information complete request
111	TTDB – train information complete reply
112 – 119	Additional Comlds reserved for TTDB

comId	Description
120	ECSP – control telegram
121	ECSP – status telegram
122	ECSP – Confirmation/Correction request
123	ECSP – Confirmation/Correction reply
124 – 129	Additional ComIds reserved for ECSP
130	ETBN – control request
131	ETBN – status reply
132	ETBN – train network directory request
133	ETBN – train network directory reply
134 – 139	Additional ComIds reserved for ETBN
140	TCN-DNS – resolving request telegram (query)
141	TCN-DNS – resolving reply telegram
142 – 149	Additional ComIds reserved for TCN-DNS
150 – 199	Additional ComIds reserved for communication framework and ETB control service
200 – 999	ComIds reserved for other standards and norms

A.6 Process Data

A.6.1 Communication model

The TRDP process data protocol defines the exchange of PD-PDUs for the transfer of process data.

PD-PDUs shall be cyclically transmitted or transmitted on request between a publisher and one or many subscribers using a connectionless and unconfirmed TRDP service.

A.6.2 Roles

The communication partners in a process data transfer can have different roles:

- publisher: the source of process data
- subscriber: the sink of process data
- requester: requesting the publisher(s) to publish its process data

Publisher and requester coincide in push communication pattern.

A subscriber may also be a requester in pull communication patterns.

A.6.3 Communication pattern

A.6.3.1 Push communication pattern

Process data exchange shall support the following push communication pattern as defined in IEC 61375-1:

- a) point to point, cyclic without acknowledge, source knows the sink (Figure A.5);
- b) point to multipoint, cyclic without acknowledge, source knows the sink, e.g. redundancy groups (Figure A.6);

- c) point to multipoint, cyclic without acknowledge, source does not know the sink (Figure A.6).

NOTE From communication point of view there will be no difference in a point to multipoint communication pattern whether the source knows the sinks or not. It is listed here only for completeness.

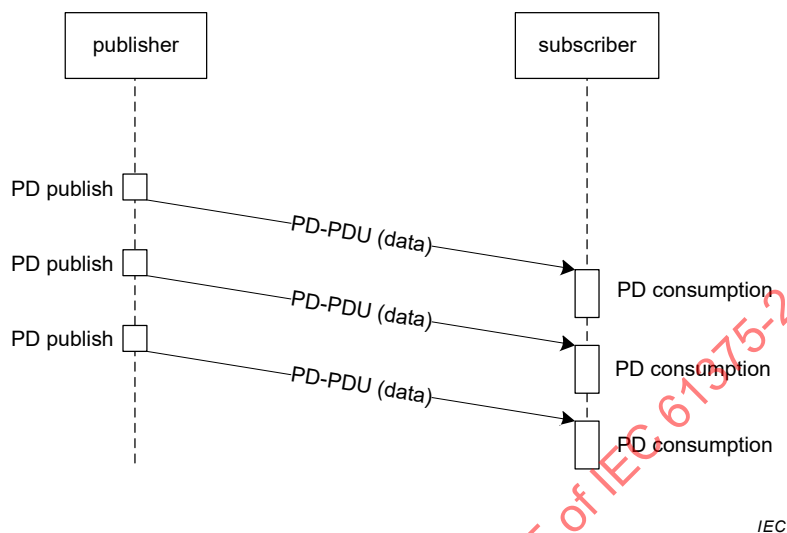


Figure A.5 – PD push pattern (point to point)

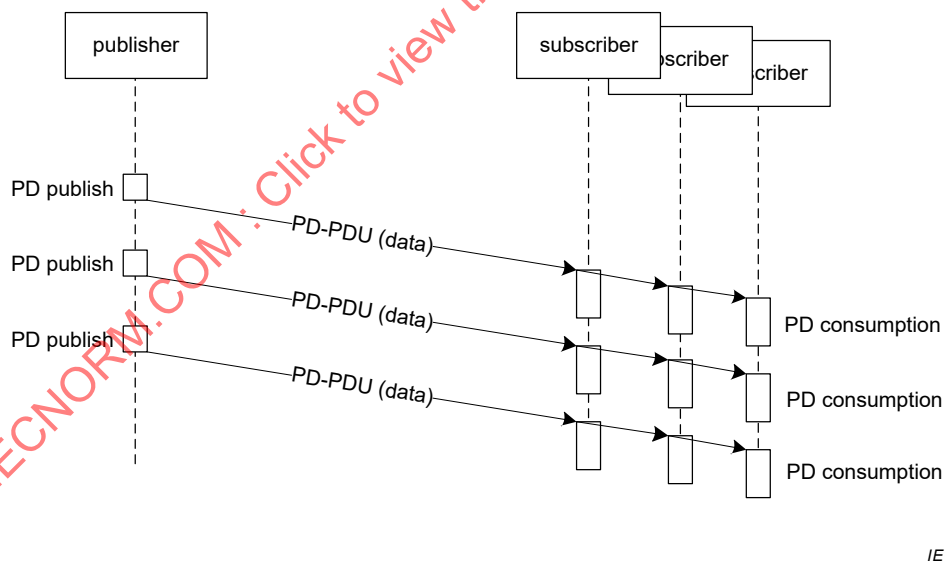


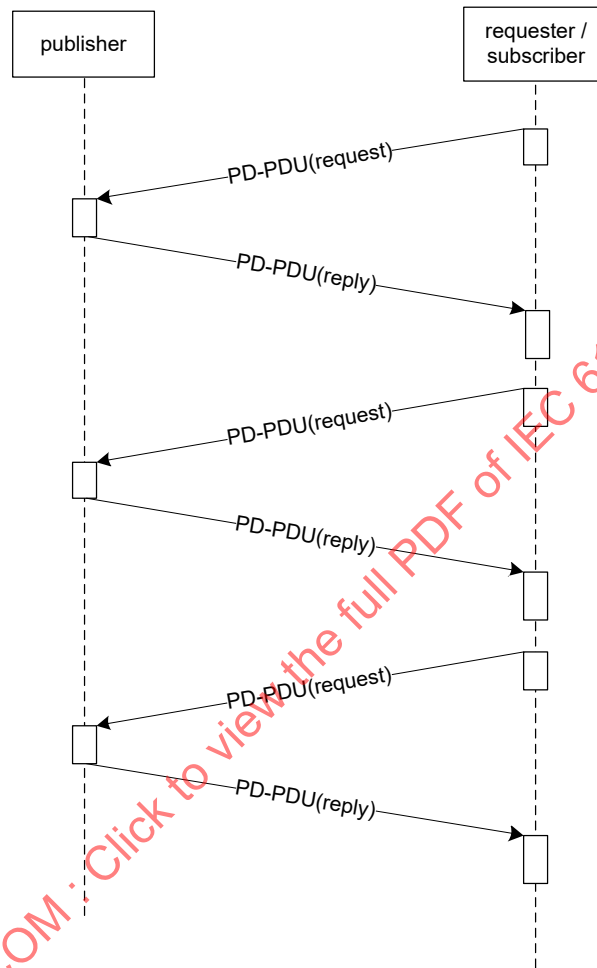
Figure A.6 – PD push pattern (point to multipoint)

A.6.3.2 Pull communication pattern

Process data exchange shall support the following pull communication pattern as defined in IEC 61375-1:

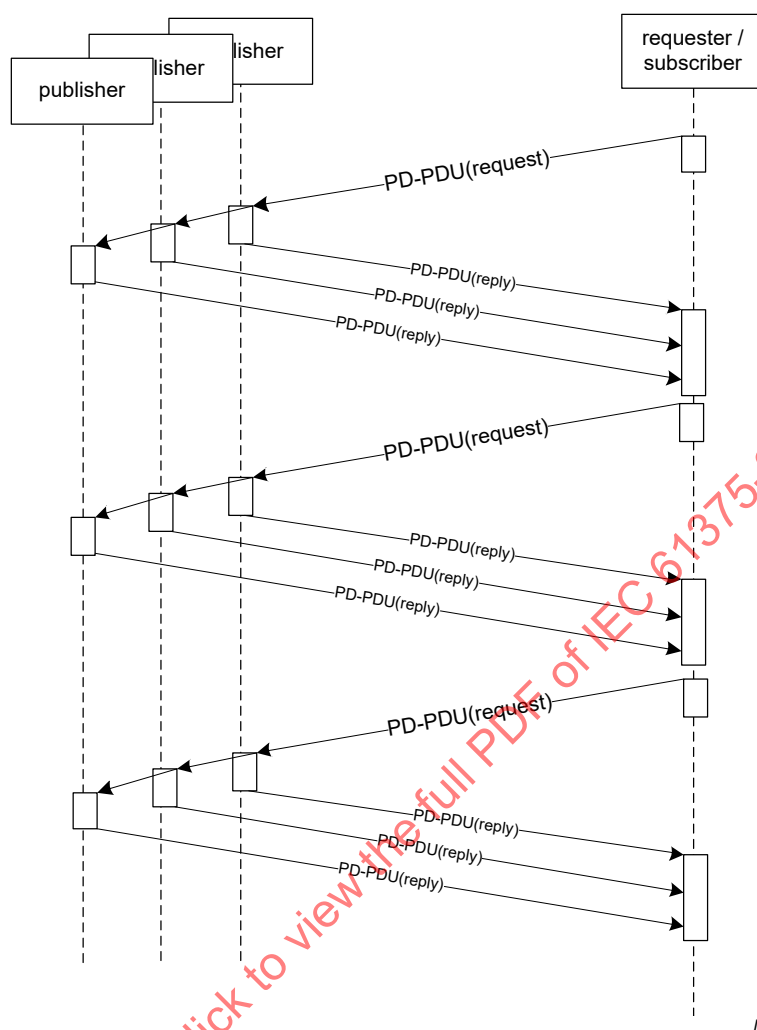
- Point to point, without acknowledge, sink knows the source (Figure A.7).
- Multipoint to point, without acknowledge, sink does not know the source (Figure A.8).

- Point to multipoint, without acknowledge, sink knows the source (Figure A.9). Here, one dedicated subscriber is requesting the known publisher to send its PD-PDU.
- Multipoint to multipoint, without acknowledge, sink does not know the source (Figure A.10). Here, one dedicated subscriber is requesting one or multiple unknown publisher to send their PD-PDU.



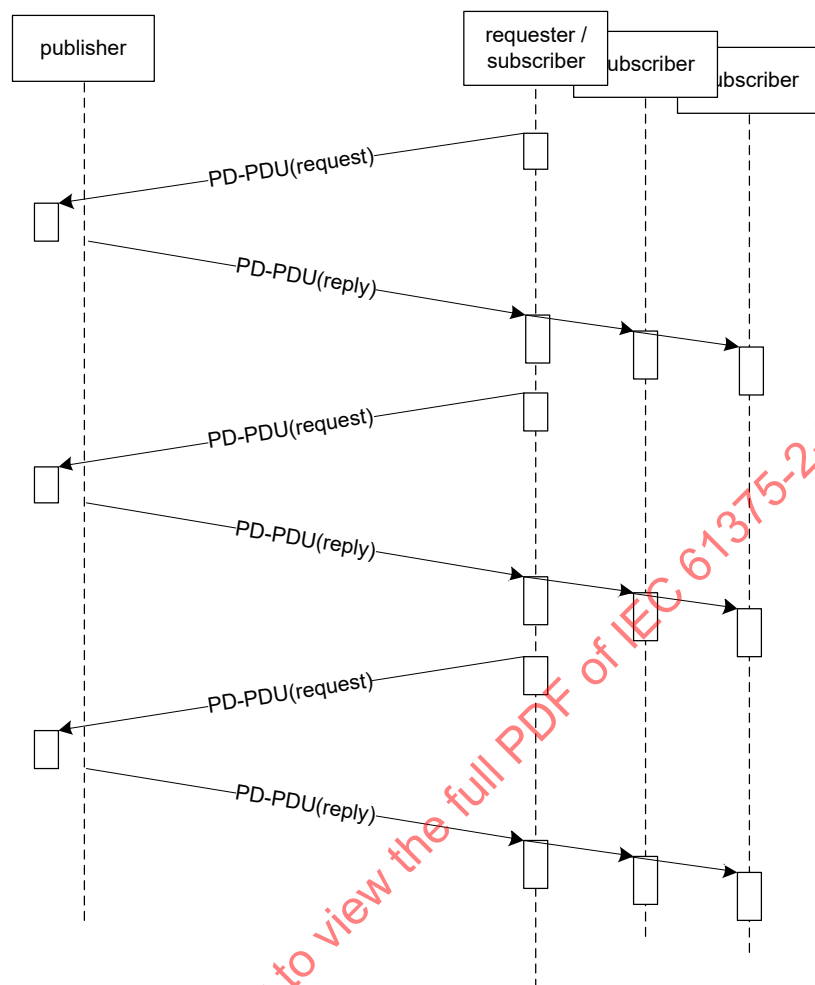
IEC

Figure A.7 – PD pull pattern (point to point, sink knows source)



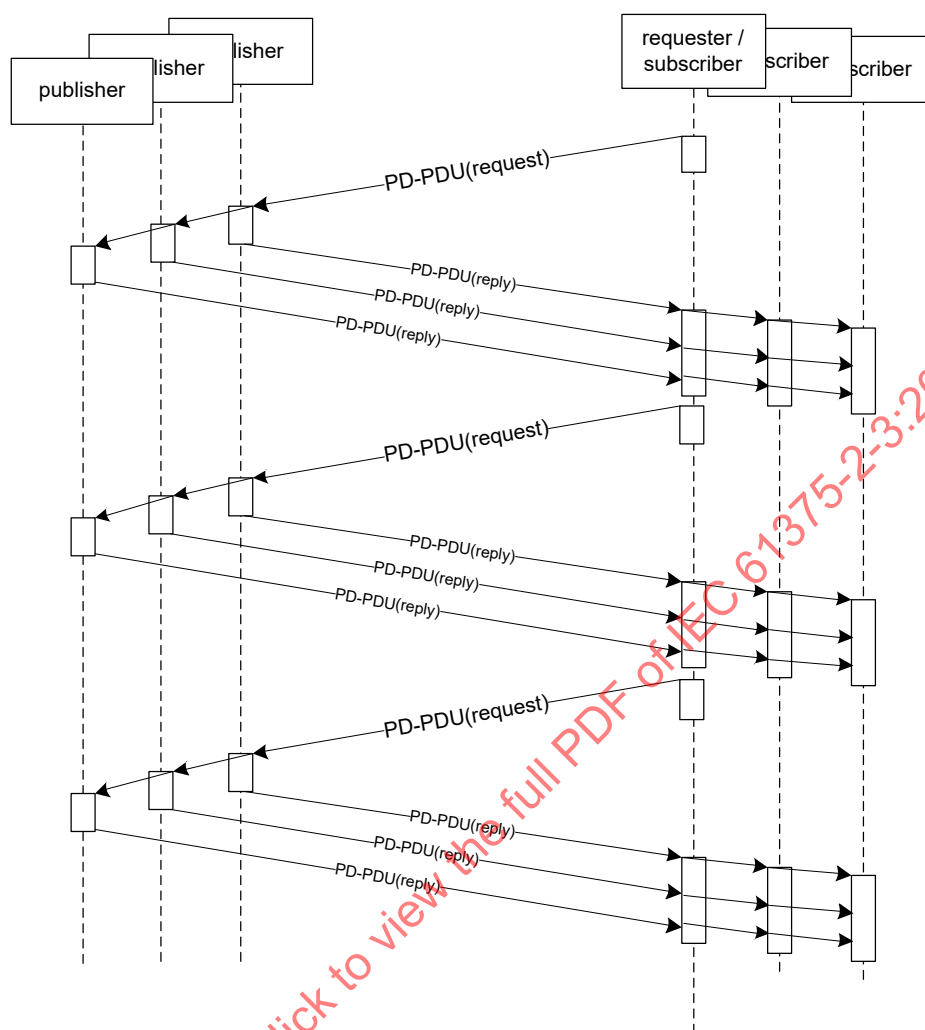
IEC

Figure A.8 – PD pull pattern (multipoint to point, sink does not know source)



IEC

Figure A.9 – PD pull pattern (point to multipoint, sink knows source)



IEC

Figure A.10 – PD pull pattern (multipoint to multipoint, sink does not know source)

A.6.4 Addressing

A publisher/subscriber shall use an IP unicast address for addressing a known subscriber/publisher.

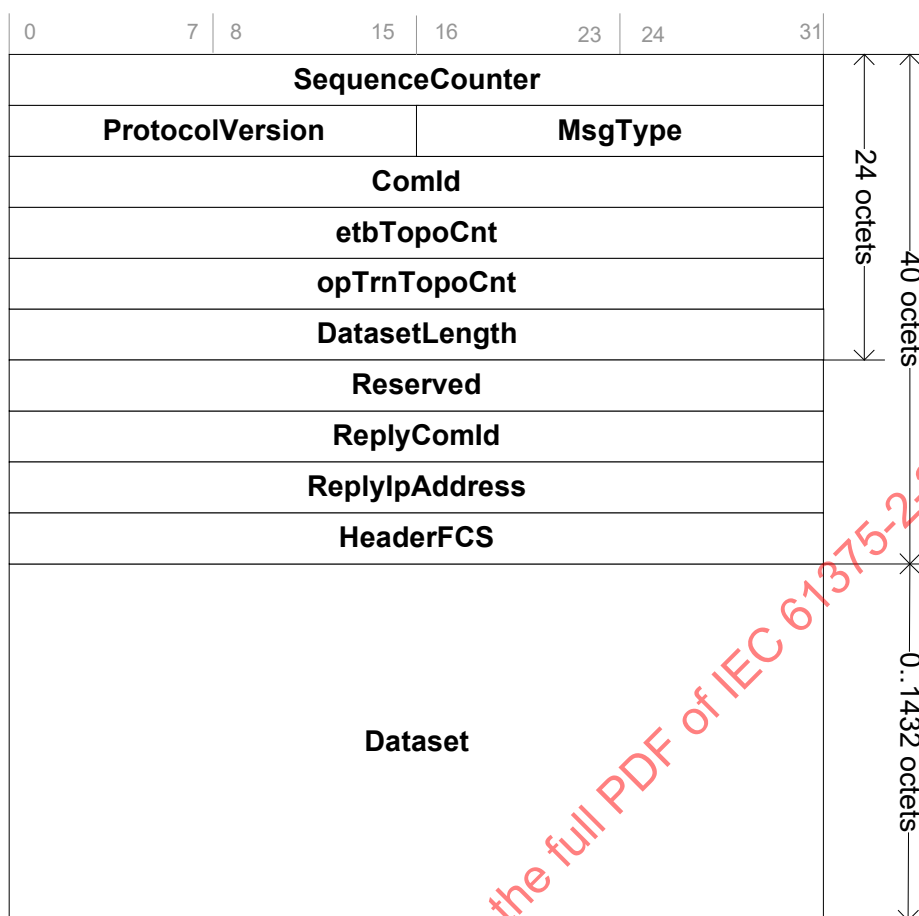
A publisher/subscriber shall use an IP multicast address for addressing groups of known subscribers/publishers (e.g. redundancy groups).

A publisher/subscriber shall use an IP multicast address for addressing unknown subscribers/publishers.

Address ranges which can be used for process data communication are defined in IEC 61375-2-5 and in 5.4.5.

A.6.5 PD-PDU

The structure of a PD-PDU is defined in Figure A.11 and in ASN.1 notation, with additional explanation given in Table A.3.



IEC

Figure A.11 – PD-PDU

```

PD_PDU ::= RECORD
{
    sequenceCounter  UINT32          -- message sequence counter
    protocolVersion  VERSION         -- protocol version
    msgType          UINT16          -- type of the message
    comId            UINT32          -- communication identifier
    etbTopoCnt       UINT32          -- etbTopoCnt value
    opTrnTopoCnt     UINT32          -- opTrnTopoCnt value
    datasetLength    UINT32          -- length of the array 'dataset'
    reserved01       UINT32          -- reserved for future use (= 0)
    replyComId       UINT32          -- requested communication identifier
    replyIpAddress   UINT32          -- reply IP address
    headerFcs        UINT32          -- header checksum
    dataset          UINT8 [datasetLength] -- user data set
}

```

Table A.3 – PD-PDU parameters

Parameter	Description	Value
sequenceCounter	The sequence counter: • Shall be managed for sending process telegrams per ComId/MsgType at each requester/publisher. • Shall be managed (stored) for received process telegrams per SourceIPAddr/ComId/MsgType at each publisher/subscriber. • Shall be incremented with each sending of the process telegram. Telegrams sent in parallel via different subnets are sent with the same sequence counter to detect duplication at receiver side. • Can be used for communication layer surveillance (PD sending). A surveillance that the application is still updating its process data (user data) can be done via the safe data transmission protocol (SDTv2, see Annex B) or can be implemented by the application (e.g. via a life sign).	Computed, start value 0
protocolVersion	The protocol version shall consist of: • Higher significant octet: mainVersion, incremented for incompatible changes • Lower significant octet: subVersion, incremented for compatible changes EXAMPLE – '0102'H = protocol version 1.2	Fixed
msgType	Type of the telegram. 'Pr' = PD Request 'Pp' = PD Reply 'Pd' = PD Data 'Pe' = PD Data (Error) NOTE Coded in ASCII for better reading with open protocol analyzer tools (e.g. Wireshark) MsgType 'Pe' is only for notification purpose and should not be sent over the bus	'5072'H ('Pr') '5070'H ('Pp') '5064'H ('Pd') '5065'H ('Pe')
comId	Identifier of the user dataset. See also A.5.	set by user
etbTopoCnt	The ETB topography counter: • shall be used (train addressing) as defined in IEC 61375-2-5 (parameter 'etbTopoCnt') • shall be set by the user • shall be set for all communication over the ETB • shall be set if a valid opTrnTopoCnt is set • optional in all other cases. Shall be set to 0 (= invalid) if not used.	0..2³²-1 set by user
opTrnTopoCnt	The operational train topography counter: • shall be used as defined in 5.3.3 • shall be set by the user • shall be set for communication between functions which use information from the operational train directory (e.g. side selective operations) • shall be set when the source device used the operational train directory to retrieve the destination IP address (e.g. resolving of URI 'vcu.leadVeh.anyCst.anyCITrn.ITrn') • optional in all other cases. Shall be set to 0 (= invalid) if not used.	
datasetLength	The dataset length:	0..1432

Parameter	Description	Value
	- shall be the length of the user data set in number of octets without padding octets. - shall be the primary information about the user data size NOTE In case of fixed size data sets this is redundant information because the dataset length is already defined by the ComId.	Computed
reserved01	Reserved for future use	set to 0
replyComId	The requested ComId: - shall be used only in a PD request - shall be used as ComId in the reply - If set to 0, the ComId of the request shall be used for the reply - If set to 0, and the ComId of the request is also set to 0, the reply shall be sent as an unspecified PDU.	set by user in PD request, in other telegram types set to 0
replyIpAddress	The reply IP address: - shall be used only in a PD request - shall be used as destination IP address in the reply - If set to 0, the source IP address of this request shall be used for the reply.	set by user for request otherwise set to 0
headerFCS	The header frame check sequence: - shall be calculated for the PD-PDU header - shall not include the HeaderFCS itself.	Computed according to A.3
dataset	The user data If the user data length is not a multiple of 4 octets, octets with a value of 0 (zero) should be appended until a multiple of 4 octets is reached (padding bytes).	

A.6.6 Interaction between application and TRDP protocol layer

A.6.6.1 Service primitives

The TRDP layer shall provide a service for process data communication to the TRDP user at upper layer. The service primitives listed in Table A.4 shall be provided.

NOTE The service primitives are defined in an abstract way in order not to restrict implementations. The specification of the service interface itself is not in the scope of this part of the standard.

Table A.4 – TRDP service primitives

Role	Service primitive	Parameter	Description
publisher	PD.publish		TRDP user → TRDP layer service user hands over a new process data packet
		Handle	Handle for this publishing, returned by the TRDP layer
		etbTopoCnt	as defined in Table A.3
		opTrnTopoCnt	as defined in Table A.3
		ComId	as defined in Table A.3
		DatasetLength	Actual user data length to be sent as defined in Table A.3

Role	Service primitive	Parameter	Description
		Dataset	Buffer with user data to be sent as defined in Table A.3
		CycleTime	Cycle time (txTime) for sending PD. Unit: 10 ⁻⁶ s (µs)
		DestinationIpAddress[]	Destination IP address(es) to send the PD to.
		redundant	FALSE means that the PD are published immediately, TRUE means that PD are only published if redundant ComIds are activated
	PD.unPublish		TRDP user → TRDP layer Stop publishing
		Handle	Handle which has been given by the TRDP layer
	PD.putData		
		Handle	Handle which has been given by the TRDP layer
		DatasetLength	Actual user data length to be sent as defined in Table A.3
		Dataset	Buffer with user data to be sent as defined in Table A.3
Requester	PD.activateRed		TRDP user → TRDP layer service user activates the publishing of the ComIds marked as redundant
	PD.deactivateRed		TRDP user → TRDP layer service user deactivates the publishing of the ComIds marked as redundant
	PD.request		TRDP user → TRDP layer TRDP layer sends a PD-PDU.request to one or multiple publishers. Cyclically called from application.
		Handle	handle for this subscription of the reply ComId; returned by the TRDP layer
		etbTopoCnt	as defined in Table A.3
		opTrnTopoCnt	as defined in Table A.3
		ComId	ComId to be sent, as defined in Table A.3
		DatasetLength	data set length to be sent, as defined in Table A.3
		DataSet	User data set to be sent, as defined in Table A.3
		DestinationIpAddress[]	Destination IP address(es) to send the PD request to.
		redundant	FALSE means that the PD are published immediately, TRUE means that PD are only published if redundant ComIds are activated

Role	Service primitive	Parameter	Description
Subscriber	PD.subscribe	ReplyComId	as defined in Table A.3
		ReplyIpAddress	as defined in Table A.3
			TRDP user → TRDP layer User subscribes to process data using possible filter for ComId, SourceIpAddress, SourceIpAddress range, DestinationIpAddress.
		Handle	handle for this subscription; returned by the TRDP layer
		etbTopoCnt	as defined in Table A.3
		opTrnTopoCnt	as defined in Table A.3
		ComId	as defined in Table A.3
		DatasetLength	as defined in Table A.3
		Dataset	as defined in Table A.3
		SourceIpAddress	IP source address, generated out of respective URI's using DNS. Defines the lower IP address in case of an IP address range (see 5.4.4.6.2) Set to 0 if no filtering requested
		SourceIpAddress2	Defines the upper IP address in case of an IP address range (see 5.4.4.6.2) IP source address, generated out of respective URI's using DNS Set to 0 if not used
		DestinationIpAddress	IP destination address, generated out of respective URI's using DNS Set to 0 if no filtering requested
		Subnet	Select the subnet(s) to subscribe; designated one or all of the redundant networks.
		Timeout	Timeout (rxTime) for PD to be received Unit: 10^{-6} s (μs)
	PD.unsubscribe		TRDP user → TRDP layer User unsubscribes to process data using possible filter for ComId, SourceIpAddress, DestinationIpAddress.
		Handle	handle returned by the TRDP layer in PD.subscribe
	PD.indicate		TRDP layer → TRDP user TRDP layer indicates new process data or a timeout. The timeout value is given with PD.subscribe.
		Handle	handle returned by the TRDP layer in PD.subscribe
		IndType	- process data ('Pd') - process data reply ('Pp') - process data request ('Pr') - process data error ('Pe')

Role	Service primitive	Parameter	Description
		etbTopoCnt	Received topo count, as defined in Table A.3
		opTrnTopoCnt	Received topo count, as defined in Table A.3
		ComId	Received ComId, as defined in Table A.3
		DatasetLength	Received user data size, as defined in Table A.3
		Dataset	Received user data, as defined in Table A.3
		SequenceCounter	Received sequence counter, as defined in Table A.3
		SourceIpAddress	Received source IP address
		DestinationIpAddress	Received destination IP address, as defined in Table A.3
		ReplyComId	Received reply ComId, as defined in Table A.3
		ReplyIpAddress	Received reply IP address, as defined in Table A.3
		Status	0 OK < 0 NOK
		In case of a TRDP error ('Pe') the value is supplied by TRDP.	
		• 1 – timeout • 6 – no subscription • 5 – no memory • 7 – data invalid	
	PD.poll		TRDP user → TRDP layer User polls for new PD instead of using the indication
		Handle	handle returned by the TRDP layer in PD.subscribe
		IndType	• process data ('Pd') • process data reply ('Pp') • process data request ('Pr') • process data error ('Pe')
		etbTopoCnt	Received topo count, as defined in Table A.3
		opTrnTopoCnt	Received topo count, as defined in Table A.3
		ComId	Received ComId, as defined in Table A.3
		DatasetLength	Received user data size, as defined in Table A.3
		Dataset	Received user data, as defined in Table A.3
		SequenceCounter	Received sequence counter, as defined in Table A.3
		SourceIpAddress	Received source IP address
		DestinationIpAddress	Received destination IP

Role	Service primitive	Parameter	Description
		ReplyComId	Received reply ComId, as defined in Table A.3
		ReplyIpAddress	Received reply IP address, as defined in Table A.3
		Status In case of a TRDP error ('Pe') the value is supplied by TRDP. • 1 – timeout • 5 – no memory • 6 – no subscription • 7 – data invalid	0 OK < 0 NOK

A.6.6.2 PD User access

The TRDP user shall have two possibilities to retrieve PD: either via a poll mechanism, typically used in cyclic user tasks, or via indication mechanism, where the TRDP layer notifies the user when new data are available or when there is a timeout.

A.6.6.3 Interaction sequence – PD Pull Pattern

The sequence chart in Figure A.12 defines the allowed sequence of the service primitives for the PD pull pattern, PD.request service primitive and related request telegram are only called by a requester.

NOTE 1 Dotted lines in the sequence charts are used to indicate options.

Process data are prepared cyclically by the publisher and shall be given to the TRDP layer calling the PD.putData primitive.

To receive a request, the publisher shall subscribe for it.

Until receiving the first request telegram matching to the filter criteria of the subscription, related data in the receive buffer of the replier shall be marked as invalid.

Also a request for already cyclically published PDU shall be possible.

Topography counters of an incoming request telegram shall be checked in accordance to A.6.7.

Each incoming request telegram shall be responded by the TRDP layer using the available process data, the reply IP address and the reply ComId given in the request telegram.

If the reply IP address of the request telegram is 0, the source IP address of the request telegram shall be used as destination IP address for the reply.

To receive the related reply for a request, the requester needs to subscribe for it. Destination IP address, source IP address and source IP address range are possible filter criteria.

Until receiving the first reply telegram matching to the filter criteria of the subscription, related data in the receive buffer of the subscriber shall be marked as invalid. Before sending the request telegram, the related reply data in the receive buffer of the subscriber shall be set to invalid.

NOTE 2 This is to prevent that the user fetches outdated reply data after it has requested new reply data.

Timeout supervision at subscriber TRDP layer shall be restarted after sending the request telegram.

The request can contain user data.

Before sending out a request, the TRDP layer shall check the topology counters submitted with the request against the actual topology counters. At least one of the cases listed in Table A.5 for the topology counters shall be fulfilled.

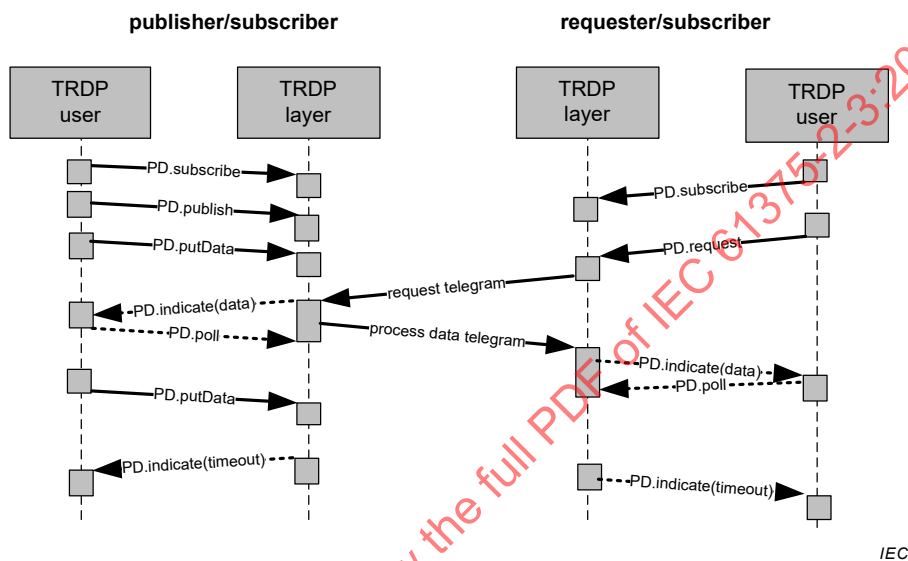


Figure A.12 – Interaction sequence chart for PD pull pattern

A.6.6.4 Interaction sequence – PD Push Pattern

The sequence chart in Figure A.13 defines the allowed sequence of the service primitives for the PD push pattern.

Process data are prepared cyclically by the publisher and shall be given to the TRDP layer calling the PD.putData primitive.

The publisher TRDP layer shall send the data in the configured cycle to the configured address.

The publisher TRDP layer shall send out the data only after checking the locally stored topology counters submitted with the publish against the actual topology counters. At least one of the cases listed in Table A.5 for the topology counters shall be fulfilled.

Any subscriber can subscribe for the process telegram using ComID, destination IP address and source IP address of the process telegram as possible filter criteria.

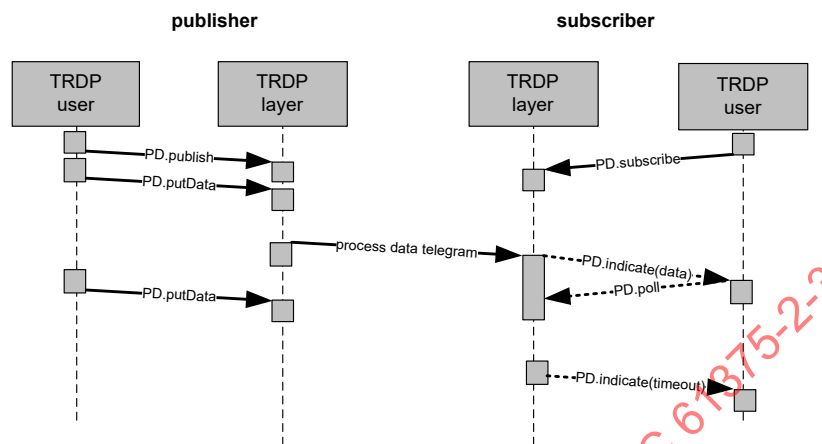
Timeout supervision at subscriber TRDP layer shall be started after subscription.

Timeout supervision at subscriber TRDP layer shall be restarted after receiving the related PD PDU.

The subscriber TRDP layer shall check the topology counters of the received telegram against the actual topology counters and against the topology counters submitted with the

subscription. At least one of the cases listed in Table A.5 for the topography counters shall be fulfilled.

Until receiving the first valid telegram matching to the filter criteria the data shall be marked as invalid.



IEC

Figure A.13 – Interaction sequence chart for PD push pattern

A.6.6.5 PD Redundancy Handling

The sequence chart in Figure A.14 shows the sequence of the service primitives for redundancy handling of the push pattern.

The service primitives for redundancy handling shall be used in the same way by the publisher for the pull and the push pattern.

If a redundant device enters the redundancy leader state it shall call PD.activate to start publishing process data related to ComIds marked as redundant.

If a redundant device enters the redundancy follower state it shall call PD.deactivate to stop publishing process data related to ComIds marked as redundant.

Starting TRDP, publishers of redundant process data (identified by ComId marked as redundant) shall be initialized in redundancy follower mode (publishing deactivated).

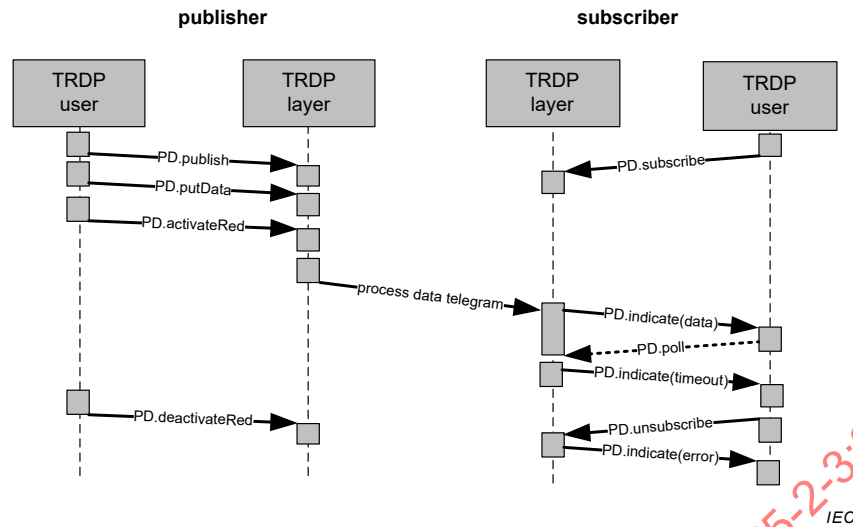


Figure A.14 – Interaction sequence chart for redundant PD handling

A.6.7 Topography counter check

Before sending a telegram, the topography counters of the telegram shall be checked against the actual topography counters to ensure that the sending application shares the actual train backbone view and operational train view.

After reception of a telegram, the topography counter values shall be checked to ensure that publisher/requester and subscriber share an identical train backbone view and operational train view. For this check, the topography counter values received are compared with the topography counter values submitted in the PD.subscribe service primitive.

The topography counter check is passed if at least one of the cases listed in Table A.5 came true.

Table A.5 – Topography counter check

Publisher	Actual topography counter values		Locally stored topography counter values submitted with the publish	
Subscriber	Actual topography counter values		Topography counter values of the received telegram	
	Topography counter values of the received telegram		Locally stored topography counter values submitted with the subscription	
Case	etbTopoCnt	opTrnTopoCnt	etbTopoCnt	opTrnTopoCnt
1	any	any	0	0
2	0	equal	0	equal
3	equal	any	equal	0
4	equal	equal	equal	equal
5 (1)	0	0	equal	equal

(1) Case applies only for subscriber comparing topography counter values of the received telegram with the locally stored topography counter values submitted with the subscription. Used to receive local telegrams.

Key:

0 = topography counter value set to 0 (don't care)

any = topography counter has any value in the range $0..(2^{32}-1)$

equal = local and received topography counter values are valid ($1..(2^{32}-1)$) and identical

A.6.8 State Machine

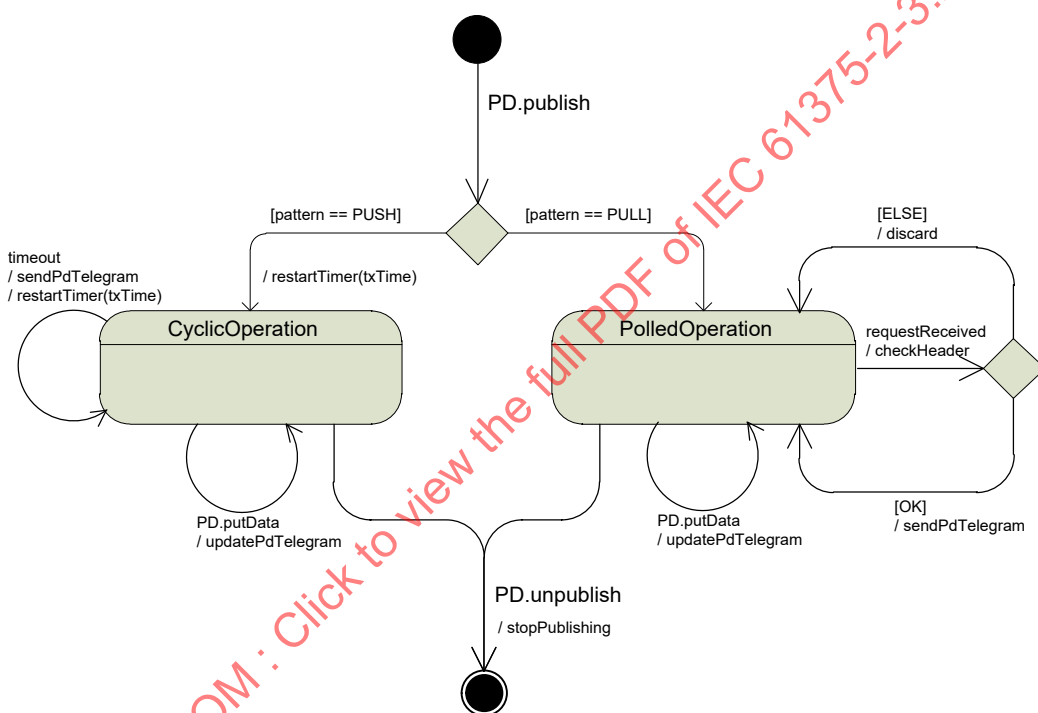
A.6.8.1 Publishing PD-PDU(data)

A publisher shall publish PD-PDU(data) when a defined time cycle terminates (push pattern) and/or when it receives a request from a dedicated subscriber or an independent communication device.

Not to overload subscribers, the publisher of pushed PD-PDU shall apply a traffic shaping mechanism for equal distribution of the PD-PDU's over the time.

The publishing of PD-PDU(data) is defined with the state diagram depicted in Figure A.15.

NOTE If there is a pull request for a PD-PDU that is already published using the push pattern, the related reply is sent to the address given by the request independently of the push pattern address and cycle.



IEC

Figure A.15 – PD State diagram publisher

Guards, triggers, actions and states are described in Table A.6 to Table A.9.

Table A.6 – PD publisher state diagram – guards

Guards	Description
pattern	configured communication pattern for the PD-PDU exchange Values: PUSH PULL
checkTopoCounts	Check the locally stored topology counters submitted with the publish against the actual topography counters. At least one of the cases listed in Table A.5 for the topography counters is fulfilled.

Table A.7 – PD publisher state diagram – triggers

Triggers	Description
PD.publish	User called PD.publish service primitive
PD.unpublish	User called PD.unpublish service primitive
PD.putData	New process data packet has been prepared by the application for publishing (service primitive PD.putData)
requestReceived	A request telegram (type 'Pr') has been received
timeout	The time txTime expired

Table A.8 – PD publisher state diagram – actions

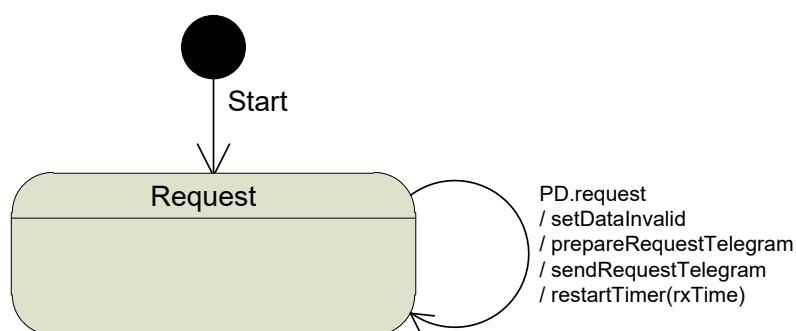
Actions	Description
updatePdTelegram	Take process data packet from application and generate process data telegram for publishing
checkHeader	Check validity of received PD request telegram: - HeaderFCS - ProtocolVersion Check existence of a publisher for that ComId If a publisher exists, check that at least one of the cases listed in Table A.5 for the topography counters is fulfilled. OK = all checks passed
sendPdTelegram	publish (send) the process data telegram
restartTimer(txTime)	restart the timer txTime = cycle time for sending PD as defined in PD.publish service primitive
stopPublishing	Stop publishing on request, stop the timer
discard	Discard the telegram

Table A.9 – PD publisher state diagram – states

States	Description
CyclicOperation	The publisher sends PD-PDUs cyclically triggered by a timer
PolledOperation	The publisher sends PD-PDUs triggered by a PD-PDU(request)

A.6.8.2 Request publishing of PD-PDU (reply)

One dedicated subscriber or an independent communication device may request one or many publishers to send their PD-PDUs. The requesting of PD-PDUs is defined with the state diagram depicted in Figure A.16.



IEC

Figure A.16 – PD State diagram requester

Triggers, actions and states are described in Table A.10 to Table A.13.

Table A.10 – PD publisher state diagram – guards

Guards	Description
checkTopoCounts	Check the topography counters submitted with the request against the actual topography counters. At least one of the cases listed in Table A.5 for the topography counters shall be fulfilled.

Table A.11 – PD requester state diagram – triggers

Triggers	Description
PD.request	PD.request service primitive called by application

Table A.12 – PD requester state diagram – actions

Actions	Description
prepareRequestTelegram	prepare a request telegram to one or many publishers
sendRequestTelegram	send a request telegram to one or many publishers
restartTimer	(re-)start timer time = rxTime (timeout value given in PD.subscribe service primitive)
setDataInvalid	Sets the related data in the receive buffer to initialized/invalid.

Table A.13 – PD requester state diagram – states

States	Description
Request	The requester sends request telegram. Sending of the request telegram is triggered by the application using PD.request.

A.6.8.3 Receiving PD-PDU(reply/data)

See Figure A.17.

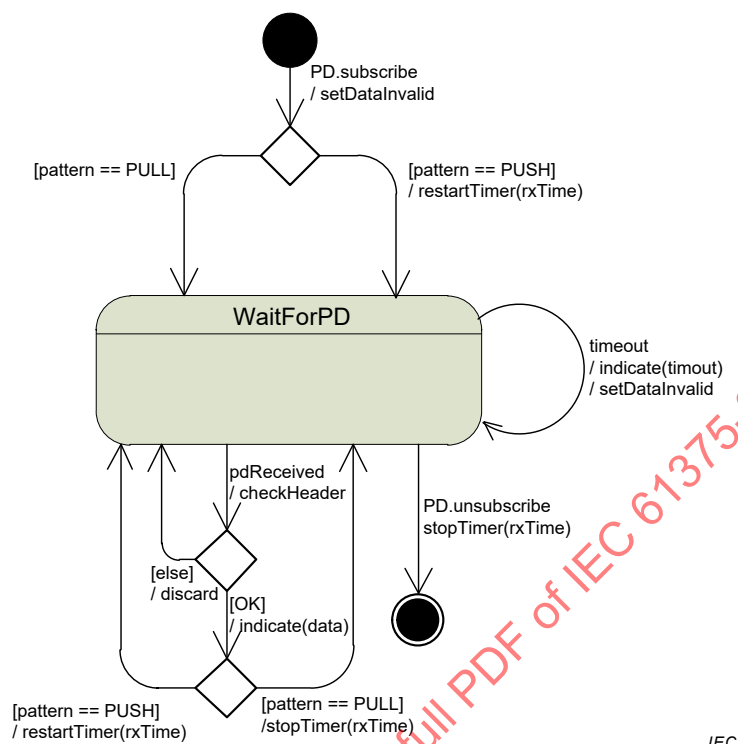


Figure A.17 – PD State diagram subscriber

Triggers, actions and states are described in Table A.14 to Table A.17.

Table A.14 – PD subscriber state diagram – triggers

Triggers	Description
PD.subscribe	PD.subscribe service primitive called by application
pdReceived	A process or request telegram has been received
Timeout	The rxTime expired
PD.unsubscribe	PD.unsubscribe service primitive called by application

Table A.15 – PD subscriber state diagram – guards

Guards	Description
pattern	configured communication pattern for the PD-PDU exchange Values: PUSH PULL
checkTopoCounts	Check the topography counters of the received telegram against the topography counters submitted with the subscription. At least one of the cases listed in Table A.5 for the topography counters shall be fulfilled.

Table A.16 – PD subscriber state diagram – actions

Actions	Description
indicate(data)	Notify application about a new received process or request telegram (service primitive PD.indicate)
indicate(timeout)	Notify application about timeout (service primitive PD.indicate)
checkHeader	Check header for correct FCS, version and type. Check filter criteria OK = all checks passed
setDataInvalid	Sets the PD to initialized/invalid
stopTimer(timer)	Stop the timer which is related to the subscription
restartTimer(time)	restart the timer time = rxTime (timeout value given in PD.subscribe service primitive) The timer shall be started first time when the subscription is done. For PUSH pattern it shall be restarted with every received process data telegram. For PULL pattern it shall be restarted with each request.
discard	Discard the telegram

Table A.17 – PD subscriber state diagram – states

States	Description
WaitForPD	Wait until a new process telegram is received

A.7 Message Data

A.7.1 Communication model

The TRDP message data protocol defines the message data exchange between a caller and one or many repliers over a confirmed TRDP service. Since it is thought only for real-time message data, the length of a telegram is limited to 64 kBytes. The caller is sending a request message with or without user data to the replier(s) and the replier(s) shall send a reply message with or without user data in return, if required by the caller. The caller then shall quit reception of the reply by sending a confirmation if required by the replier.

NOTE This gives a “guaranteed” transfer of data as the caller/replier is notified about the correct reception and processing of the message data.

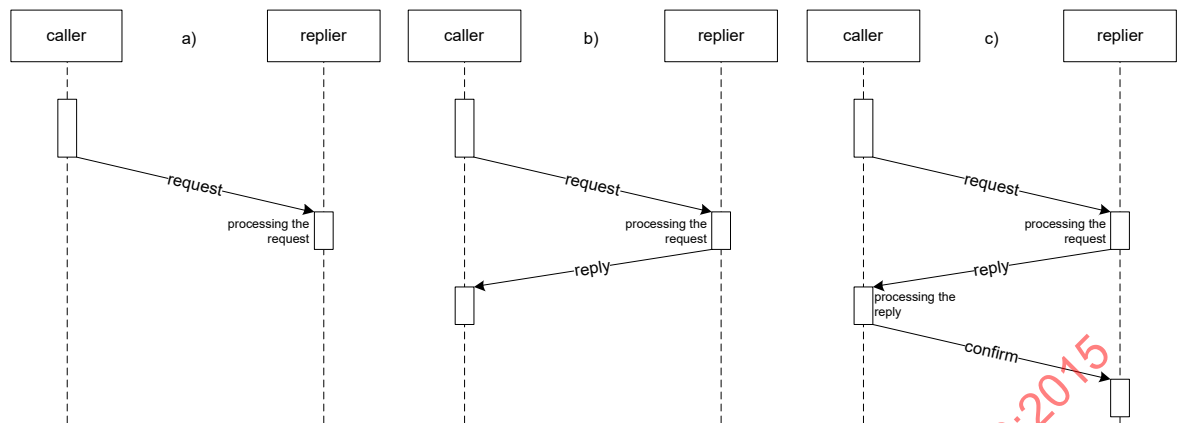
A caller shall be able to define by the request type whether a reply is expected or not.

A replier shall be able to define by the reply type whether a confirmation of its reply is expected or not.

As a consequence, three message data transfer options shall be provided by TRDP (Figure A.18):

- request without reply (‘notification’);
- request with reply but without confirmation (‘request without confirmation’);
- request with reply and confirmation (‘request with confirmation’).

TRDP shall provide two mechanisms to transfer message data (via UDP and via TCP); the different service primitives of the two possibilities shall not be mixed.



IEC

Figure A.18 – Message data transfer options

A.7.2 Roles

The communication partners in a message data transfer can have different roles:

caller:	the source of message data in push communication pattern
	the sink of message data in pull communication pattern
replier:	the source of message data in pull communication pattern
	the sink of message data in push communication pattern

A.7.3 Communication pattern

A.7.3.1 Push communication pattern

Message data exchange shall support the following push communication pattern as defined in IEC 61375-1 (for exceptions see note):

- point to point , sporadic with acknowledge, source knows the sink;
- point to point , sporadic without acknowledge, source knows the sink;
- point to multipoint, sporadic with acknowledge, source knows the sink;
- point to multipoint, sporadic without acknowledge, source knows the sink;
- point to multipoint, sporadic with acknowledge, source does not know the sink;
- point to multipoint, sporadic without acknowledge, source does not know the sink.

NOTE e) is not required in IEC 61375-1.

A.7.3.2 Pull communication pattern

Message data exchange shall support the following pull communication pattern as defined in IEC 61375-1 (for exceptions see note):

- point to point , sporadic with acknowledge, sink knows the source;
- point to point , sporadic without acknowledge, sink knows the source;
- point to multipoint, sporadic with acknowledge, sink knows the source;
- point to multipoint, sporadic without acknowledge, sink knows the source;

- e) point to multipoint, sporadic on first acknowledge, sink does not know the source;
- f) point to multipoint, sporadic without acknowledge, sink does not know the source.

NOTE e) is not required in IEC 61375-1.

A.7.4 Addressing

A caller shall use an IP unicast address or an IP multicast address for addressing known replier(s).

A caller shall use an IP multicast address for addressing unknown repliers.

A caller may use an IP multicast address for addressing a known replier redundancy group.

A replier shall respond to the caller's unicast address.

Address ranges which can be used for message data communication are defined in IEC 61375-2-5.

A.7.5 MD-PDU

The structure of a MD-PDU is defined in Figure A.19 and subsequently in ASN.1 notation, with additional explanation given in Table A.18.

IECNORM.COM : Click to view the full PDF of IEC 61375-2-3:2015

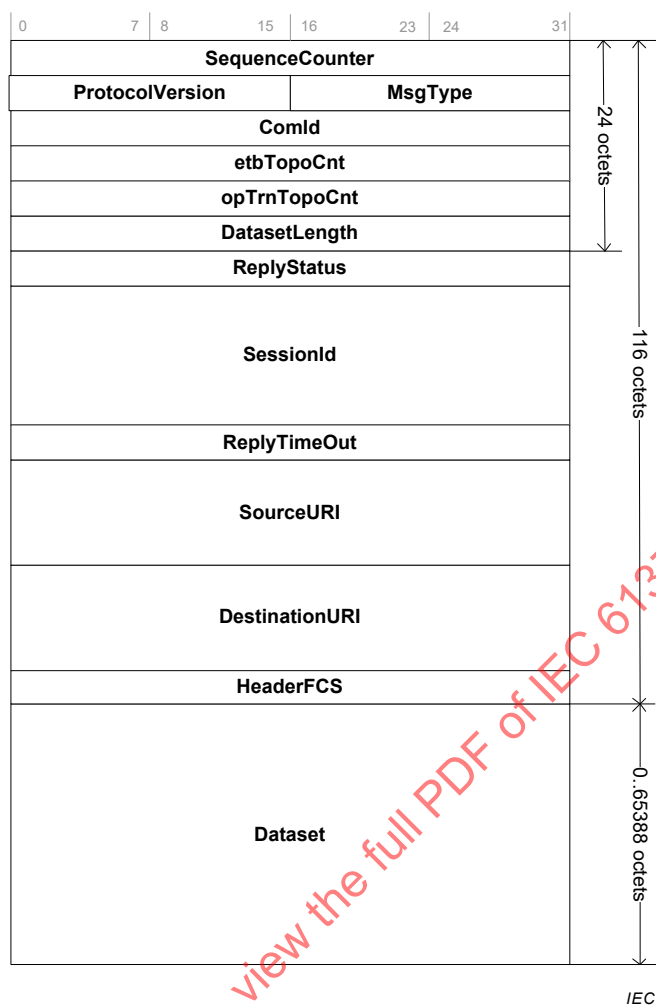


Figure A.19 – MD-PDU

MD_PDU ::= RECORD

```
{
sequenceCounter  UINT32      -- message sequence counter
protocolVersion  VERSION     -- protocol version
msgType          UINT16      -- type of the message
comId            UINT32      -- communication identifier
etbTopoCnt       UINT32      -- etbTopoCnt value
opTrnTopoCnt     UINT32      -- opTrnTopoCnt value
datasetLength    UINT32      -- length of the array 'dataset'
replyStatus      INT32       -- reply status indication
sessionId        UINT32[4]   -- session identifier
replyTimeout     UINT32      -- reply timeout
sourceUri        CHAR[32]    -- source URI (user part)
destinationUri   CHAR[32]    -- destination URI (user part)
headerFcs        UINT32      -- header checksum
dataset          ARRAY[] of  -- user data set
  UINT8
}
```

Table A.18 – MD-PDU parameters

Parameter	Description	Value
sequenceCounter	The sequence counter - shall be incremented with each repetition of the message (return to 0 on overflow) - shall be returned with the reply message - shall start with value: 0	Computed
protocolVersion	The protocol version shall consist of: - Higher significant octet: MainVersion, incremented for incompatible changes, - Lower significant octet: SubVersion, incremented for compatible changes EXAMPLE – '0102'H = protocol version 1.2	Fixed
msgType	Type of the telegram. 'Mn' = Notification (Request without reply) 'Mr' = MD Request with reply 'Mp' = MD Reply without confirmation 'Mq' = MD Reply with confirmation 'Mc' = MD Confirm 'Me' = MD error NOTE Coded in ASCII for better reading with open protocol analyzer tools (e.g. Wireshark)	'4D6E'H ('Mn') '4D72'H ('Mr') '4D70'H ('Mp') '4D71'H ('Mq') '4D63'H ('Mc') '4D65'H ('Me')
comId	Identifier of the user dataset. See also A.5. Unspecified PDU (ComId=0) shall be supported. NOTE source IP address might be changed by NAT, see IEC 61375-2-5	set by user
etbTopoCnt	The topography counter: - shall be used (train addressing) as defined in standard part IEC 61375-2-5 (parameter 'etbTopoCnt') - shall be set by the user - shall be set for all communication over the ETB - shall be set if a valid opTrnTopoCnt is set - optional in all other cases. Shall be set to 0 (= invalid) if not used.	0..2 ³² -1 set by user
opTrnTopoCnt	The topography counter: - shall be used as defined in 5.3.3 - shall be set by the user - shall be set for communication between functions which use information from the operational train directory (e.g. side selective operations) - shall be set when the source device used the operational train directory to retrieve the destination IP address (e.g. resolving of URI 'vcu.leadVeh.anyCst.anyCITrn.ITrn') - optional in all other cases. Shall be set to 0 (= invalid) if not used.	
datasetLength	Length of the user data set in number of octets without padding octets	0 ..65388 Computed
replyStatus	The status value shall be set by the replier to report the execution result of a request message or by the caller sending a confirmation. The execution result is supplied by the replying application and transmitted to the requesting	< 0: NOK 0: OK > 0: user reply status

Parameter	Description	Value
	application in addition to the reply message itself. In case of a TRDP error reply ('Me') the value is supplied by TRDP. • –1 – reserved • –2 – session abort • –3 – no replier instance (at replier side) • –4 – no memory (at replier side) • –5 – no memory (local) • –6 – no reply • –7 – not all replies • –8 – no confirm • –9 – reserved • –10 – sending failed	
sessionId	The session identification: • shall identify a “request-reply” or a “request-reply-confirm” session which is composed of a caller session and a reply session. • shall identify a “request-error” session when the replier’s TRDP layer returns an error message. • shall be computed at caller side and reused at replier (listener) side. • shall be used at caller side to relate a reply or error message to the original request message • shall be used at replier side to identify a retransmission of the request in case the reply message was not received and to identify the confirm message • shall be a UUID according to RFC 4122, time based version	Computed
replyTimeout	The reply timeout value shall be set to define the expected reply time in a request / reply session. Unit: 10^{-6} s (µs) Shall be set to 0 for 'Mn', 'Mp', 'Mc' and 'Me' type telegrams.	1 .. (2³²-1) 0 == infinite time
sourceURI	The source URI: • shall be the used for functional addressing • shall be a null terminated string • filling bytes at the end shall be set to 0 • shall contain only the “user part” without “host part” and “@” • may be an empty string (all 0)	
destinationURI	The destination URI: • shall be used for functional addressing • shall be a null terminated string • filling bytes at the end shall be set to 0 • shall contain only the “user part” without “host part” and “@” • may be an empty string (all 0)	
headerFCS	The header frame check sequence: • shall be calculated for the MD header • shall not include the HeaderFCS itself	Computed according to A.3

Parameter	Description	Value
dataset	The user data. If the user data set length is not a multiple of 4 octets, octets with a value of 0 (zero) shall be appended until a multiple of 4 octets is reached (padding bytes).	

A.7.6 Interaction between application and TRDP layer

A.7.6.1 Service primitives

The TRDP layer shall provide services for message data communication to the TRDP user. The service primitives listed in Table A.19 and Table A.20 shall be provided.

NOTE The service primitives are defined in an abstract way in order to not restrict implementation. The specification of the service interface itself is not in the scope of this part of the standard.

Table A.19 – TRDP service primitives – Caller

Service primitive	Parameters	Direction/Description
MD.request		TRDP user → TRDP layer TRDP user asks to send a request message (notification only or requesting a reply).
	MsgType	- send notification message ('Mn') - send request message ('Mr')
	UserIdentifier	identifier returned with the MD.indication. Used by user to associate the MD.request with the MD.indication
	SessionId	session identifier used and returned by the TRDP layer.
	TransProtocol	UDP or TCP transport layer protocol
	NumRepliers	number of repliers if the request message is sent to known replier(s), e.g. '1' if sent to one replier. Shall be set to '0' if number of repliers is unknown or if notification message is sent.
	etbTopoCnt	as defined in Table A.18
	opTnTopoCnt	as defined in Table A.18
	ComId	as defined in Table A.18
	DatasetLength	as defined in Table A.18
	Dataset	as defined in Table A.18
	SourceURI	as defined in Table A.18
	DestinationURI	as defined in Table A.18
	DestinationIpAddress	IP destination address, generated out of respective URI's using DNS
	ReplyTimeOut	as defined in Table A.18
	MaxNumRetries	maximum number of retries in case NumRepliers == 1. Value range 0..2.
MD.indicate		TRDP layer → TRDP user TRDP layer informs about an event (received data, timeout or error).
	IndType	- reception of a reply message ('Mp') - reception of a query message ('Mq')

Service primitive	Parameters	Direction/Description
		TRDP layer error ('Me')
	UserIdentifier	identifier which was submitted with MD.request
	SessionId	session identifier used by the TRDP layer.
	NoOfRepl	Number of received replies
	etbTopoCnt	etbTopoCnt value obtained from received message. Set to 0 if unknown (error return)
	opTrnTopoCnt	opTrnTopoCnt value obtained from received message. Set to 0 if unknown (error return)
	ComId	as defined in Table A.18
	DatasetLength	as defined in Table A.18
	User Dataset	as defined in Table A.18
	SourceURI	as defined in Table A.18
	DestinationURI	as defined in Table A.18
	SourceIpAddress	IP source address
	DestinationIpAddress	IP destination address
	ReplyTimeOut	as defined in Table A.18
	ReplyStatus	as defined in Table A.18
MD.confirm		TRDP user → TRDP layer TRDP user asks to send a confirm message
	SessionId	session identifier as returned by MD.indication. Used by the TRDP layer to associate the MD.confirm with the session
	ReplyStatus	as defined in Table A.18
MD.abort		TRDP user → TRDP layer TRDP user asks to abort an open session e.g because of a TCN inauguration
	SessionId	session identifier used by the TRDP layer.

Table A.20 – TRDP service primitives – Replier

Service primitive	Parameters	Direction/Description
MD.addListener		TRDP user → TRDP layer TRDP user asks the TRDP layer to prepare for reception of notification/request messages (e.g. by joining defined MC groups and providing telegram buffer). Filters can be set for ComId, SourceIpAddress, SourceIpAddress range, DestinationIpAddress, SourceURI and DestinationURI.
	Handle	Handle for this listener, returned by the TRDP layer
	ComId	Notification/request with that ComId(s) to be accepted
	etbTopoCnt	actual etbTopoCnt value
	opTrnTopoCnt	actual opTrnTopoCnt value
	SourceURI[]	as defined in Table A.18, only notification/request from that source URI(s) to be accepted, only user part taken into account, host part only in case of group addressing See also A.7.6.3
	DestinationURI	as defined in Table A.18, only notification/request with that destination URI to be accepted, only user part taken

Service primitive	Parameters	Direction/Description
		into account, host part only in case of group addressing See also A.7.6.3
	SourcelpAddress	IP source address, generated out of respective URI's using DNS Defines the lower IP address in case of an IP address range (see 5.4.4.6.2)
	SourcelpAddress2	Defines the upper IP address in case of an IP address range (see 5.4.4.6.2) IP source address, generated out of respective URI's using DNS Set to 0 if not used
	DestinationIpAddress	IP destination address, generated out of respective URI's using DNS
	UserIdentifier	Identifier to be returned with MD.indicate
MD.updateListener		TRDP user → TRDP layer TRDP user asks the TRDP layer to update the listener parameters and eventually MC group memberships after a change of the train topology.
	Handle	Handle for this listener
	etbTopoCnt	actual etbTopoCnt value
	opTrnTopoCnt	actual opTrnTopoCnt value
	SourcelpAddress	IP source address, generated out of respective URI's using DNS Defines the lower IP address in case of an IP address range (see 5.4.4.6.2)
	SourcelpAddress2	Defines the upper IP address in case of an IP address range (see 5.4.4.6.2) IP source address, generated out of respective URI's using DNS Set to 0 if not used
	DestinationIpAddress	IP destination address, generated out of respective URI's using DNS
MD.remListener		TRDP user → TRDP layer TRDP user asks the TRDP layer to remove a listener
	Handle	Handle returned by the TRDP layer in MD.addListener
MD.indicate		TRDP layer → TRDP user TRDP layer informs about an event
	IndType	- reception of a request message ('Mr') - reception of a notification message ('Mn') - TRDP layer error ('Me')
	UserIdentifier	identifier which was submitted with MD.addListener
	SessionId	Session identifier of the session created by the TRDP layer
	etbTopoCnt	etbTopoCnt value obtained from received message. Set to 0 if unknown (error return)
	opTrnTopoCnt	opTrnTopoCnt value obtained from received message.

Service primitive	Parameters	Direction/Description
		Set to 0 if unknown (error return)
	ComId	opTrnTopoCnt value obtained from received message. Set to 0 if unknown (error return)
	DatasetLength	as defined in Table A.18
	Dataset	as defined in Table A.18
	SourceURI	as defined in Table A.18
	DestinationURI	as defined in Table A.18
	SourceIpAddress	IP source address
	DestinationIpAddress	IP destination address. Can also be an IP multicast address.
	ReplyTimeOut	as defined in Table A.18
	ReplyStatus	as defined in Table A.18
MD.reply		TRDP user → TRDP layer TRDP user asks to send a reply message
	MsgType	- send reply message ('Mp') - send query message ('Mq')
	SessionId	Session identifier as indicated with MD.indication. Used by the TRDP layer to associate the MD.reply with the reply session
	ComId	as defined in Table A.18
	DatasetLength	as defined in Table A.18
	UserDataset	as defined in Table A.18
	SourceURI	as defined in Table A.18
	SourceIpAddress	IP source address, generated out of respective URI's using DNS
	ReplyTimeOut	as defined in Table A.18
	ReplyStatus	as defined in Table A.18
MD.release		TRDP layer → TRDP user TRDP layer informs about a received confirm message, related timeout or error.
	MsgType	- reception of a confirm message ('Mc') - TRDP layer error ('Me')
	SessionId	Session identifier used by the TRDP layer.
	etbTopoCnt	etbTopoCnt value obtained from received message. Set to 0 if unknown (error return)
	opTrnTopoCnt	opTrnTopoCnt value obtained from received message. Set to 0 if unknown (error return)
	ComId	as defined in Table A.18
	SourceURI	as defined in Table A.18
	SourceIpAddress	IP source address
	ReplyStatus	as defined in Table A.18

A.7.6.2 Interaction sequence

The sequence chart in Figure A.20 defines the sequence of the service primitives.

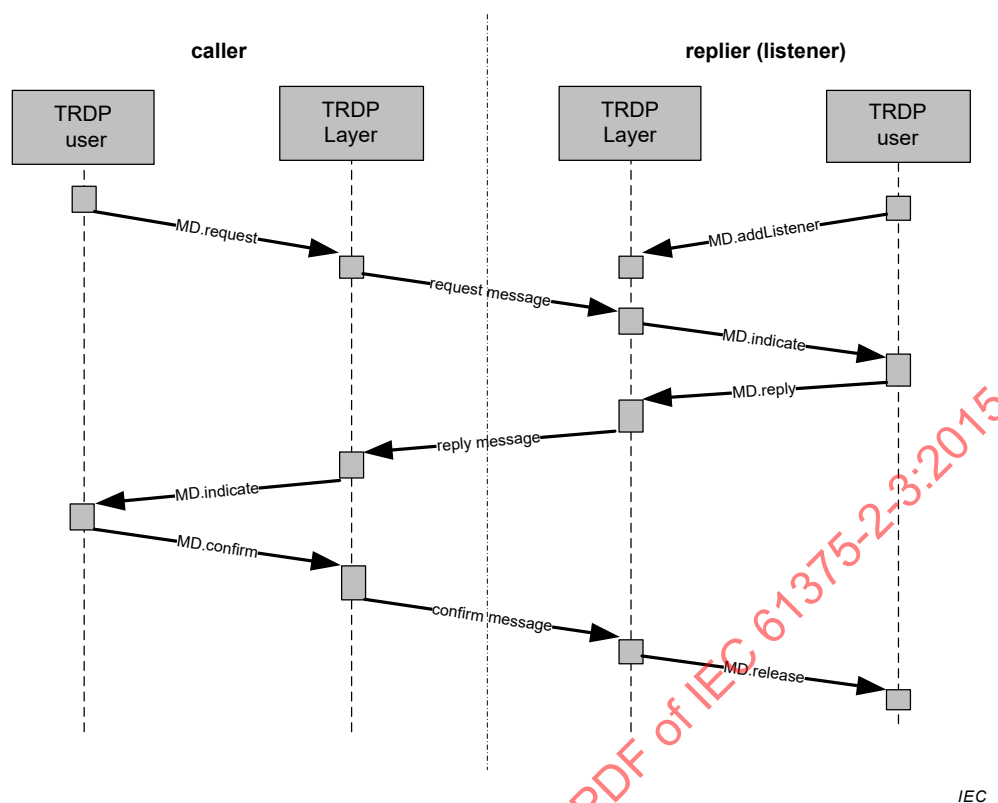


Figure A.20 – Interaction sequence chart

A.7.6.3 Filtering rules

The service primitive MD.addListener allows to optionally define SourceURI and DestinationURI for filtering received MD telegrams. For the filtering, the following rules shall apply:

- An empty DestinationURI in a MD telegrams shall be received by a listener regardless of any specified DestinationURI.
- An empty SourceURI in a MD telegrams has no special meaning.
- An empty DestinationURI filter entry in MD.addListener service primitive means that the listener shall receive MD telegrams regardless of any defined DestinationURI in the MD telegrams.
- An empty SourceURI filter entry in MD.addListener service primitive means that the listener shall receive MD telegrams regardless of any defined SourceURI in the MD telegrams.
- Any two listeners shall have a disjunctive set of filter parameters ComId, SourceURI, DestinationURI, SourceIpAddress and DestinationIpAddress. If empty filter parameters are used, any two listeners have to differ in at least one non-empty filter parameter.
- MD telegrams with empty DestinationURI may be received by several listeners with different DestinationURI filters on one device. TRDP has to handle the duplication of received data for the different listeners.

NOTE A MD telegram sent to an unicast IP destination address with empty DestinationURI is a potential multicast – multiple replies may be received.

A.7.7 Topography counter check

Before sending a telegram, the topography counters of the telegram shall be checked against the actual topography counters to ensure that the sending application shares the actual train backbone view and operational train view.

After reception of a telegram, the topography counter values shall be checked to ensure that caller and replier share an identical train backbone view and operational train view. For this check, the topography counter values received are compared first with the actual topography counter values and then with those submitted in the MD.addListener service primitive.

The topography counter check is passed if at least one of the cases listed in is true.

Table A.21 – Topography counter check

Caller	Actual topography counter values		Topography counter values of the telegram to be sent	
Replier	Actual topography counter values		Topography counter values of the received telegram	
	Topography counter values of the received telegram		Locally stored topography counter values submitted with adding a listener	
Case	etbTopoCnt	opTrnTopoCnt	etbTopoCnt	opTrnTopoCnt
1	any	any	0	0
2	0	equal	0	equal
3	equal	any	equal	0
4	equal	equal	equal	equal
5(1)	0	0	equal	equal
(1) Case applies only for replier comparing topography counter values of the received telegram with the locally stored topography counter values submitted with adding the listener. Used to receive local telegrams. Key: 0 = topography counter value set to 0 (don't care) any = topography counter has any value in the range $0..(2^{32}-1)$ equal = local and received topography counter values are valid ($1..(2^{32}-1)$) and identical				

A.7.8 MD protocol state machine

A.7.8.1 TRDP Layer – Caller

If a TRDP user (caller) requests sending a message, its topography counters shall be checked against the actual topography counters. If none of the cases listed in Table A.21 is TRUE, the message shall be discarded.

If a TRDP user (caller) requests sending a notification, a notification message (MsgType 'Mn') shall be sent. The session identifier of the notification message shall be set to 0.

If a TRDP user (caller) requests sending a request message, a caller session shall be opened and a request message (MsgType 'Mr') shall be sent.

To ensure that the session identifier is unique, each request-reply/request-reply-confirm session shall be identified by a 16 byte UUID according to RFC 4122, time based version. The session identifier is calculated at caller side in the TRDP layer, transmitted within each message and used at caller and replier side to identify the related caller and replier session.

A timeout value for the reply message(s) shall be defined by the TRDP user (caller).

NOTE If retries are selected, the timeout at application layer can be up to $(\text{maxNumRetries} + 1) \times \text{replyTimeout}$.

During the caller session, the TRDP layer shall wait for incoming reply messages (MsgType 'Mp', 'Mq' or 'Me'). All incoming reply messages related to the caller session (identified by the received session id) shall be given immediately to the TRDP user.

If the number of expected incoming replies is reached and there are no more outstanding confirmations from TRDP user (caller), the caller session shall be closed.

If the replier requests a confirmation (MsgType 'Mq'), the TRDP layer shall start a timer with the given confirm timeout time of the reply and wait for the confirmation from the TRDP user (caller).

The confirmation shall use as destination URI the source URI received in the reply message.

After getting the confirmation from TRDP user (caller), the TRDP layer shall send a confirmation message (MsgType 'Mc') to the replier.

A confirmation message shall not contain user data and shall be sent only as unicast.

The TRDP user (caller) shall take care to provide the confirmation in time as indicated by the ReplyTimeOut parameter of the reply.

If the number of incoming replies is reached and the confirm timeout timer expires while waiting for outstanding confirmations from TRDP user (caller), the TRDP user (caller) shall be notified that confirmations are missing and the caller session shall be closed.

If the reply timeout timer of the caller session expires because of a missing reply and the number of expected repliers is 1, the TRDP layer shall, depending on the given parameter value 'MaxNumRetries', repeat the request up to two times before it notifies the TRDP user (caller) about the missing reply and closes the caller session.

If the reply timeout timer expires and the number of expected repliers is greater than 1 and less than the expected replies have been received, the TRDP Layer shall notify its TRDP user (caller) about the timeout and the number of missing replies. The caller session shall be closed when all requested confirmations are sent or the confirmation timeout timer has timed out.

If the number of repliers is not known (parameter NoOfRepliers = 0 and the reply timeout timer expires, the total number of replies shall be indicated to the TRDP user (caller) and the caller session shall be closed.

There shall be no retransmission of request messages if TCP is used. If for TCP retransmission or continuing of the transmission is wished, this shall be done in a higher level service or in the application. TRDP shall provide the interface to continue a transmission after the connection was lost.

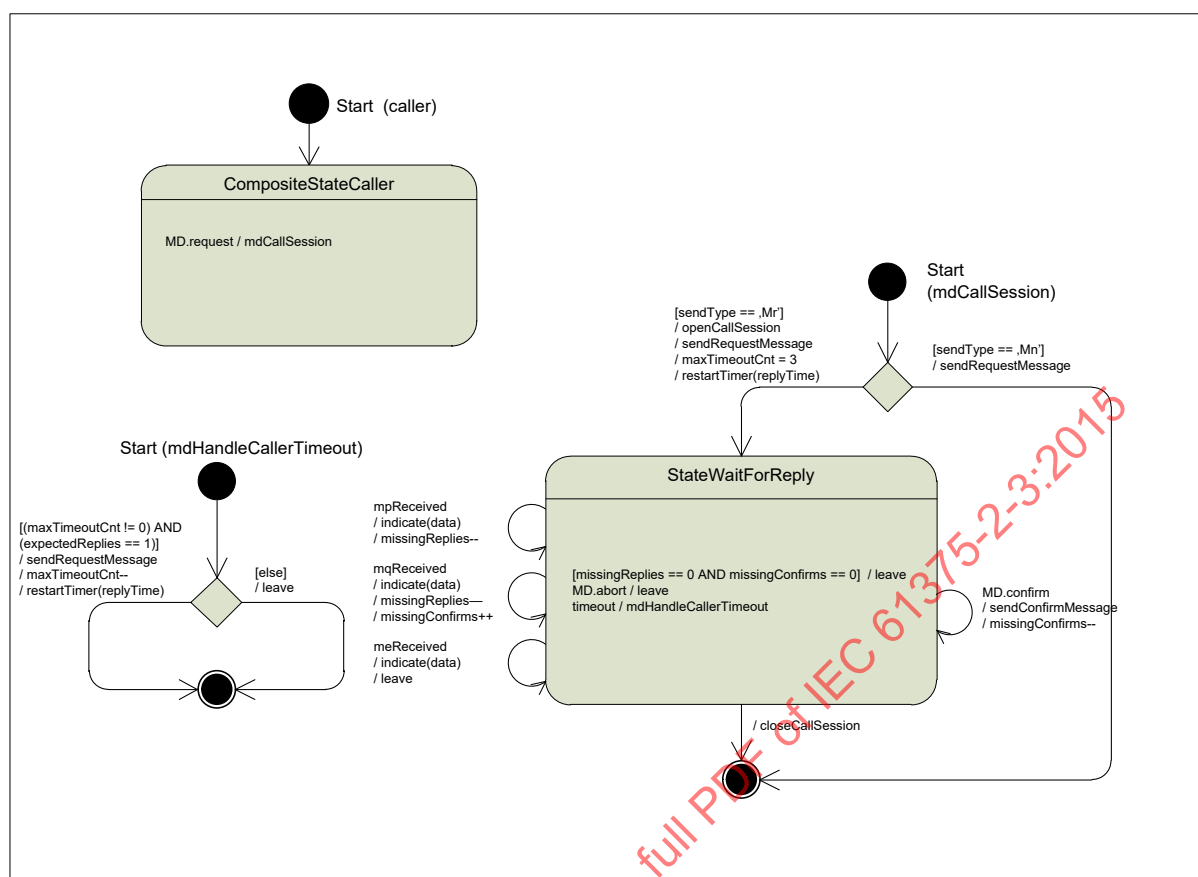
If the TRDP layer receives a reply message (MsgType 'Mp', 'Mq' or 'Me') without having opened a caller session for the indicated session id, the message shall be discarded.

The topography counter values obtained during caller session opening shall be used throughout the session (for request and confirmation messages).

If a TRDP user receives a reply message with topography counter values different to the expected ones (see Table A.5), the message shall be discarded.

If a TRDP user aborts an (open) session (e.g. after a train topology change), running timers shall be stopped, the session identifier shall be destroyed and the session shall be closed.

The behavior of the caller's TRDP layer is depicted in Figure A.21 with explanations given in Table A.22 to Table A.25.



IEC

Figure A.21 – TRDP layer MD caller state chart

Table A.22 – MD caller state diagram – triggers

Triggers	Description
MD.request	User called MD.request service primitive
mpReceived	Valid reply without requested confirmation received, see A.7.8.3
mqReceived	Valid reply with requested confirmation received, see A.7.8.3
meReceived	Valid MD error message received, see A.7.8.3
MD.confirm	User called MD.confirm service primitive
MD.abort	User called MD.abort service primitive to close the open session (e.g. after a change of topography counters).
timeout	Reply timeout (no reply message received within replyTimeout)

Table A.23 – MD caller state diagram – guards

Guards	Description
sendType	Type of the MD request telegram as requested by the user in the MD.request service primitive
expectedReplies	Number of expected replies as indicated in MD.request service primitive (parameter 'NoOfRepliers')
missingReplies	Counter for missing replies
missingConfirms	Counter for missing confirms
maxTimeoutCnt	Counter for timeouts. Shall be decremented with each timeout.

Guards	Description
checkMessage	Check received message for correct CRC, protocol version, protocol type, length and topography counter (at least one of the cases listed in Table A.21 for the topography counters is fulfilled).
checkTopoCounts	Check the topology counters of the received message against the actual topography counters. At least one of the cases listed in Table A.21 for the topography counters is fulfilled.

Table A.24 – MD caller state diagram – actions

Actions	Description
mdCallSession	Handles the states of a caller session.
openCallSession	Creates a caller session
closeCallSession	Closes the caller session
mdGetCallSessionState	Returns the state of the caller session for a given sessionId
sendRequestMessage	Sends the request message. If sending the request message is repeated after timeout telegram parameter 'SequenceCounter' shall be incremented.
sendConfirmMessage	Sends the confirm message
Indicate	Inform user about a received reply or error message by invoking MD.indicate service primitive
restartTimer(replyTime)	Start a timer with the user defined replyTimeout value (MD.request service primitive)
discard	discard received message without further processing

Table A.25 – MD caller state diagram – states

States	Description
CompositeStateCaller	Composite state of the caller
StateWaitForReply	State of a caller session that has sent out a request and is now waiting for the reply/replies and optional requested confirmations from the application

A.7.8.2 TRDP Layer – Replier (Listener)

Each TRDP user (replier) that wants to receive MD shall register as listener for MD sent to a specific URI (multicast or unicast) or for MD of a specific ComId.

All incoming messages shall be checked against the actual topography counters. Messages not fitting to the actual topography counters (see Table A.21) shall be discarded.

All incoming messages shall be checked against registered listeners. Any message to a not registered listener or to a listener expecting another value of the topography counters shall be discarded.

In case of a unicast request message to a not registered listener or to a listener expecting another value of the topography counters an error message (MsgType = 'Me') shall be sent indicating the error with replyStatus == -3 (no replier instance).

If a notification message (MsgType 'Mn') is received, the message shall be passed to the related listener.

If a request message (MsgType 'Mr') is received, a reply session shall be opened using the received session id, a timeout timer with the received reply timeout shall be started, the message shall be passed to the related listener and the TRDP layer shall wait for the MD.reply of the listener.

If, due to the lack of resources, a reply session can't be opened, the received message shall be discarded and an error message (MsgType = 'Me') shall be sent, indicating the error with replyStatus == -4 (no memory (at replier side)).

If a request message (MsgType 'Mr') is received for an open reply session with the received session id while waiting for the MD.reply of the listener, the request message shall be discarded.

After receiving the MD.reply from the TRDP user (listener), the TRDP layer shall send, depending on MD.reply of the listener, a reply message without confirmation (MsgType = 'Mp') or a reply message with confirmation (MsgType = 'Mq') to the caller.

A reply message shall be sent only as unicast using source IP address and source URI of the related request message as destination IP address and destination URI.

The MD.reply of a listener shall use as source IP address the IP address of the listener and as source URI the unique URI (user part) of the listener.

If a reply message without confirmation (MsgType = 'Mp') was sent, the reply session shall be closed.

If a reply message with confirmation (MsgType = 'Mq') was sent, the timeout timer of the reply session shall be restarted using the given confirm timeout value.

If a request message (MsgType 'Mr') is received with a different sequence counter for an already open reply session, after the TRDP layer has received the MD.reply from the TRDP user (replier), the TRDP layer shall repeat sending the reply message (MsgType = 'Mq') to the caller with an incremented sequence counter. After resending the reply message the timeout timer of the reply session shall be restarted using the given confirm timeout value.

If a request message (MsgType 'Mr') is received for an already open reply session and the same sequence counter like received before, the request message shall be discarded.

If a confirmation message (MsgType 'Mc') is received for an open reply session with the received session id, the related listener shall be notified and the session shall be closed.

If a confirmation message (MsgType 'Mc') is received without having an open reply session with the received session id, the message shall be discarded.

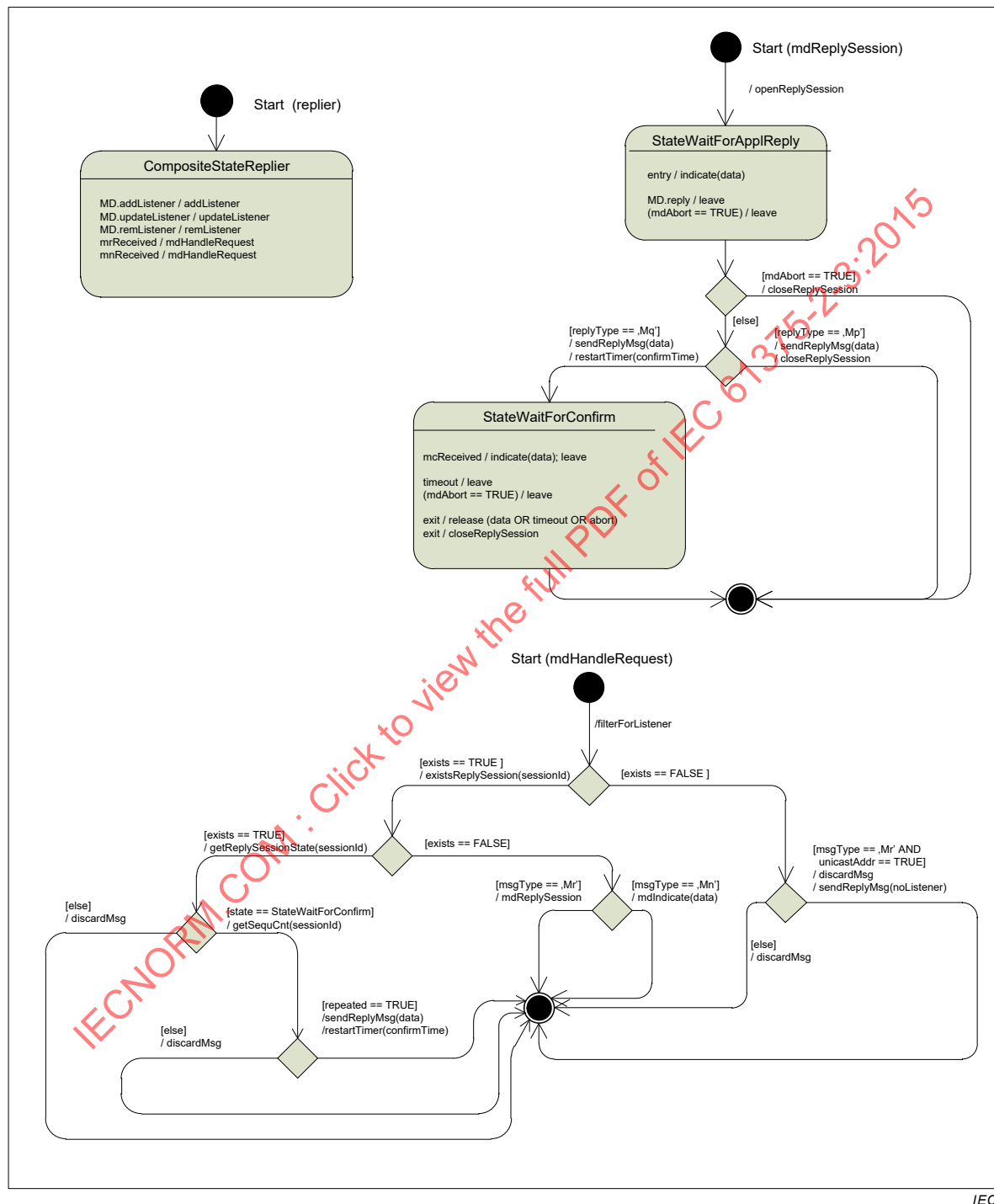
If the timeout timer of a reply session expires, the reply session shall be closed. In case of a timeout because of a missing confirmation or a missing MD.reply of the listener, the listener shall be notified.

The topography counter values obtained in an MD.addListener service primitive shall be used for upcoming reply sessions unless they are updated by the user (MD.updateListener service primitive).

If a TRDP user receives a request message with topography counter values different to the expected ones (see Table A.21), the message shall be discarded.

If a TRDP user removes or updates a listener (e.g. after a train topology change) during an open session, running timers shall be stopped, the session identifier shall be destroyed and the session shall be closed.

The behavior of the repliers's TRDP layer is depicted in Figure A.22 with explanations given in Table A.26 until Table A.29.



IEC

Figure A.22 – TRDP layer MD replier state chart

Table A.26 – MD replier state diagram – triggers

Triggers	Description
MD.addListener	User called MD.addListener service primitive
MD.updateListener	User called MD.updateListener service primitive
MD.remListener	User called MD.remListener service primitive
mrReceived	Valid request received, see A.7.8.3
mnReceived	Valid notification received, see A.7.8.3
mcReceived	Valid confirmation received, see A.7.8.3
MD.Reply	User called MD.reply service primitive
timeout	Confirm timeout (no confirm message received within replyTimeout)

Table A.27 – MD replier state diagram – guards

Guards	Description
existsReplySession (sessionId)	The reply session for a given session id exists
existsListener	A listener for the given request with fitting topography counters (see Table A.21) exists.
replyType	Type of MD reply telegram as requested by the user in the MD.reply service primitive
stateReplySession(sessionId)	replier session state for a given session id
repeated	Indication of request message repetition
unicastAddr	destination address is a unicast address
checkMessage	Check received message for correct CRC, protocol version, protocol type, length and topography counter (at least one of the cases listed in Table A.21 for the topography counters is fulfilled).
checkTopoCounts	Check the topology counters of the received message against the actual topography counters (at least one of the cases listed in Table A.21 for the topography counters is fulfilled).

Table A.28 – MD replier state diagram – actions

Actions	Description
mdReplySession	Handles the states of a reply session.
mdHandleRequest	Handles the received request message.
openReplySession	Creates a reply session
closeReplySession	Closes the reply session
sendReplyMessage	Sends the reply message
indicate	Inform user about a received request message or a received confirmation message by invoking MD.indicate service primitive
release	Inform user about a received confirm message or a timeout by invoking MD.release service primitive
addListener	Add a listener for given filter criteria mdAbort = FALSE
updateListener	Update listener parameters mdAbort = TRUE
remListener	Remove a listener mdAbort = TRUE
getSequCnt(sessionId)	Checks if the request message has been repeated by the caller (is

Actions	Description
	the case if the sequence counter of the previous received request message is different to the actual received value) repeated == TRUE: sequence counters different repeated == FALSE: sequence counters equal
discardMsg	Discard the received message
restartTimer(confirmTime)	Start a timer with the user defined replyTimeout value (MD.reply service primitive)
sendReplyMsg(param)	Send reply message Param = 'data': – Reply type: 'Mq' OR 'Mp' (user defined) – Reply status: user defined – Reply ComId: user defined – Dataset: user defined
sendReplyErr(replyStatus)	Send error reply message Param = 'noListener' – Reply type: 'Me' – Reply status: replyStatus – Reply ComId: 0 – Dataset: empty

Table A.29 – MD replier state diagram – states

States	Description
CompositeStateReplier	Composite state of the replier
StateWaitForApplReply	State of a reply session that has sent a received request message requesting a reply to the application and is now waiting for the MD.reply of the application.
StateWaitForConfirm	State of a reply session that has sent out a reply requesting a confirmation and is now waiting for the confirmation.

A.7.8.3 Handling of received messages (action mdHandleReceived)

The state diagram shown in Figure A.23 with explanations given in Table A.30 until Table A.33 describes the handling of received message data telegrams

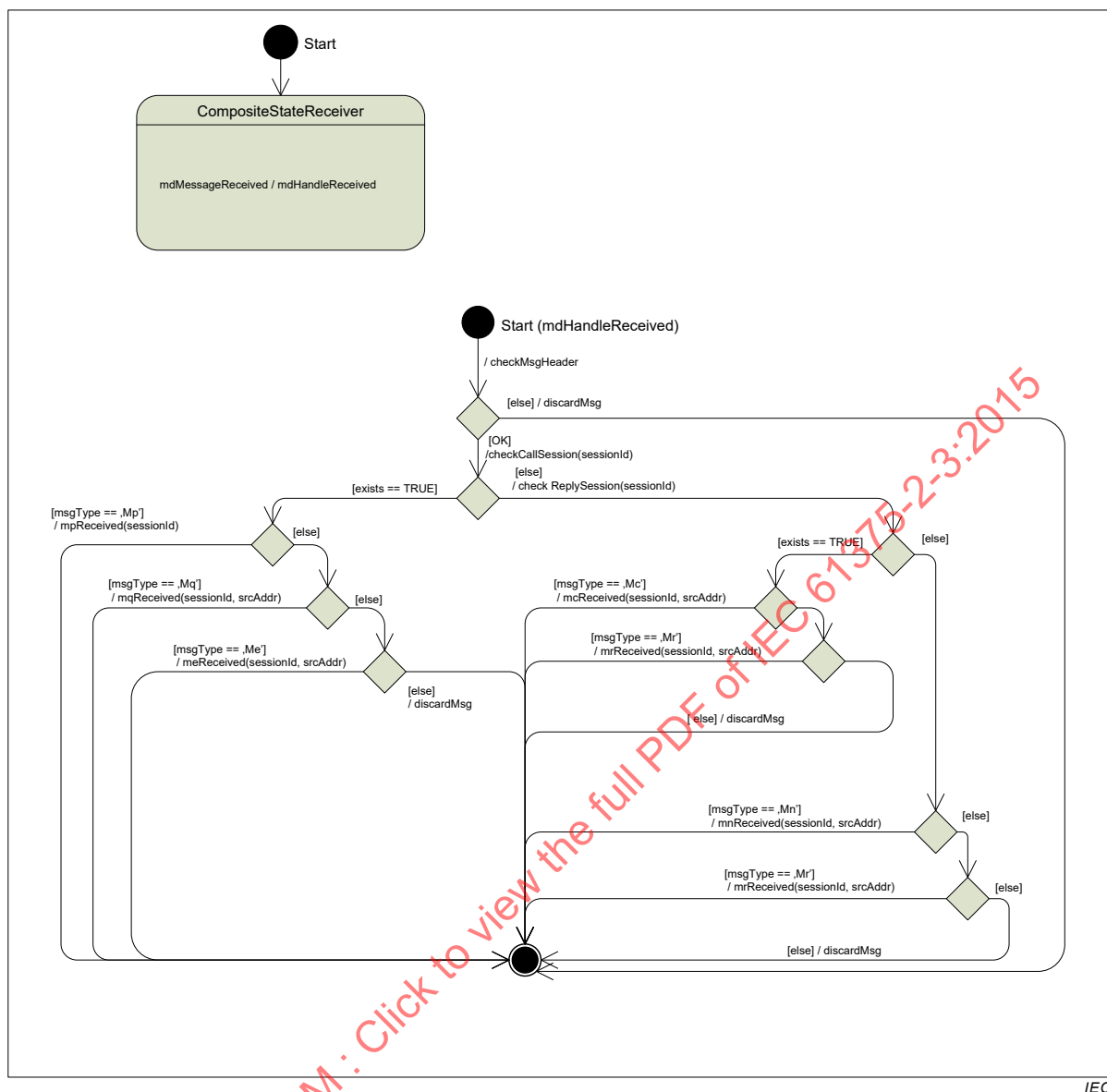


Figure A.23 – TRDP Layer MD telegram reception

Table A.30 – MD receiver state diagram – triggers

Triggers	Description
mdMessageReceived	A MD telegram has been received

Table A.31 – MD receiver state diagram – guards

Guards	Description
msgType	Message telegram type as defined in Table A.18

Table A.32 – MD receiver state diagram – actions

Actions	Description
checkCallSession	Check if a caller session to the given reply exists If a caller session exists, check that at least one of the cases listed in Table A.5 for the topography counters is fulfilled. exists = TRUE: both checks passed
checkReplySession	Check if a reply session to the given request or confirmation exists exists = TRUE: check positive NOTE There may be multiple reply sessions
checkMsgHeader	Check the MD telegram header for FCS, version and sequence counter and discard incorrect telegram
mpReceived	Trigger reception of 'Mn' type MD telegram
mqReceived	Trigger reception of 'Mq' type MD telegram
mcReceived	Trigger reception of 'Mc' type MD telegram
mrReceived	Trigger reception of 'Mr' type MD telegram
mnReceived	Trigger reception of 'Mn' type MD telegram
discardMsg	Discard the message

Table A.33 – MD receiver state diagram – states

States	Description
CompositeStateReceiver	Composite state of the receiver

A.7.9 TCP Connection Handling

This subclause defines the handling of TCP connections for TRDP message data transmission.

Using TCP for data transmission requires an established TCP connection between the caller and the replier device. Thus the first request to a specified device shall open a connection if there was not yet a connection established before.

As long as a TCP connection to another device in the required direction is open, it should be reused for all further calls to this device.

The caller shall close an existing TCP connection (active end) in the following cases:

- A signal that the TCP connection will be closed has been received.
- TRDP shut down or re-initialization.
- A timeout occurred because the TCP connection has not been used for a defined time.

The replier shall use the connection opened by the caller.

The replier shall close an existing TCP connection (passive end) in the following cases:

- A signal that the TCP connection will be closed has been received.
- TRDP shut down or re-initialization.
- Another TCP connection was opened from the same caller device and the old connection is not used anymore for a defined time.

A.8 Message data echo server (option)

End devices with TRDP message data support can provide a message data echo function. A system that wants to test communication with an end device sends an echo request (ERQ) to the end device by using the MD.request service primitive. The end device activates an echo function that creates an echo reply (ERP), copies data from the ERQ to the ERP and then returns it to the sender.

The Message Data Echo should be implemented in the TRDP layer.

When message data arrive with the echo ComId (see Table A.2) an echo message is created and returned to the source. The user data of the returned message shall be a copy of the received user data.

IECNORM.COM : Click to view the full PDF of IEC 61375-2-3:2015

Annex B

(normative)

Safe Data Transmission (SDTv2)

B.1 General

This annex defines an extension to the communication profile for the transmission of safety related (vital) data over ETB.

B.2 Overview of SDTv2 (informal)

SDTv2 provides a safe communication path between a source of safety related (vital) data (SDSRC) and one or several sinks of those data (SDSINK). This safe communication path is called "SDTv2 channel". Figure B.1 provides a logical model of such a SDTv2 channel. The SDTv2 channel starts and ends at safe applications, and covers the entire path along from the SDSRC application to the SDSINK application(s). This includes the train communication network (TCN) as well as the communication layers related to the TCN which reside on the safe end devices (ED-S) hosting the safe application, and which both are considered not trustable from a safety point of view. A safe end device may contain one or a number of SDSRC or SDSINK, respectively. SDSRC and SDSINK itself can be described as a composition of the safe application and the SDTv2 protocol layer. The SDTv2 protocol layer provides two interfaces: the communication channel interface and the SDTv2 application interface. The communication channel interface is defined by the communication technology underneath (TRDP). This is the interface where SDTv2 protocol data units, called VDPs (= Vital Data Packets) are sent to or received from the TCN. The SDTv2 application interface, which is product specific, is the place where safety related process data are put to or get from the application. The SDTv2 Layer at SDSRC side has mainly the task to add protocol information, in such a way forming a VDP, which is necessary for a safe transfer of those data over the network, prior to sending. On receiver side, the SDTv2 layer validates received VDPs, and if validation is successful, contained vital data are exposed in the SDTv2 application interface.

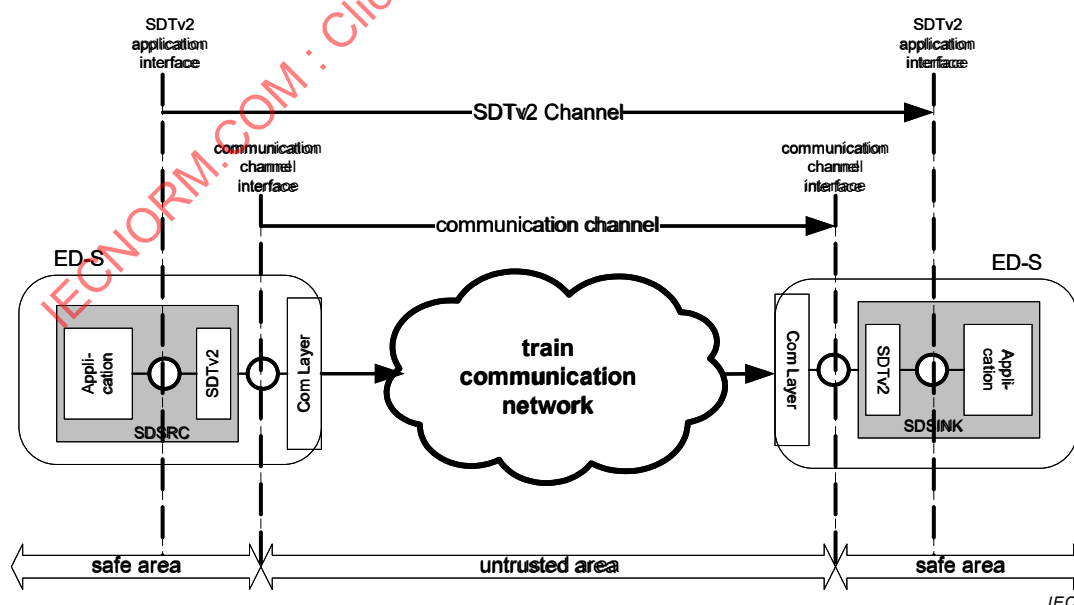


Figure B.1 – SDTV2 Channel

VDPs are specified for cyclic transmission, e.g. transported within the payload of TRDP process data.

B.3 Safety functional requirements

The following requirements have been mandatory for the development of the SDTv2 protocol:

- a) Safety communication and standard (regular) communication shall be independent. However, standard devices and safety devices shall be able to use the same communication channel.
- b) Safety communication shall be suitable for Safety Integrity Level SIL2 for continuous mode of operation (see IEC 61508-1).
- c) Safety communication shall use a single-channel communication system. Redundancy may only be used optionally for increased availability.
- d) Implementation of the safe transmission protocol shall be restricted to the communication end devices.
- e) Due to the dynamic nature of train compositions with a varying number of consists, a 1:n communication relationship between the safe data source and safe data sinks has to be supported.
- f) The transmission duration times shall be monitored.
- g) Environmental conditions shall be according to general railway requirements, mainly IEC 60571, if there are no particular product standards.
- h) Transmission equipment such as controllers, ASICs, Ethernet switch cores, cables, couplers, etc., shall remain unmodified (black channel). The safety functions shall be above OSI layer 7 (i.e. profile, no standard protocol changes or enhancements).
- i) The safety communication shall not reduce the permitted number of devices.
- j) The safety communication shall support data exchange over the ETB in dynamically changing train compositions.

B.4 Safety measures

The safety measures listed in Table B.1 for mastering possible transmission errors are one significant component of the SDTv2.

The safety measures shall be processed and monitored within one safety device.

Table B.1 – Deployed measures to communication errors

Communication Error	Safety measures						
	Safe Sequence Counter	Sink-time supervision	Safety Code	Guard time	Source Identifier (SID)	Latency Monitoring	Channel Monitoring
Corruption			X				
Unintended repetition	X						
Incorrect sequence	X						
Loss		X					
Unacceptable delay		X				X	
Insertion	X				X		
Masquerade	X				X		
Addressing					X		
Redundancy failure (active redundant sources)				X			
Unacceptable communication channel error rate		X	X				X

B.5 Operational states of the SDTv2 channel

A SDTv2 channel can be in two basic operational states (see Figure B.2):

State RegularCommunication: In this state, transmitted vital process data cannot be considered to be safe

State SafeCommunication: In this state, transmitted vital process data can be considered to be safe

The conditions under which state changes occur are specified in the following subclauses.

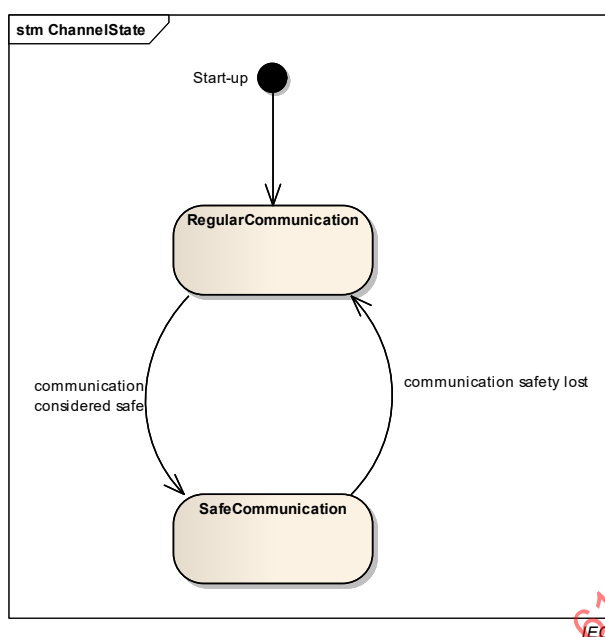


Figure B.2 – SDTv2 Channel States

B.6 Data presentation

All data within SDTv2 shall be transmitted in big-endian format (most significant octet of a data item first).

All data structures used within SDTv2 shall be naturally aligned (data items stored at offset address which is a multiple of their size).

Variable length data structures (open arrays, records) shall not be used.

The VDP data elements of a structured type (record, sequence) shall be arranged in the order they are declared.

B.7 SC-32

SC-32 defines a cyclic redundancy code used for the computation of SID and SafetyCode. The polynomial to be used is '1F4ACFB13' as defined in IEC 61784-3-3. The algorithm in C programming language notation depicted in Figure B.3 (and defined in IEC 61784-3-3) for SC-32 computation should be used, see also Figure B.4.


```
/* *****  
 * GLOBAL FUNCTION DEFINITIONS  
 */  
  
/**  
 * @internal  
 * Calculates and returns a 32-bit CRC.  
 *  
 * @param buf    Input buffer  
 * @param len    Length of input buffer  
 * @param crc    Initial (seed) value for the CRC calculation  
 *  
 * @return Calculated CRC value  
 */  
static UNSIGNED32 sdt_crc32(const UNSIGNED8 buf[], UNSIGNED32 len, UNSIGNED32 crc)  
{  
    UNSIGNED32 i;  
  
    for (i = 0; i < len; i++)  
    {  
        crc = crctab32[((UNSIGNED32) (crc >> 24) ^ buf[i]) & 0xff] ^ (crc << 8);  
    }  
  
    return crc;  
}
```

Figure B.3 – SC-32 Computation

IECNORM.COM : Click to view the full PDF of IEC 61375-2-3:2015

```

/*
 * Table for 32-bit CRC calculation according to IEC 61784-3-3 (2007)
 * CRC of the string "123456789" is 0x1697d06a
 */
static const UNSIGNED32 crc32tab[256]={
    0x00000000U, 0xF4ACFB13U, 0x1DF50D35U, 0xE959F626U,
    0x3BEA1A6AU, 0xCF46E179U, 0x261F175FU, 0xD2B3EC4CU,
    0x77D434D4U, 0x8378CFC7U, 0x6A2139E1U, 0x9E8DC2F2U,
    0x4C3E2EBEU, 0xB892D5ADU, 0x51CB238BU, 0xA567D898U,
    0xEFA869A8U, 0x1B0492BBU, 0xF25D649DU, 0x06F19F8EU,
    0xD44273C2U, 0x20EE88D1U, 0xC9B77EF7U, 0x3D1B85E4U,
    0x987C5D7CU, 0x6CD0A66FU, 0x85895049U, 0x7125AB5AU,
    0xA3964716U, 0x573ABC05U, 0xBE634A23U, 0x4ACFB130U,
    0x2BFC2843U, 0xDF50D350U, 0x36092576U, 0xC2A5DE65U,
    0x10163229U, 0xE4BAC93AU, 0x0DE33F1CU, 0xF94FC40FU,
    0x5C281C97U, 0xA884E784U, 0x41DD11A2U, 0xB571EAB1U,
    0x67C206F9U, 0x936EFDEEU, 0x7A370BC8U, 0x8E9BF0DBU,
    0xC45441EBU, 0x30F8BAF8U, 0xD9A14CDEU, 0x2D0DB7CDU,
    0xFFBE5B81U, 0x0B12A092U, 0xE24B56B4U, 0x16E7ADA7U,
    0xB380753FU, 0x472C8E2CU, 0xAE75780AU, 0x5AD98319U,
    0x886A6F55U, 0x7CC69446U, 0x959F6260U, 0x61339973U,
    0x57F85086U, 0xA354AB95U, 0x4A0D5DB3U, 0xBEA1A6A0U,
    0x6C124AECU, 0x98BEB1FFU, 0x71E747D9U, 0x854BBCCAU,
    0x202C6452U, 0xD4809F41U, 0x3DD96967U, 0xC9759274U,
    0x1BC67E38U, 0xEF6A852BU, 0x0633730DU, 0xF29F881EU,
    0xB850392EU, 0x4CFC23DU, 0xA5A5341BU, 0x5109CF08U,
    0x83BA2344U, 0x7716D857U, 0x9E4F2E71U, 0x6AE3D562U,
    0xCF840DFAU, 0x3B28F6E9U, 0xD27100CFU, 0x26DDFBD9U,
    0xF46E1790U, 0x00C2EC83U, 0xE99B1AA5U, 0x1D37E1B6U,
    0x7C0478C5U, 0x88A883D6U, 0x61F175F0U, 0x955D8EE3U,
    0x47EE62AFU, 0xB34299BCU, 0x5A1B6F9AU, 0xAEB79489U,
    0xBD04C11U, 0xFF7CB702U, 0x16254124U, 0xE289BA37U,
    0x303A567BU, 0xC496AD68U, 0x2DCF5B4EU, 0xD963A05DU,
    0x93AC116DU, 0x6700EA7EU, 0x8E591C58U, 0x7AF5E74BU,
    0xA8460B07U, 0x5CEAF014U, 0xB5B30632U, 0x411FFD21U,
    0xE47825B9U, 0x10D4DEAAU, 0xF98D288CU, 0x0D21D39FU,
    0xDF923FD3U, 0x2B3EC4C0U, 0xC26732E6U, 0x36CBC9F5U,
    0xAFF0A10CU, 0x5B5C5A1FU, 0xB205AC39U, 0x46A9372AU,
    0x941ABB66U, 0x60B64075U, 0x89EFB653U, 0x7D434D40U,
    0xD82495D8U, 0x2C886ECBU, 0xC5D198EDU, 0x317B63FEU,
    0xE3CE8FB2U, 0x176274A1U, 0xFE3B8287U, 0x0A977994U,
    0x4058C8A4U, 0xB4F433B7U, 0x5DADC591U, 0xA9013E82U,
    0x7BB2D2CEU, 0x8F1E29DDU, 0x6647DFFBU, 0x92EB24E8U,
    0x378CFC70U, 0xC3200763U, 0x2A79F145U, 0xDE50A56U,
    0x0C66E61AU, 0xF8CA1D09U, 0x1193EB2FU, 0xE53F103CU,
    0x840C894FU, 0x70A0725CU, 0x99F9847AU, 0x6D557F69U,
    0xBF69325U, 0x4B4A6836U, 0xA2139E10U, 0x56BF6503U,
    0xF3D8BD9BU, 0x07744688U, 0xEE2DB0AEU, 0x1A814BBDU,
    0xC832A7F1U, 0x3C9E5CE2U, 0xD5C7AAC4U, 0x216B51D7U,
    0x6BA4E0E7U, 0x9F081BF4U, 0x7651EDD2U, 0x82FD16C1U,
    0x504EFA8DU, 0xA4E2019EU, 0x4DBBF7B8U, 0xB9170CABU,
    0x1C70D433U, 0xE8DC2F29U, 0x0185D906U, 0xF5292215U,
    0x279ACE59U, 0xD336654AU, 0x3A6FC36CU, 0xCEC3387FU,
    0xF808F18AU, 0x0CA40A99U, 0xE5FDFCBFU, 0x115107ACU,
    0xC3E2EBE0U, 0x374E10F3U, 0xDE17E6D5U, 0x2ABB1DC6U,
    0x8FDCC55EU, 0x7B703E4DU, 0x9229C86BU, 0x66853378U,
    0xB436DF34U, 0x409A2427U, 0xA9C3D201U, 0x5D6F2912U,
    0x17A09822U, 0xE30C6331U, 0x0A559517U, 0xFE96E04U,
    0x2C4A8248U, 0xD8E6795BU, 0x31BF8F7DU, 0xC513746EU,
    0x6074ACB6U, 0x94D857E5U, 0x7D81A1C3U, 0x892D5AD0U,
    0x5B9EB69CU, 0xAF324D8FU, 0x466BBBA9U, 0xB2C740BAU,
    0xD3F4D9C9U, 0x275822DAU, 0xCE01D4FCU, 0x3AAD2FEFU,
    0xE81EC3A3U, 0x1CB238B0U, 0xF5EBCE96U, 0x01473585U,
    0xA420ED1DU, 0x508C160EU, 0xB9D5E028U, 0x4D791B3BU,
    0x9FCAF777U, 0x6B660C64U, 0x823FFA42U, 0x76930151U,
    0x3C5CB061U, 0xC8F04B72U, 0x21A9BD54U, 0xD5054647U,
    0x07B6AA0BU, 0xF31A5118U, 0x1A43A73EU, 0xEEEF5C2DU,
    0x4B8884B5U, 0xBF247FA6U, 0x567D8980U, 0xA2D17293U,
    0x70629EDFU, 0x84CE65CCU, 0xD9793E9AU, 0x993B68F9U
};

```

Figure B.4 – SC-32 Table

B.8 SID

All sources of safety related data (SDSRC) shall be identified by a source identifier (SID). The SID shall be an UNSIGNED32 value which is computed as a SC-32 signature of the following data structure, as shown in Figure B.5. The initial (seed) value shall be 'FFFFFFFF'H.

```

SID_STRUCT ::= RECORD
{
    SMI                UINT32          -- user defined safety message
                                   identifier. Shall be unique within
                                   a consist.
                                   value: 1..'FFFFFFFF'H (0 = reserved)

    reserved01         UINT16          -- reserved, shall be set to 0.

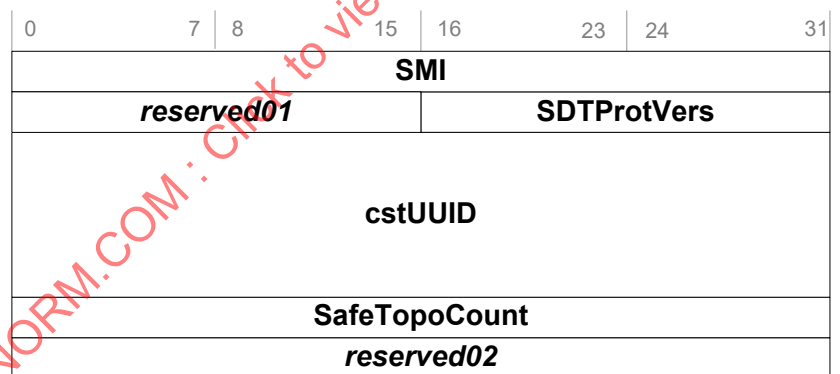
    SDTProtVers        UINT16          -- version of the SDTv2 protocol.
                                   shall be set to '0002'H

    cstUUID             ARRAY[16] OF  -- unique consist identifier
                                   UINT8

    SafeTopoCount       UINT32          -- safe topography counter (STC), unique
                                   identification of the actual train
                                   composition. Provided by the
                                   communication subsystem. To be set
                                   to 0 for consist network internal
                                   communication

    reserved02         UINT32          -- reserved, shall be set to 0.
}

```



IEC

Figure B.5 – SID Computation

The SafeTopoCount value shall equal the opTrnTopoCnt value as specified in 5.3.3.2.13 for train wide communication over ETB unless otherwise specified.

SMI values from 1 until 999 shall be reserved for generic VDPs which are specified within this part of the standard.

SDSINKs can compute the SIDs of expected VDPs with information retrieved from the TTDB. Solely the SMI needs to be preconfigured

EXAMPLE A SDSINK expecting VDPs with SMI = 4861 from the leading vehicle look up the CstUUID of the leading vehicle and the opTrnTopoCnt in the TTDB and compute the SID.

SDSINKs expecting VDPs from a redundant SDSRC have to compute two SIDs because the SMIs must be different, see B.12.3.2.

B.9 Vital Data Packet

A SDSRC shall encapsulate safety critical data in a vital data packet (VDP). VDPs transferred over ETB (ETB-VDP) are transmitted within the user data part of a TRDP process data telegram.

A VDPs may also be transmitted within the user part of a TRDP message data telegram, in which case the restrictions listed in X shall be considered.

The ETB-VDP is defined as shown in Figure B.6:

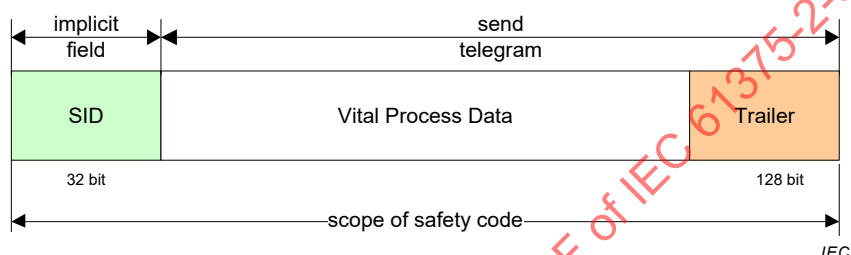


Figure B.6 – ETB-VDP

The ETB-VDP shall be represented by the following data structure, as shown in Figure B.7:

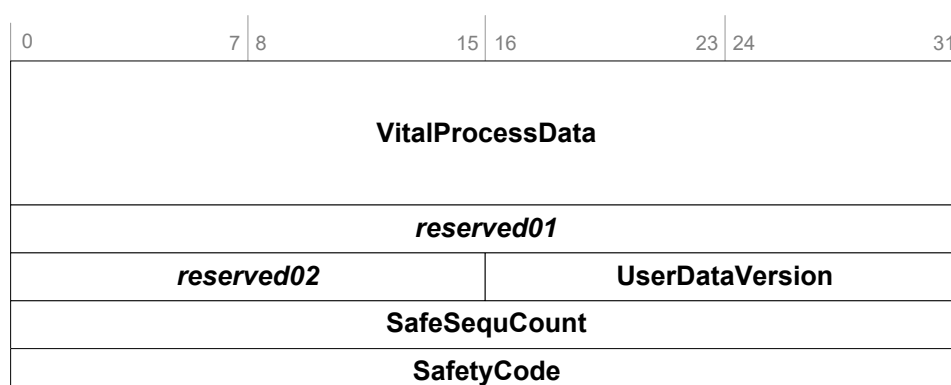
```
ETB_VDP ::= RECORD
{
    VitalProcessData    ARRAY [VDPSize] of UINT8
                        -- user defined data set
                        -- with safe and non-safe process data
                        -- VDPsize = 0 .. 984 (padding with
                        -- trailing bytes may be necessary to
                        -- have a total
                        -- length of a multiple of 32 bits).

    reserved01          UINT32    -- reserved, shall be set to 0.

    reserved02          UINT16    -- reserved, shall be set to 0.
    UserDataVersion     VERSION   -- version of the vital process data part

    SafeSequCount       UINT32    -- safe sequence counter (SSC),
                                -- implemented as a counter which is
                                -- incremented each time a new VDP
                                -- is generated and stored to the
                                -- communication channel interface

    SafetyCode          UINT32    -- SC-32, computed over VDP starting
                                -- with most significant byte of
                                -- VitalProcessData up to the
                                -- SafeSequCount.
                                -- Seed value: SID
}
```



IEC

Figure B.7 – Format of ETB-VDP

B.10 Exclusivity

The end device running the SDTv2 protocol shall not implement any other communication protocol which uses the same SC-32 or the same coding of SID.

B.11 Configuration time parameters

SDTv2 requires a set of (configuration) time parameters which are listed underneath. A more detailed specification of those parameters is given in later sections.

T_{tx_period}	Time period for sending VDP, as defined for SDSRC
T_{rx_period}	VDP receive (sampling) period, as defined for SDSINK
T_{rx_safe}	Maximal time for which SDSINK tolerates the absence of new (fresh) vital data
T_{guard}	Time used by SDSINK to detect the undesired presence of more than one active SDSRC in case of redundant SDSRC

B.12 Safe data source (SDSRC)

B.12.1 General

This clause defines protocol requirements on safe data sources. A safe data source has to produce VDPs, meaning that the VDPs are generated and are subsequently passed to the communication layer for transmission.

B.12.2 Safe Data Preparation (Application)

The following requirements define some application conditions. In general, the application is responsible for providing the vital process data to be sent with SDTv2. Two input data classes are distinguished:

- Continuous data.** Those data are characterized by changing their value more or less continuously over time (example: speed signal). Only samples of those data need to be transmitted. After sampling, the sampled data value is kept constant until the next sample

(sample and hold principle). The time, during which such a sampled data value is constant, is subsequently called "signal sample duration time" (T_{sig}).

- b) **Discrete data.** Those data are characterized by changing their value on event (example: doors close/open signal). All different values of those data need to be transmitted, because otherwise safety related information might get lost. The minimal time during which the signal value is constant is as well referred to as the "signal sample duration time" (T_{sig}). Contrary to continuous input signals where T_{sig} is exactly one sampling period, can T_{sig} be a multiple of a sample period in the case of discrete signals.

By this, T_{sig} defines the time during which a data value sample is kept constant within the SDTv2 application interface of SDSRC.

In a VDP containing more than one discrete data item which might be typically the case, the signal sample duration time can be different for the individual data items, because it is a property of the source data items itself. The value of T_{sig_min} defines the lowest signal sample duration time occurrence of all the data items within a VDP.

The application shall ensure that all value changes of a discrete input data item (signal) are sampled and that the samples are kept stable in the SDTv2 application interface for a time T_{sig} .

Sampling rate of the application for sampling the input data item needs to be higher than the change rate (frequency) of the input data item.

NOTE T_{sig} can be different for the different input data items.

B.12.3 Safe data sending

B.12.3.1 Protocol

Safety related data shall be sent within the VitalProcessData part of Vital Data Packets (VDP).

The producer (SDSRC) of VDP shall produce the VDP periodically with a cycle time of T_{tx_period} .

NOTE 1 T_{tx_period} will be defined by application, e.g. within IEC 61375-2-4.

The selection of T_{tx_period} shall comply with the following condition:

$$T_{tx\_period} \leq T_{sig\_min} / 3$$

with T_{sig_min} being the lowest signal sample duration time of a sampled input data item (signal) exposed in the SDTv2 application interface.

NOTE 2 This ensures that all signal values are transmitted even in the case of two subsequent VDP losses during transmission.

VDPs shall not be produced if no valid SID can be computed.

NOTE 3 A valid SID can for instance not be computed if SID input parameters are missing.

VDPs shall not be produced if the end device hosting the SDSRC is not a safety device.

The SDSRC shall increment the SSC for each produced VDP:

$$SSC(i+1) = (SSC(i) + 1) \bmod 2^k$$

with $i = 0 \dots \infty$ and k being the cardinality of the safety sequence counter used (ETB-VDP: $k = 32$).

B.12.3.2 Redundant SDSRC

This subclause defines specific requirements on redundant safe data sources. The redundancy principle is to have two redundant source devices (SDSRC-A and SDSRC-B) forming one redundancy group. The input signal is read by both source devices, but only one device (redundancy leader) is actively sending to the sink (SDSINK), the other device (redundancy follower) is not sending. Both source devices supervise each other, and if the redundancy follower detects a failure of the redundancy leader it starts actively sending to SDSINK, see Figure B.8.

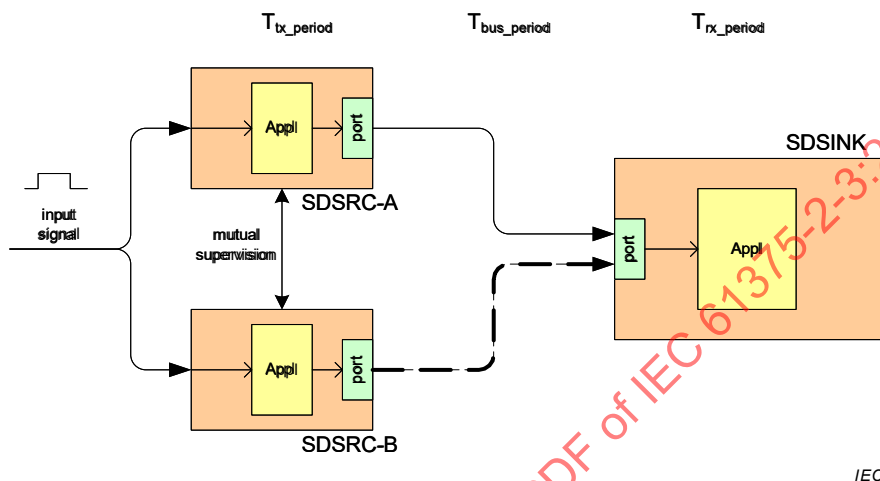


Figure B.8 – Redundancy Group (Example with 2 SDSRC)

NOTE 1 It is not defined herein how the mutual supervision between a redundancy leader and a redundancy follower, and how the switch-over from a redundancy leader to a redundancy follower is realized. This is an implementation choice.

In a redundant producer group of SDSRC (e.g. hosted by a redundant group of safety devices forming a redundancy group) only one SDSRC shall send VDPs.

All SDSRC of a redundancy group shall use different safety message identifiers (SMI).

The time span T_{red} between the redundancy leader ceasing to send VDPs and the redundancy follower start to send VDPs shall not exceed a value of:

$$T_{red} \leq T_{rx\_safe} - 2 \times \max(T_{tx\_period}, T_{rx\_period})$$

NOTE 2 A violation of this rule may trigger the sink time supervision of SDSINK.

NOTE 3 For the definition of T_{rx_safe} and T_{rx_period} see below.

NOTE 4 T_{rx_period} might be longer than T_{tx_period} in case of undersampling.

It has to be considered that there might be multiple SDSINK connected to SDSRC, in which case the lowest value of T_{red} has to be used.

NOTE 5 By fault both SDSRC may send VDPs. This will be detected by SDSINK with the guard time check (see below).

B.13 Safe data sink (SDSINK)

B.13.1 General

This clause defines specific protocol requirements on safe data sinks. SDSINK has to receive ("sample") VDPs, to validate the VDPs, and to expose received process data in the SDTv2

application interface. "Sampling" of VDPs in this context means that the most recent VDP is read from the communication channel interface. The terms "sampling" and "receiving" are used synonymously.

B.13.2 Definitions

B.13.2.1 Variables

For the subsequent specification, the following set of variables is used:

SSC	this is the non-stored SSC value of the actually sampled VDP
SSC_i	this is the SSC value of the last valid VDP
$SSC_{received}$	SSC value of the actually sampled correct VDP
$SSC_{initial}$	SSC value of the initial VDP
SSC_{last}	this is the SSC value of the previously sampled valid VDP
$SID_{initial}$	SID value of an initial VDP

B.13.2.2 VDP Classification

B.13.2.2.1 Duplicate VDP

A VDP is considered a **duplicate** if it is identical to the VDP received before, meaning that the computed SafetyCode of that VDP is identical to the computed SafetyCode of the VDP received before.

NOTE 1 For VDPs already validated with correct SafetyCode and correct user data main version, it is sufficient to compare the SSC in order to identify a duplicate: if $SSC = SSC_i$ or $SSC = SSC_{initial}$ then the VDP is a duplicate.

NOTE 2 "Computed" SafetyCode means that the SafetyCode computed by the SDSINK during reception is used, not the SafetyCode value written in the VDP.

B.13.2.2.2 Correct VDP

A received VDP is considered **correct**, if:

- SafetyCode is correct (computed SafetyCode value is identical to the SafetyCode value contained in the VDP);
- UserDataMainVersion is correct (equals the expected user data version value).

B.13.2.2.3 Initial VDP

A received VDP is considered **initial** in one of the following cases:

- a) it is not a duplicate;
- b) it is the first correct VDP received after power-up/reset;
- c) it is the first correct VDP received after a communication loss as indicated by the sink time supervision;
- d) it is a correct VDP, but where the SafetyCode evaluation has been done with the alternative SID of the redundant SDSRC (not the stored $SID_{initial}$ value of the previously received initial VDP).

NOTE A VDP with a different SID than $SID_{initial}$ may for instance be received when a redundancy shift occurs within a SDSRC redundancy group.

B.13.2.2.4 Fresh VDP

A VDP is considered **fresh** if:

- it is correct;
- the VDP is not the initial VDP;

- the SID of the VDP is identical to $SID_{initial}$;
- it is a real successor to the initial or fresh VDP received before, meaning that $SSC \in \{(SSC_i + 1) \bmod (2^k), \dots, (SSC_i + N_{SSC}) \bmod (2^k)\}$, with $N_{SSC} = (T_{rx\_safe} / T_{tx\_period})$, rounded up to an integer value, and k being the cardinality of the SSC.

B.13.2.2.5 Valid VDP

A received VDP is considered **valid**, if it is a fresh VDP or a duplicate of the fresh VDP received before.

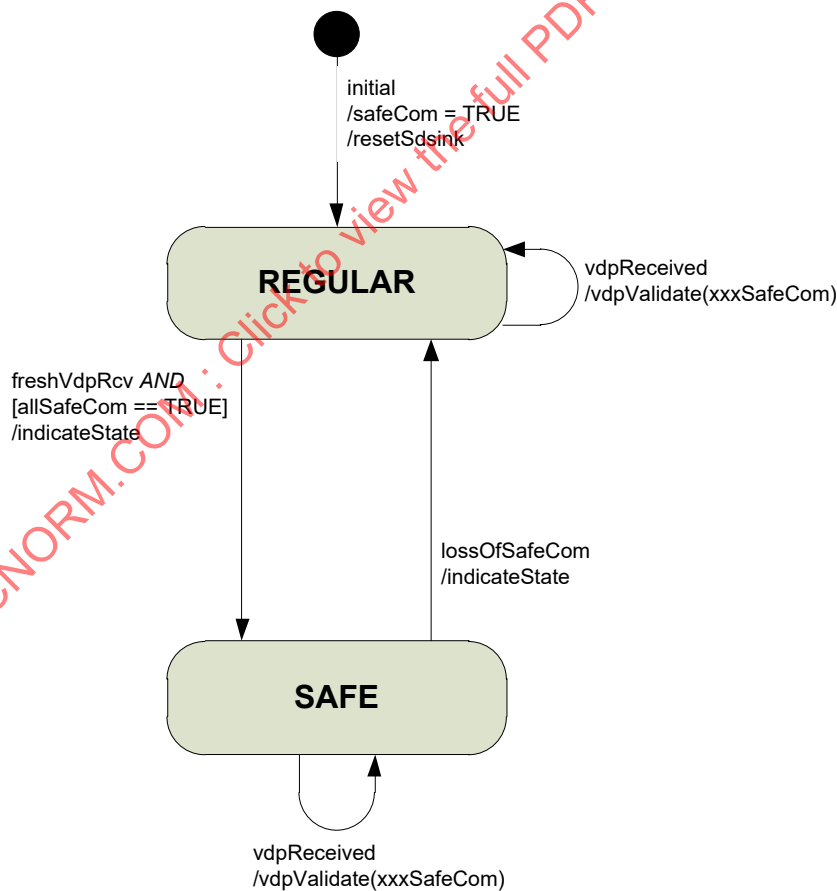
In all other cases it shall be considered invalid.

B.13.2.2.6 Discarded VDP

To **discard** a VDP means to not expose its data to the application.

B.13.3 SDSINK States

The state diagram shown in Figure B.9 defines the two possible main states a SDSINK can be in. The related triggers, guards and operations are defined in Table B.2 to Table B.4.



IEC

Figure B.9 – SDSINK state diagram

Table B.2 – SDSINK state diagram – triggers

Trigger	Description
Initial	Power-up or re-boot of SDSINK
vdpReceived	The most recent VDP is read from the communication channel interface, see B.13.4.
freshVdpRcv	A fresh VDP has been received
lossOfSafeCom	During VDP validation a loss of safe communication has been detected

Table B.3 – SDSINK state diagram – guards

Guard	Description
allSafeCom	Logical AND over the check results (variable 'xxxSafeCom', see below)

Table B.4 – SDSINK state diagram – operations

Operation	Description
vdpValidate	Validation of received VDP, in particular: - VDP integrity check, see B.13.5 - Sink time supervision check, see B.13.6 - Guard time check (result: gtcSafeCom), see B.13.7 - Latency monitoring (result: lmcSafeCom), see B.13.8 - Channel monitoring (result: cmcSafeCom), see B.13.9 The returned check results can have the following values: xxxSafeCom=TRUE: SDTv2 channel communication is considered safe xxxSafeCom=FALSE: SDTv2 channel communication is considered not safe (regular) (xxx stands for 'gtc', 'lmc' or 'cmc')
resetSdsink	Reset the SDSINK
IndicateState	Indicate a state change to the application

B.13.4 VDP Sampling

A configurable time T_{rx_safe} shall be defined for detecting the loss of safe communication. By default, T_{rx_safe} shall be set in a way that the loss of two subsequently send VDPs is tolerated, e.g. $T_{rx\_safe} := 3 \times T_{tx\_period}$.

Tolerances in timing should be considered for real implementations (e.g. T_{rx_safe} can be set to $3,5 \times T_{tx\_period}$ to compensate tolerances in T_{tx_period} and to respect transmission jitter).

The default setting of T_{rx_safe} might be changed in the case of under-sampling, see below.

A configurable time T_{rx_period} shall be defined which specifies the cycle in which SDSINK reads (samples) VDPs from the communication channel interface.

NOTE 1 T_{rx_period} defines the time between two samplings of the communication channel interface. The model of a periodic sampling of the communication channel interface is used for the purpose of this specification, but it does not imply a specific implementation. A real implementation can also use an event-driven VDP reception procedure.

The period T_{rx_period} of reading VDPs should be shorter than T_{tx_period} of the SDSRC.

NOTE 2 This is necessary to sample all received VDPs.

Undersampling, meaning that T_{rx_period} is longer than T_{tx_period} , can be configured. If so, it shall be defined whether information loss is allowed or not.

Under-sampling without information loss might for instance be used if the SDSINK is only interested in specific discrete vital process data items (signals) within the received VDP, which will change their value less frequently as other data items (signals) in the same VDP.

Under-sampling with information loss might for instance be used when continuous signals are sampled (e.g. train speed signal) and a lower granularity is sufficient.

In case of SDSINK undersampling without data loss, the time T_{rx_safe} shall be shorter than the symbol sample duration time of the signal the SDSINK is interested in, but longer than T_{rx_period} .

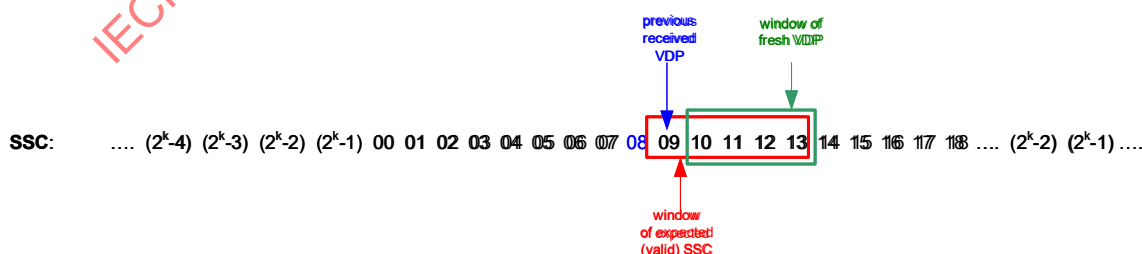
NOTE 3 If T_{rx_safe} were longer than the symbol sample duration time, a signal change would get lost undetected.

B.13.5 VDP Integrity Check

B.13.5.1 General

The VDP integrity check aims to filter VDPs which are not correct, meaning that data are corrupted or the user data main version is not the expected one. Data within those 'invalid' VDPs cannot be used (consumed) by the application.

After a power-up, reset, redundancy shift or a loss of safe communication, the receiver waits for the reception of an "initial" VDP. This initial VDP is used to "synchronize" the SDSINK with the SDSRC. Receiving the initial VDP is however not sufficient to indicate the reception of valid and safe data to the application: this will be done only with the next received fresh VDP, which matches the "window of expected SSC". This window defines a range of allowed SSC values which have to be matched by the VDPs following the initial VDP, in order to ensure that the received VDPs are in correct sequence. This window is shifted to the right with each received VDP carrying a new matching SSC (see Figure B.10), so subsequent VDPs need to match the shifted window. In the example below, a VDP with SSC = 9 has been received. The next VDP is expected to have a SSC in the range of 09 to 13. If the next received VDP has a SSC of 09, it is a duplicate to the previously received VDP. If it has a SSC of 10 to 13, it will be a fresh VDP. All VDPs matching the window are called "valid", but only those with a new SSC value are called "fresh". After receiving the VDP with SSC = 10, the window is shifted by one covering now the range of 10..14 (not shown).



IEC

Figure B.10 – Window of expected SSC (example)

B.13.5.2 VDP processing

After reading a VDP from the communication channel interface, SDSINK shall first check the correctness of the VDP.

If VDPs from a redundancy group are expected, the correctness check shall be made with the expected SIDs associated to the SDSRC of the redundancy group.

NOTE In a redundancy group of two SDSRC, first a check can be made with the SID already known from the previously received VDP, and only if this fails it will be done with the second SID.

SDSINK shall check whether a received VDP is initial. Upon reception of an initial VDP, SSC_i and $SSC_{initial}$ shall be set to the SSC of this initial VDP and $SID_{initial}$ shall be set to the SID value used for the validation.

SDSINK shall check whether a received VDP is valid.

SDSINK shall check whether a received VDP is fresh. If the received VDP is a fresh VDP, SSC_i shall be set to the value of the SSC of the received fresh VDP.

This means that the receiver needs only to implement one sequence counter.

User data contained in fresh or valid VDPs shall be exposed to the application for consumption if SDSINK is in state 'SAFE'.

User data contained in invalid VDPs shall not be exposed to the application for consumption (shall be discarded).

The application may consume the user data received from the most recently received valid VDP in those cases as long as data is indicated as safe.

B.13.6 Sink time supervision

With the reception of an initial VDP, the receiver shall start a timer which expires after a time T_{rx_safe} ("sink time supervision timer").

The sink time supervision timer shall be retriggered with the reception of a fresh VDP.

If the sink time supervision timer expires, a loss of safe communication shall be indicated (trigger 'lossOfSafeCom') and the SDSINK shall wait for the next received VDP which is not a duplicate. This VDP shall be treated as an initial VDP.

B.13.7 Guard time check

B.13.7.1 General

The guard time check intends to detect two redundant active SDSRC (both sending VDPs). For this, a "guard time" is introduced which shall start with the reception of an initial VDP and shall last for a multiple of T_{rx_safe} (configurable). If a VDP with another SID than expected is received during that time, SDSINK assumes that both redundant SDSRC became active and shall indicate loss of communication safety. This situation is depicted in Figure B.11. Here, SDSINK receives first VDPs with SID=A and then VDPs with SID=B. So SDSINK assumes that a redundancy shift at SDSRC side has taken place and expects to receive only VDP with SID=B further on. If it receives a VDP with SID=A now during the time T_{guard} , SDSINK assumes that the SDSRC sending SID=A is still active, which is a redundancy fault at SDSRC side. Such an event is called a "guard time violation". As mentioned, the guard time starts with the reception of an initial VDP. In the example, there are four initial VDPs: the first at the very beginning and the second after the redundancy shift. The VDP with the unexpected SID=A is as well interpreted as an initial VDP according to the rules defined for initial VDPs. This means, that the guard time supervision is retriggered in that case. The same is the case for the next received VDP with SID=B again. Consequently, guard time will be retriggered all the way and practically never expires if there is a mixed reception of VDPs with SID=A and SID=B. The guard timer will only expire if there is a stable reception from one SDSRC (again). So, with expiring guard timer, a loss of safe communication can be negated.

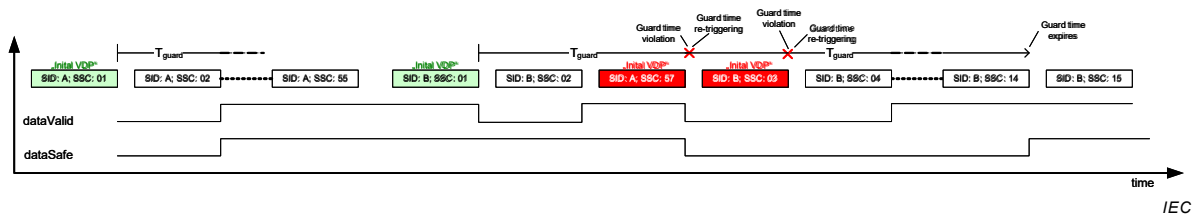


Figure B.11 – Guard time violation (example)

B.13.7.2 Requirements

With the reception of an initial VDP with a SID different to $SID_{initial}$, the receiver shall start a timer, which expires after a time T_{guard} (guard timer).

The receiver shall indicate a loss of safe communication (trigger 'lossOfSafeCom', 'gtcSafeCom = FALSE') if it receives, during the time the guard timer is active, an initial VDP with a SID different to $SID_{initial}$. This is called a "guard time violation".

NOTE 1 The intention with a guard time supervision is to detect two active redundant SDSRC sending VDPs, which normally never should happen.

The parameter T_{guard} shall be defined in a range of:

$$2 \times T_{rx\_safe} \leq T_{guard} \leq 1000 \times T_{rx\_safe}$$

NOTE 2 A good practical value will be $T_{guard} = 10 \times T_{rx\_safe}$.

In case of a guard time violation, the guard timer shall be retriggered in order to start a new guard time supervision time interval.

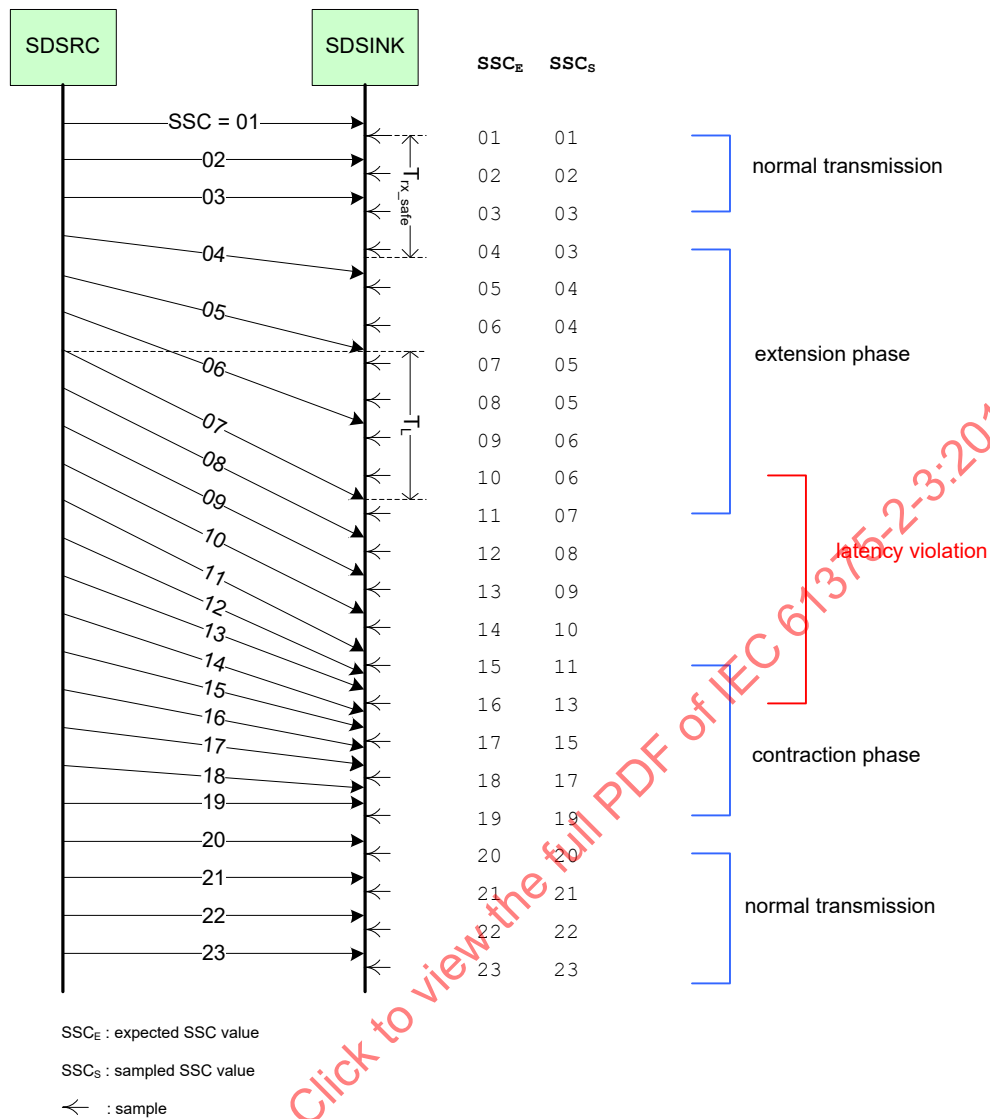
The guard time supervision shall cancel the loss of safe communication indication (gtcSafeCom = TRUE) if the guard timer expires.

B.13.8 Latency monitoring

B.13.8.1 General

The intention of latency monitoring is to supervise the latency of VDP transmissions. The latency time T_L is the VDP transmission time from SDSRC to SDSINK. In a fault free and properly configured network, this latency time will vary around a constant mean value T_{L_mean} . In case of a network fault⁵, this latency time mean value may increase over time, especially in networks with high queuing capabilities, as it is the case in networks with gateways or IP routers. This can be seen in the Figure B.12. Here, beginning with the VDP SSC=04, latency is slowly increasing (extension phase). The increase is not that much that sink time supervision triggers. With VDP SSC=06 the latency time T_L becomes higher than T_{rx_safe} , meaning that received VDPs are outdated (latency violation). The system can then return to normal condition with a contraction phase, during which (queued) VDPs are received closely. During the contraction phase, vital data may get lost, because the receiving frequency may be much higher than the sampling rate.

⁵ A possible network fault is when a high priority VDP is stored by fault within a low priority queue which carries low priority bulk data like for instance stream data. The probability of such a fault is mainly determined by the hardware of the communication channel.



IEC

Figure B.12 – Latency violation sequence chart (example)

The latency time can be expressed by the following formula⁶:

$$T_L = T_{L\_mean} + K_2 \times t,$$

with

T_{L_mean} : mean latency time (in absence of an extension)

K_2 : gradient of latency time increase/decrease,

t : time, expressed in T_{tx_period} cycles

In a fault free network, $K_2 = 0$.

The latency cannot be measured directly because a VDP does not contain a time stamp⁷. But it can be measured indirectly when the SDSINK knows the sending period T_{tx_period} of the

⁶ For this formula a linear increase/decrease of latency is assumed.

⁷ A time stamp is not used because maintaining a safe synchronized clock with a dynamically changing TCN is considered to be not manageable.

SDSRC. It then can predict which SSC value is expected at a certain point in time. For doing so, the expected SSC values of VDPs (SSC_E) to be received can be predicted and can be compared with the SSC value of the actually received SSC (SSC_S).

SSC_E can be predicted with the calculation:

$$SSC_E = SSC_0 + (t - t_0) / T_{tx\_period}$$

with t_0 , SSC_0 being the time and the SSC value at begin of the measurement, t being the actual time and ' $(t - t_0) / T_{tx\_period}$ ' rounded up to an integer value.

In order to compensate for tolerances in SDSRC and SDSINK time base, the latency measurement can be executed in subsequent measurement intervals of a limited duration (measurement time T_M), where at the begin of each interval the measurement is calibrated. This calibration is done by setting SSC_0 to the SSC value received with the first VDP after measurement start. For detecting latency violations, T_M needs to be higher than T_{rx_safe}/k_2 and lower than T_{rx_safe}/ξ , with ξ being the precision of the time base.

EXAMPLE In order to detect a latency increase of $k_2 = 0,1$, the duration of the measurement interval shall be at least 3,0 s, if $T_{rx\_safe} = 0,3$ s, and maximally 300,0 s for an assumed time base tolerance of $\xi = 1$ ‰. In praxis it is recommended to choose a value close to the minimal value, e.g. $T_M = 4,0$ s.

NOTE The measurement calibration at the beginning of a measurement interval eventually hides a latency violation detected during the measurement interval before, although the latency violation is still persisting. This is a known limitation of the procedure. This limitation can be reduced or even abolished by using more accurate time bases or by synchronising the time bases.

The time $T_{L\_violation}$ defines the time span between start of latency increase until it is detected.

B.13.8.2 Procedure

With the reception of an initial VDP, SDSINK shall start to monitor the latency time increase.

SDSINK shall indicate a loss of safe communication (trigger 'lossOfSafeCom', guard 'lmcSafeCom = FALSE') if the following condition is fulfilled:

$$\{SSC_E(t) - SSC_S(t)\}_{mod(2^k)} \times T_{tx\_period} \geq T_{rx\_safe}, \text{ with } t \text{ being the SDSINK time}$$

SDSINK shall be able to detect a latency violation in case of a latency time increase with a constant gradient of $K_2 \geq 0,1$ within a time of:

$$T_{L\_violation} \leq 2 \times (1 + 1/K_2) \times T_{rx\_safe}$$

EXAMPLE $K_2 = 0,1$, $T_{tx\_period} = 0,1$ s, $T_{rx\_safe} = 0,3$ s. Then the mean latency T_{L_mean} increases with 0,01 s for each transmitted VDP. It then should take maximally 6,6 s until latency monitoring triggers.

SDSINK shall cancel the loss of safe communication indication (guard 'lmcSafeCom = TRUE') if the following condition is fulfilled:

$$\{SSC_E(t) - SSC_S(t)\}_{mod(2^k)} \times T_{tx\_period} < T_{rx\_safe}, \text{ with } t \text{ being the SDSINK time}$$

B.13.9 Channel monitoring

B.13.9.1 General

Channel monitoring intends to detect a sudden increase of the transmission failure rate within the SDTv2 channel which might follow a hardware or software fault in one of the components belonging to the SDTv2 channel. With an increased transmission failure rate, the probability that a corrupted VDP slips through the SafetyCode checker undetected grows and might become unacceptable. Those failures might be permanent, in which case a repair action will be required, or temporary, in which case the system might recover itself.

B.13.9.2 Procedure

The channel monitoring shall indicate a loss of safe communication (trigger 'lossOfSafeCom', result 'cmcSafeCom = FALSE') if the number of VDPs received with incorrect SafetyCode exceeds a threshold defined by a configurable parameter K_3 (channel monitoring threshold).

Channel monitoring should cancel the loss of safe communication indication ('cmcSafeCom = TRUE') if the rate of VDPs received with incorrect SafetyCode is lower than K_3 .

VDP duplicates shall not be considered for channel monitoring (shall be filtered out).

By default, K_3 shall be set to a value of $K_3 = 43$ per hour.

NOTE K_3 is the quotient of the tolerated hazard rate R_{H3} for undetected corrupted VDPs and the performance $p_{us} \approx 2^{-32}$ of the SafetyCode SC-32 for detecting corrupted VDPs (see IEC 62280). For $R_{H3} = 10^{-8}$, this results in a value of $K_3 = 43$.

B.13.9.3 Activation

With the creation of a SDTv2 channel, channel monitoring shall be started and shall be active as long as the SDTv2 channel is setup.

In case a SDTv2 channel is relinquished or reconfigured, channel monitoring should be aborted and restarted again when reconfiguration is completed.

NOTE During the reconfiguration of a SDTv2 channel spurious VDPs with incorrect SafetyCode might be detected because the SDSINK may use another value of the SafeTopoCount for VDP validation than the SDSRC. This is typically the case during train composition changes or changes in the operational train directory.

EXAMPLE If the operational train directory changes (e.g. change of leading vehicle), existing SDTv2 channels between a function in the leading vehicle and another function are relinquished and are setup again when the operational train directory is updated.

B.13.9.4 Channel monitoring algorithm (informative)

There are surely different ways to implement channel monitoring, with different characteristics and performance. The following algorithm allows an easy implementation of channel monitoring:

First filter out all duplicated VDPs (before checking the SafetyCode for correctness).

Provide a counter Z which:

- Increments each time it receives a VDP with incorrect SafetyCode by a value of f_M / K_4 , until a maximum value of $(f_M \times K_3 / K_4)$.
- Decrement each time it receives a VDP with correct SafetyCode by 1, until 0 is reached.

f_M is the frequency of VDPs sent per hour: $f_M = 3600 / T_{tx\_period}$

Parameter K_4 to be set to $K_4 = 36$, for the case that $K_3 = 43$

If Z exceeds a threshold $CMMAX = (f_M / K_4) \times (K_3 - K_4)$, a loss of safe communication is indicated.

The indication of a loss of safe communication is canceled if Z reaches 0.

B.13.10 SDTv2 Application Interface

SDSINK state transitions from state REGULAR to state SAFE and vice versa shall be indicated to the application.

Vital process data exposed in the application interface shall be invalidated (set to 0) in SDSINK state REGULAR.

B.13.11 Change of operational train composition

B.13.11.1 General

In case of a change of the operational train directory, the SDSRC will thereafter use another SID value in the computation of the safety code. This means that SDSINK also has to change the value of the expected SID(s). There are two use cases:

- a) SDSRC is the same as before, only SID changed because there is a new SafeTopoCount value
- b) SDSRC is another than before, e.g. in the case the SDSRC belongs to a function of the leading car and the leading car position itself changed.

The second case is comparable to a redundancy shift of a redundant SDSRC couple. Consequently, SDSINK has to behave in the same manner – after getting informed about new SID value(s) SDSINK shall treat the next received correct VDP, which then uses the new SID, as an initial VDP.

B.13.11.2 Interrupting channel monitoring

Special attention has to be paid to the potential asynchrony between a SID change at SDSRC and SDSINK side.

In both cases, either when SDSRC changes first and then the SDSINK, or vice versa, there will be a certain time where SDSINK detects safety code errors, which could trigger the channel monitoring.

Therefore, SDSINK shall stop channel monitoring as soon as it gets informed by its local ETBN about a new train inauguration.

SDSINK shall resume channel monitoring with the initial VDP received after the SID change.

B.14 Diagnosis and statistics

The statistic counter listed in Table B.5 should be provided.

Table B.5 – SDTV2 statistic counters

Statistic counter	Provided by
statistic counter of produced VDPs. NOTE How statistic counters are made accessible has to be defined on application level.	SDSRC
statistic counter of received fresh VDP	SDSINK
statistic counter of received non-duplicated VDP with a SafetyCode error	SDSINK
statistic counter of received VDP with a SID different to the expected SID NOTE This is the case when a redundancy shift at SDSRC occurs	SDSINK
statistic counter of VDP received "out of sequence".	SDSINK
statistic counter of received VDP duplicates	SDSINK
statistic counter of latency gap occurrence	SDSINK
statistic counter of received VDP with wrong UserDataMainVersion	SDSINK
statistic counter of communication safety losses caused by SDTV2 channel monitoring	SDSINK

SDSINK statistic counters should be maintained individually for each expected (configured) SID.

Statistic counter should be implemented as 32-bit unsigned integer

SDSINK statistics counter shall be reset at power up and device reset.

SDSINK statistic counters shall stop counting after reaching their maximal value (no overflow)

B.15 Safe data transmission over MVB (informative)

B.15.1 General

Although SDTV2 is basically defined for the safe communication over ETB and ECN, there might still be end devices connected to MVB which have to communicate safely. In principle, the same protocol can be used, but due to some limitations on MVB some modifications of the SDTV2 protocol are necessary. These deviations from SDTV2 are defined within the next subclauses.

B.15.2 MVB-VDP

The MVB-VDP is defined as shown in Figure B.13 and Figure B.14:

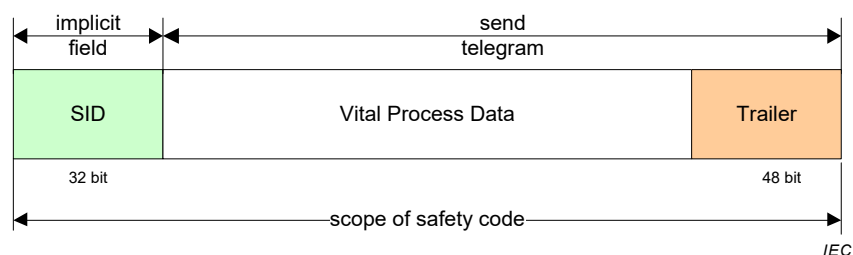


Figure B.13 – MVB-VDP

The MVB-VDP is defined as follows:

```

MVB_VDP ::= RECORD
{
  VitalProcessData  ARRAY [VDPSize] of WORD8
    -- user defined data set with
    -- safe and non-safe process data.
    -- VDPSize = 2, 10 or 26
    -- (dependant on MVB frame data size)

  UserDataMainVersion  UNSIGNED4 -- main version of the vital process
    -- data part, shall be set by
    -- the application
    -- Value: 1..15 (0 = reserved)

  reserved01           UNSIGNED4 -- reserved value, shall be set to 0

  SafeSequCount        UNSIGNED8 -- safe sequence counter (SSC),
    -- implemented as a counter which is
    -- incremented each time a new MVB-VDP
    -- is produced to the MVB traffic store port

  SafetyCode           UNSIGNED32 -- SC-32, computed over VDP starting
    -- with most significant byte of
    -- VitalProcessData
    -- up to and including the SafeSequCount.
    -- Start value: SID
}

```

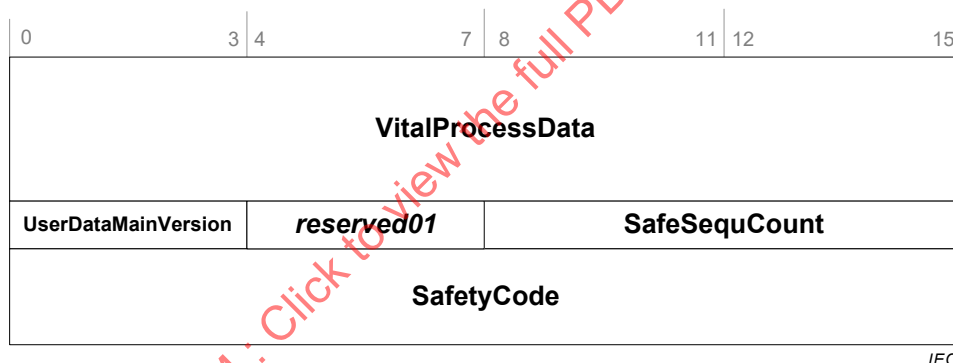


Figure B.14 – Format of MVB-VDP

B.15.3 SDTV2 protocol deviations for MVB

Vital MVB process data shall be packed into MVB-VDPs for communication.

- The parameter ConsistId of the SID as defined in Clause B.8 shall be set to 0 since there is only intra consist communication via MVB.
- The parameter SafeTopoCount of the SID as defined in Clause B.8 shall be set to 0 since there is only intra consist communication via MVB.
- Latency monitoring as specified in B.13.8 is only optional for MVB sink.

B.16 SDTv2 with TRDP message data

Safety related data may be transmitted within the payload of TRDP message data telegrams, if the restrictions defined in this clause are considered. Contrary to process data are message data transmitted event based, and there is no guarantee that data sent by SDSRC are safely received by SDSINK in time, nor is there a safe feedback from SDSINK to SDSRC that data have arrived. To cope with those shortages, the following requirements shall be applied:

- a) A SDSRC shall encapsulate safety critical data in an ETB-VDP as defined in B.9, with ETB_VDP parameter 'SafeSequCount' = 'FFFFFFFF'H.
- b) ETB-VDP shall only be sent on a SDTv2 channel on which a parallel safe SDTv2 connection based on TRDP process data is existing.

This shall ensure that the SDTv2 channel is supervised with respect to availability (sink time supervision), redundancy (guard time supervision), latency (latency monitoring) and error rate (channel monitoring).

- c) SDSINK shall inform SDSRC about the reception of the ETB-VDP. This can be done directly, e.g. by sending a TRDP-MD reply telegram with a reply VDP, or indirectly via another mechanism).

EXAMPLE If the CSTINFOCTRL telegram is not received by the SDSINK ECSP, the SDSRC will indirectly recognize this because it cannot transit from train directory computation state 'WaitForCstInfo' to state 'TrnDirSetup'.

Annex C (informative)

Train Real-Time Data Protocol Configuration (TRDP)

C.1 General

This annex defines, as an extension to the communication profile, an XML format that can be used to configure the TRDP.

In order to exchange data over TCN, TRDP needs to be configured, e.g. with information on how to handle the data, where to send data, etc. This information is stored in a **Configuration Database**.

The database can be populated from configuration file(s) in XML-format or with the aid of API functions as it is shown in Figure C.1. The database is usually read in at start up from a configuration file in XML-format. The file(s) is created by an off-line tool and downloaded as a DLU (Downloadable Unit).

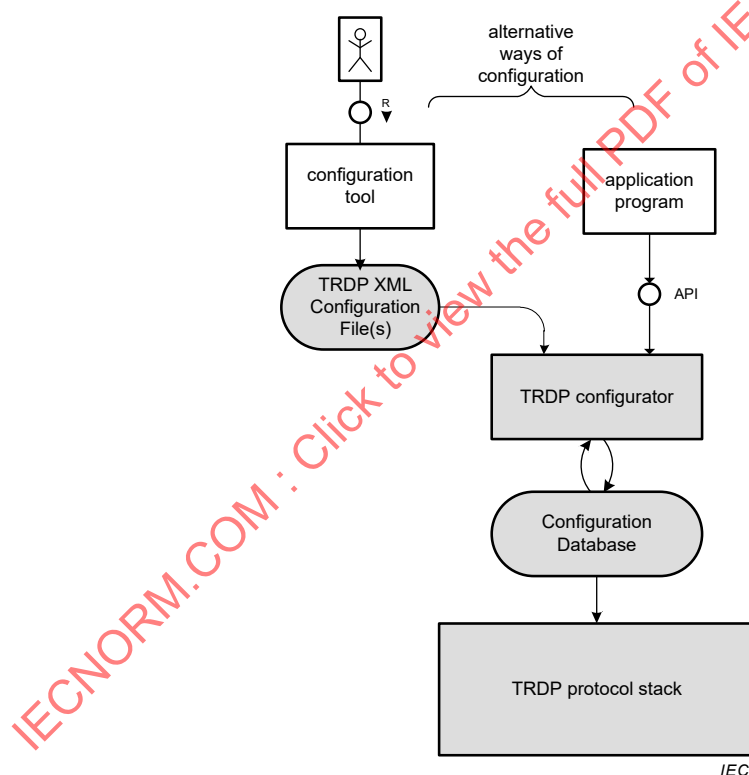


Figure C.1 – TRDP configuration block diagram

The structure of this XML file can be described in a XML Schema Definition (.XSD). By use of XML schema definition, it is possible to validate a configuration file. The schema definition describes only the formal structure of the configuration and can be used to validate the configurations by TRDP.

The definition files can be common for different communication concepts. In the clauses below, only features relevant to TRDP are described. The TRDP configuration includes:

- Device Configuration Parameters
- Bus Interface Parameters including related Telegram Parameters (Exchange Parameters)
- Mapped device parameters
- Communication Parameters
- Dataset Parameters
- Debug Parameters

NOTE The column “Optional/Required” in the tables below is used for what is optional and required for TRDP and not what is optional and required for the XML file or for other use of the XML file, e.g. tools.

The overall structure of the XML configuration description is defined as:

```
<device attributes>
  <device-configuration attributes/>
  <bus-interface-list>
    <bus-interface attributes>
      <telegram attributes >
        :
      </telegram>
      :
    </bus-interface>
    :
  </bus-interface-list>
  <mapped-device-list>
    <mapped-device attributes>
      <mapped-bus-interface attributes>
        <mapped-telegram attributes/>
        :
      </mapped-bus-interface>
      :
    </mapped-device>
    :
  </mapped-device-list>
  <com-parameter-list>
    <com-parameter attributes/>
    :
  </com-parameter-list>
  <data-set-list>
    <data-set attributes >
      :
    </data-set>
    :
  </data-set-list>
  <debug attributes/>
</device>
```

C.2 Device Parameters

The parameters given with the device can be used to initialize the TRDP stack of a device.

XML Element:

```
<device >
:
</device>
```

The tag “device-configuration” is optional.

XML Attributes:

All attributes that can be specified for the tags are described in the following Table C.1.

Table C.1 – Attributes for device tag

Name	Data Type	Optional/Required	Description
host-name	STRING	Required	Device host name
leader-name	STRING	Optional	Leader host name, depending on redundancy concept
type	STRING	Optional	Device type, for information only

C.3 Device Configuration Parameters

The parameters given with the device configuration can be used to initialize the TRDP stack of a device.

Device configuration parameters may be provided for:

- Memory configuration

XML Element:

```
<device >
  <device-configuration attributes />
:
</device>
```

The tag “device-configuration” is optional.

XML Attributes:

All attributes that can be specified for the tags are described in the following Table C.2.

Table C.2 – Attributes for device-configuration tag

Name	Data Type	Optional/Required	Description
memory-size	UINT32	Optional	Size of TRDP total memory

C.4 Bus Interface List

C.4.1 General

The Bus interface list contains the configuration parameter needed for the specific interfaces.

XML Element:

```

<device>
  <bus-interface-list>
    <bus-interface attributes>
      <trdp-process attributes/>
      <pd-com-parameter attributes/>
      <pd-com-parameter attributes/>
      <telegram attributes >
        :
      </telegram>
    :
  </bus-interface>
  :
</bus-interface-list>
</device>

```

C.4.2 Bus Interface Configuration**C.4.2.1 General**

The parameters given with the bus interface configuration can be used as parameters to initialize a specific TRDP session.

For each bus interfaces parameters may be provided for:

- Interface configuration
- Thread/task configuration
- Default PD Communication
- Default MD Communication
- Telegram configuration

XML Element:

```

<device>
  <bus-interface-list>
    <bus-interface attributes>
      <trdp-process attributes/>
      <pd-com-parameter attributes/>
      <pd-com-parameter attributes/>
      <telegram attributes >
        :
      </telegram>
    :
  </bus-interface>
  :
</bus-interface-list>
</device>

```

XML Attributes:

All attributes that can be specified for the tag “bus-interface” are described in the following Table C.3.

Table C.3 – Attributes for bus-interface tag

Name	Data Type	Optional/Required	Description
network-id	UINT8	Required	IP interface (1..4) on the device
name	STRING	Required	Interface name
host-ip	UINT32	Optional	Host IP address
leader-ip	UINT32	Optional	Leader IP address in case of redundancy

C.4.2.2 Process Configuration

At start up the TRDP main process is spawned for sending and receiving MD and PD. The thread/task priority and, for cyclic thread/task, the cycle time may be specified in the XML configuration file. If not specified TRDP default values will be used.

XML Element:

```

<device>
  <bus-interface-list>
    <bus-interface attributes>
      <trdp-process attributes/>
      <pd-com-parameter attributes/>
      <pd-com-parameter attributes/>
      <telegram attributes >
        :
      </telegram>
      :
    </bus-interface>
    :
  </bus-interface-list>
</device>

```

The tag “trdp-process” is optional.

XML Attributes:

All attributes that can be specified are described in Table C.4:

Table C.4 – Attributes for trdp-process tag

Name	Data Type	Optional/Required	Description
cycle-time	UINT32	Optional	Cycle time—in 10^{-6} s (μ s). Only for cyclic threads/tasks. For default value see Table C.5.
blocking	STRING	Optional	NO – non – blocking, YES – blocking For default value see Table C.5.
traffic-shaping	STRING	Optional	OFF – no traffic shaping, ON – traffic shaping For default value see Table C.5.
priority	UINT32	Optional	Priority for trdp process task. Values:1-255, 1 = highest For default value see Table C.5.

Default values are defined in Table C.5:

Table C.5 – Default values for thread/task

Name	Value
cycle-time	10000
blocking	NO
traffic-shaping	ON
priority	64

C.4.2.3 PD Communication Parameters

There are a number of parameters that may be configured for the PD communication in the XML configuration file. If not specified TRDP default values will be used.

XML Element:

```

<device>
  <bus-interface-list>
    <bus-interface attributes>
      <trdp-process attributes/>
      <pd-com-parameter attributes/>
      <telegram attributes >
        :
      </telegram>
      :
    </bus-interface>
    :
  </bus-interface-list>
</device>

```

The tag “pd-com-parameter” is optional.

XML Attributes:

All attributes that can be specified for the tag “pd-com-parameter” are described in Table C.6.

Table C.6 – Attributes for pd-com-parameter tag

Name	Data Type	Optional/R equired	Description
timeout-value	UINT32	Optional	Timeout value in 10^{-6} s (μ s), before considering received process data as invalid. Disabled if 0 or if not specified. For default value see Table C.7.
validity-behaviour	STRING	Optional	Behaviour when received process data is invalid. ZERO = zero values KEEP = keep last value For default value see Table C.7.
tll	UINT32	Optional	Default time To live for PD. For default value see Table C.7.
qos	UINT32	Optional	Default quality of service for PD. For default value see Table C.7.
marshall	STRING	Optional	ON/OFF = enable/disable internal marshall/unmarshall For default value see Table C.7.

Name	Data Type	Optional/R equired	Description
callback	STRING	Optional	ON/OFF = enable/disable call back For default value see Table C.7.
port	UINT32	Optional	Port to be used for PD communication For default value see Table C.7.

Default values are defined in Table C.7:

Table C.7 – Default values for pd-com-parameter

Name	Value
timeout-value	100000
validity-behaviour	ZERO
ttl	64
qos	5
marshall	OFF
callback	OFF
port	20548

C.4.2.4 MD Communication Parameters

There are a number of parameters that may be configured for the MD communication in the XML configuration file. If not specified TRDP hard coded default values will be used.

The configured default values herein are used if nothing else is configured for the ComId.

The default time-out time for waiting for a confirm message is used when no confirm time-out value is specified for a ComId.

The default time-out time for waiting for a reply message(s) is used when no reply time-out value is specified for a ComId.

TRDP stores received sequence numbers per source IP address, to detect resend messages that already have been received. The maximum number of stored sequence numbers per IP address may be configured.

XML Element:

```

<device>
  <bus-interface-list>
    <bus-interface attributes>
      <trdp-process attributes/>
      <pd-com-parameter attributes/>
      <md-com-parameter attributes/>
      <telegram attributes >
        :
      </telegram>
      :
    </bus-interface>
    :
  </bus-interface-list>
</device>

```

The tag “md-com-parameter” is optional.

XML Attributes:

All attributes that can be specified for the tag “md-com-parameter” are described in Table C.8.

Table C.8 – Attributes for md-com-parameter tag

Name	Data Type	Optional/R equired	Description
confirm-timeout	UINT32	Optional	Default time-out time in 10^{-6} s (μ s) for receiving a confirm message For default value see Table C.9.
reply-timeout	UINT32	Optional	Default time-out time in μ s for receiving reply message(s) For default value see Table C.9.
connect-timeout	UINT32	Optional	Default time-out time in μ s to close a not used TCP connection For default value see Table C.9.
ttl	UINT32	Optional	Default time to live for MD. For default value see Table C.9.
qos	UINT32	Optional	Default quality of service for MD. For default value see Table C.9.
protocol	STRING	Optional	TCP/UDP = default protocol For default value see Table C.9.
marshall	STRING	Optional	ON/OFF = enable/disable internal marshalling/unmarshalling For default value see Table C.9.
callback	STRING	Optional	ON/OFF = enable/disable call back For default value see Table C.9.
udp-port	UINT32	Optional	Port to be used for UDP MD communication For default value see Table C.9.
tcp-port	UINT32	Optional	Port to be used for TCP MD communication For default value see Table C.9.
num-sessions	UINT32	Optional	Maximal number of replier sessions to prevent DoS attacks For default value see Table C.9.

Default values are defined in Table C.9:

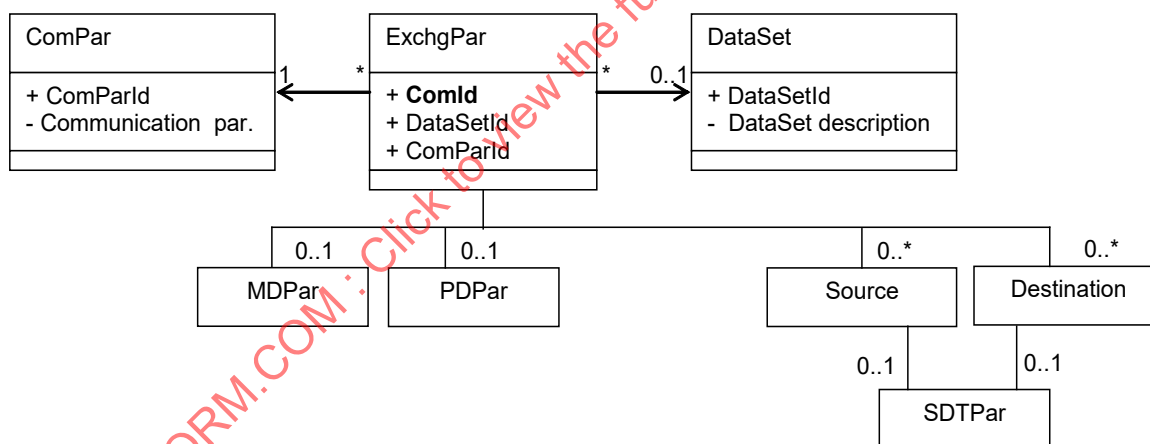
Table C.9 – Default values for md-com-parameter

Name	Value
confirm-timeout	1 000 000 µs
reply-timeout	5 000 000 µs
connect-timeout	60 000 000 µs
ttl	64
qos	3
protocol	UDP
marshall	OFF
callback	ON
udp-port	20550
tcp-port	20550
num-sessions	1000

C.4.2.5 Telegram Configuration (ExchgPar)

C.4.2.5.1 General

The telegram configuration contains the central elements for data exchange – the exchange parameters. It is identified by the central key **ComId**, see Figure C.2.



IEC

Figure C.2 – Exchange Parameters with the central key ComId

Each Exchange Parameter references one or no DataSet and one Communication Parameter. (Many Exchange Parameters can reference the same DataSet and Communication Parameter.) The Exchange Parameter also describes how to handle any Process and/or Message Data.

An Exchange Parameter is described in the XML file as a “telegram” tag.

XML Element:

```

<device>
  <bus-interface-list>
    <bus-interface attributes>
      <trdp-process attributes/>
      <pd-com-parameter attributes/>
    
```

```

<md-com-parameter attributes/>
<telegram attributes >
  <md-parameter attributes />
  <pd-parameter attributes />
  <source attributes />
  <destination attributes />
</telegram>
:
</bus-interface>
:
</bus-interface-list>
</device>

```

The tags “sdt-parameter”, “md-parameter”, “pd-parameter”, “source” and “destination” are optional.

The tags “sdt-parameter”, “md-parameter” and “pd-parameter” can only be used one time for each ComId.

For each ComId the tags “source” and “destination” may be used several times with different values.

For some attributes there are default values.

XML Attributes:

All attributes that can be specified for the telegram tag are described in Table C.10.

Table C.10 – Attributes for telegram tag

Name	Data Type	Optional/ Required	Description
name	STRING	Optional	Required for tooling and debugging
com-id	UINT32	Required	ID for exchange parameter NOTE 1 Only the first found configuration of a ComId will be considered. NOTE 2 ComId's 1-1000 are reserved for special purpose.
data-set-id	UINT32	Required for PD Optional for MD	ID for dataset to be exchanged.
com-parameter-id	UINT32	Optional	ID for communication parameter to be used, if not given the default parameters for MD/PD communication will be used

C.4.2.5.2 MD Parameters

The MD telegram configuration defines the parameters for MD telegram exchange.

XML Element:

```

<device>
  <bus-interface-list>
    <bus-interface attributes>
      :
      <telegram attributes >
        :
        <md-parameter attributes/>

```

```

      :
    </telegram>
    :
  </bus-interface>
</bus-interface-list>
</device>

```

XML Attributes:

All attributes that can be specified for the md-parameter tag are described in Table C.11.

Table C.11 – Attributes for md-parameter tag

Name	Data Type	Optional/ Required	Description
confirm-timeout	UINT32	Optional	Confirm time-out time (unit: 10^{-6} s (μ s)) Default value will be used at absence of the tag, see Table C.9.
reply-timeout	UINT32	Optional	Response time-out time (unit: 10^{-6} s (μ s)) Default value will be used at absence of the tag, see Table C.9.
marshall	STRING	Optional	ON/OFF = enable/disable internal marshalling/unmarshalling
callback	STRING	Optional	ON/OFF = enable/disable callback
protocol	STRING	Optional	UDP/TCP

C.4.2.5.3 PD Parameters

The PD telegram configuration defines the parameters for PD telegram exchange.

XML Element:

```

<device>
  <bus-interface-list>
    <bus-interface attributes>
      :
      <telegram attributes >
        :
        <pd-parameter attributes/>
        :
      </telegram>
    </bus-interface>
  </bus-interface-list>
</device>

```

XML Attributes:

All attributes that can be specified for the pd-parameter tag are described in Table C.12.

Table C.12 – Attributes for pd-parameter tag

Name	Data Type	Optional/ Required	Description
timeout	UINT32	Optional	Timeout value (unit: 10^{-6} s (μ s)), before considering received process data as invalid. Disabled if 0 or no specified value. Default value will be used at absence of the tag, see Table C.7.
validity-behaviour	STRING	Optional	Behaviour when received process data is invalid. ZERO = zero values KEEP = keep the last value Default value will be used at absence of the tag, see Table C.7.
cycle	UINT32	Required	Cycle time in 10^{-6} s (μ s), describing how often a process data should be transmitted. TRDP will round this value to a multiple of the cycle time of the TRDP process thread. Default value will be used at absence of the tag, see Table C.7.
redundant	UINT32	Optional	>0 if process data is redundant, i.e. it should be transmitted in leader mode, and not in follower mode in a redundant system. Default value = 0 (no redundancy);
marshall	STRING	Optional	ON/OFF – enable/disable internal marshall/unmarshall Default value will be used at absence of the tag, see Table C.7.
callback	STRING	Optional	ON/OFF = enable/disable callback
offset-address	UINT16	Optional	Offset-address for PD in traffic store for ladder topology

C.4.2.5.4 Source Parameters

The source configuration defines the parameters for the data source.

XML Element:

```

<device>
  <bus-interface-list>
    <bus-interface attributes>
      :
    <telegram attributes >
      :
      <source attributes>
        <sdt-parameter attributes />
      </source>
      :
    </telegram>
    :
  </bus-interface>
  :
</bus-interface-list>
</device>

```

XML Attributes:

All attributes that can be specified for the source tag are described in Table C.13.

Table C.13 – Attributes for source tag

Name	Data Type	Optional/ Required	Description
id	UINT32	Required	Source identifier.
uri1	STRING	Required	Source URI for process and message data. Used for PD filtering at receiver side to only receive data from a specific end device(s) or for MD as information provided to the receiving side. Syntax: [[user@]host] User part only required for MD. NOTE If not specified here or overridden in the subscribe call, then no filtering for PD used. If not specified here or overridden in any MD send call, the URI (host part) of the own device is used. NOTE Only URI strings that are resolved as unicast IP addresses can be used.
uri2	STRING	Optional	For process data a second source URI string can be given for source filtering, e.g. for redundant devices. Syntax: [host] NOTE If not specified here or overridden in the subscribe call, then no filtering for PD used. If not specified here or overridden in any MD send call, the URI (host part) of the own device is used. NOTE Only URI strings that are resolved as unicast IP addresses can be used.
name	STRING	Optional	Optional name for the connection

C.4.2.5.5 Destination Parameters

The source configuration defines the parameters for the data destination.

XML Element:

```

<device>
  </bus-interface-list>
  <bus-interface attributes>
    :
    <telegram attributes >
      :
      <destination attributes
        <sdt-parameter attributes />
      </destination>
      :
    </telegram>
    :
  </bus-interface>
  :
</bus-interface-list>
</device>

```

XML Attributes:

All attributes that can be specified for the source tag are described in Table C.14.

Table C.14 – Attributes for destination tag

Name	Data Type	Optional/ Required	Description
id	UINT32	Required	Destination identifier.
uri	STRING	Required	Destination URI for process and message data. Syntax: [user@]host If not specified here it has to be set/overridden in any send/publish call. The URI user part is used in the MD header frame to address a URI listener. The URI host part is used to resolve the destination IP address used for sending and at receiver side to check if the IP address is a multicast address that has to be joined.
name	STRING	Optional	Optional name for the connection

C.4.2.5.6 SDTv2 Parameters

The SDTv2 configuration defines the parameters for the SDTv2 SDSRC and SDSINK.

XML Element:

```

<device>
  <bus-interface-list>
    <bus-interface attributes>
      :
      <telegram attributes >
        :
        <source attributes
          <sdt-parameter attributes />
        </source>
        :
        <destination attributes
          <sdt-parameter attributes />
        </destination>
        :
      </telegram>
      :
    </bus-interface>
  </bus-interface-list>
</device>

```

The tag “sdt-parameter” is optional.

XML Attributes:

All attributes that can be specified for the SDTv2 are described in Table C.15.

Table C.15 – Attributes for sdt-parameter tag

Name	Data Type	Optional/ Required	Description
smi1	UINT32	Required	Safe message identifier – unique for this message at consist level
smi2	UINT32	Optional	Safe message identifier for a redundant device – unique for this message at consist level
udv	UINT16	Required	User data version

Name	Data Type	Optional/ Required	Description
rx-period	UINT16	Required	Sink cycle time, unit: 10^{-3} s (ms)
tx-period	UINT16	Required	Source cycle time, unit: 10^{-3} s (ms)
n-rxsafe	UINT8	Optional	Timeout in number of tx-period cycles
n-guard	UINT16	Optional	Guard time in number of tx-period cycles
cm-thr	UINT32	Optional	Channel monitoring threshold

Default values are defined in Table C.16:

Table C.16 – Default values for sdt-parameter tag

Name	Value
smi2	0
n-rxsafe	3
n-guard	100
cm-thr	43

C.5 Mapped Device Parameters

C.5.1 General

There might be the requirement to have for several identical devices the same configuration structure (e.g. the different door controllers of the consist). This is supported by the mapped device tag, containing the differences for the specific mapped devices.

XML Element:

```

<device>
  <device-configuration>
    :
  </device-configuration>
  <mapped-device-list>
    <mapped-device attributes>
      <mapped-bus-interface attributes>
        <mapped-telegram attributes>
          :
        </mapped-telegram attributes>
          :
        </mapped-bus-interface>
          :
        </mapped device>
          :
      </mapped-device-list>
        :
    </device>
  
```

The tags “mapped-device-list” and “mapped-device” are optional.

XML Attributes:

All attributes that can be specified for the tag “mapped-device” are described in the following Table C.17.

Table C.17 – Attributes for mapped-device tag

Name	Data Type	Optional/R equired	Description
host-name	STRING	Required	Device name
leader-name	STRING	Optional	Leader name, optional for redundant devices

C.5.2 Mapped Bus Interface Parameters

For each bus interface of a mapped device the different attributes regarding the process data, sources, destinations and related SDTv2 parameters of the telegrams can be specified.

XML Element:

```

<device>
  <device-configuration>
  :
</device-configuration>
  <mapped-device-list>
    <mapped-device attributes>
      <mapped-bus-interface attributes>
        <mapped-telegram attributes>
          <mapped-pd-parameter attributes />
          <mapped-source attributes>
            <mapped-sdt-parameter attributes />
          </mapped-source>
          :
          <mapped-destination attributes>
            <mapped-sdt-parameter attributes />
          </mapped-destination>
          :
        </mapped-telegram>
        :
      </mapped-bus-interface>
      :
    </mapped device>
  </mapped-device-list>
</device>

```

The tag “mapped-bus-interface” is optional.

The tag “mapped-telegram” is optional.

The tag “mapped-pd-parameter” is optional.

The tag “mapped-source” is optional.

The tag “mapped-destination” is optional.

The tag “mapped-sdt-parameter” is optional.

XML Attributes:

All attributes that can be specified for the tag “mapped-bus” are described in Table C.18.

Table C.18 – Attributes for mapped-bus-interface tag

Name	Data Type	Optional/R equired	Description
name	STRING	Required	Name for this interface
host-ip	UINT32	Optional	IP address for this interface
leader-ip	UINT32	Optional	Leader IP address in case of redundancy

All attributes that can be specified for the tag “mapped-telegram” are described in the following Table C.19.

Table C.19 – Attributes for mapped-telegram tag

Name	Data Type	Optional/R equired	Description
com-id	UINT32	Required	ComId of the mapped telegram
name	STRING	Optional	Optional different name for the telegram

All attributes that can be specified for the tag “mapped-pd-parameter” are described in Table C.20.

Table C.20 – Attributes for mapped-pd-parameter tag

Name	Data Type	Optional/ Required	Description
offset-address	UINT16	Optional	Offset address for PD in traffic store for ladder topology

All attributes that can be specified for the tag “mapped-source” are described in Table C.21.

Table C.21 – Attributes for mapped-source tag

Name	Data Type	Optional/Re quired	Description
Id	UINT32	Required	Identifier of the source of the telegram
uri1	STRING	Required	Source URI for process and message data. Used for PD filtering at receiver side to only receive data from a specific end device(s) or for MD as information provided to the receiving side. For process data the source URI string can include up to two comma separated URI strings, e.g. redundant devices. Syntax: [[user@]host[, host]] NOTE If not specified here or overridden in the subscribe call, then no filtering for PD used. If not specified here or overridden in any MD send call, the URI (host part) of the own device is used. NOTE Only URI strings that are resolved as unicast IP addresses can be used.
uri2	STRING	Optional	For process data a second source URI string can be given for source filtering, e.g. for redundant devices. Syntax: [host] NOTE If not specified here or overridden in the subscribe call, then no filtering for PD used. If not specified here or overridden in any MD send call, the URI (host part) of the own device is used.

Name	Data Type	Optional/Required	Description
			NOTE Only URI strings that are resolved as unicast IP addresses can be used.
name	STRING	Optional	Optional different name for the connection

All attributes that can be specified for the tag “mapped-destination” are described in Table C.22.

Table C.22 – Attributes for mapped-destination tag

Name	Data Type	Optional/Required	Description
Id	UINT32	Required	Identifier of the destination of the telegram
uri	STRING	Required	Destination URI for process and message data. Syntax: [user@]host If not specified here it has to be set/overridden in any send/publish call. The URI user part is used in the MD header frame to address a URI listener. The URI host part is used to resolve the destination IP address used for sending and at receiver side to check if the IP address is a multicast address that has to be joined.
name	STRING	Optional	Optional different name for the connection

All attributes that can be specified for the tag “mapped-SDTv2 parameters” are described in Table C.23.

Table C.23 – Attributes for mapped-SDTv2-parameter tag

Name	Data Type	Optional/Required	Description
smi1	UINT32	Required	Required for SDTv2 telegrams.
smi2	UINT32	Optional	Required for redundant SDTv2 telegrams.

C.6 Communication Parameters (ComPar)

C.6.1 General

The communication parameter information describes in which way the communication should take place. In most cases, one set of parameters is sufficient but it is possible to specify specific communication parameters for special situations.

A Communication Parameter is described in the XML file as a “com-parameter” tag.

XML Element:

```

<device>
  <com-parameter-list>
    <com-parameter attributes/>
    :
  </com-parameter-list>
</device>

```

XML Attributes:

All attributes that can be specified for the tag are described in Table C.24.

Table C.24 – Attributes for com-parameter tag

Name	Data Type	Optional/ Required	Description
id	UINT32	Required	ID for communication parameter NOTE Only the first configuration of a com-parameter-id will be considered.
qos	UINT32	Required	Quality of Service, what priority level data should be sent with, 0..7
ttl	UINT32	Optional	Time To Live, how many jumps a message should live, 0..255, default = 64.
retries	UINT32	Optional	Number of retries used for MD request

C.6.2 Default Communication Parameters

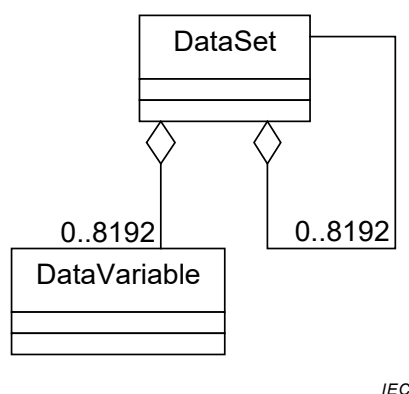
Most communication can be done with a small set of communication parameters. To simplify for the TRDP user there are always two sets of default communication parameters available as listed in Table C.25.

Table C.25 – Default communication parameters

Com-parameter-id	qos	time-to-live	retries	Description
1	5	64	0	Suitable for PD communication
2	3	64	3	Suitable for MD communication

C.7 DataSet Parameters**C.7.1 General**

Data communication over the ETB is done with DataSets (Figure C.3).



IEC

Figure C.3 – DataSet structure

A DataSet is a container of data items, like a *structure* in C language. A DataSet can contain up to 8192 DataVariables or other DataSets. A *DataVariable* is data of one of the basic data types defined in Table C.26.

Table C.26 – Basic data types

Type name	Type #	Description
BOOL8	1	=UINT8, 1 bit relevant (equal zero → false, not equal zero → true)
BITSET8		=UINT8, 8 bit relevant
ANTIVALENT8		=UINT8, 2 bit relevant
CHAR8	2	char, can be used also as UTF-8
UTF16	3	Unicode UTF-16 character
INT8	4	Signed integer, 8 bit
INT16	5	Signed integer, 16 bit
INT32	6	Signed integer, 32 bit
INT64	7	Signed integer, 64 bit
UINT8	8	Unsigned integer, 8 bit
UINT16	9	Unsigned integer, 16 bit
UINT32	10	Unsigned integer, 32 bit
UINT64	11	Unsigned integer, 64 bit
REAL32	12	Floating point real, 32 bit
REAL64	13	Floating point real, 64 bit
TIMEDATE32	14	32 bit UNIX time
TIMEDATE48	15	48 bit TCN time (32 bit seconds and 16 bit ticks)
TIMEDATE64	16	32 bit seconds and 32 bit microseconds

Each DataVariable or DataSet can be single or multiple instances. Multiple instances (arrays) can be of fix or variable size.

The maximum depth of a dataset within a dataset is 5.

The DataSet configuration information describes all datasets to be used in data exchange. Many Exchange Parameters can use one single DataSet.

The DataSet describes the structure of the data, i.e. which data variables and other datasets it contains. By use of this information TRDP can send and receive data in such a way that it complies with the data formats required on the TCN. Applications do not have to do any further adjustment to be able to use the data.

A DataSet is described in the XML file as a “data-set” tag. Inside this there are “element” tags.

XML Element:

```
<device>
  <data-set-list>
    <data-set attributes >
      <element attributes />
      :
    </data-set>
    :
  </data-set-list>
</device>
```

XML Attributes:

All attributes that can be specified for the data-set tags are described in Table C.27.

Table C.27 – Attributes for data-set tag

Name	Data Type	Optional/ Required	Description
name	STRING	Optional	Required for tooling and debugging
id	UINT32	Required	ID for dataset NOTE 1 Only the first found configuration of a data-set-id will be considered. NOTE 2 Data set id's 1-1000 are reserved for special purpose.

C.7.2 DataSet Element

C.7.2.1 General

To be able to perform marshalling of datasets, all end devices need information about structure and content of the datasets. This information is stored in a Dataset Formatting Table in the configuration database. This table describes for each data item of the DataSet its *type* and *array size*. For visualization purposes each element can have the additional attributes unit, scale and offset.

Each element of a DataSet can be either a DataVariable or DataSet. Up to 8192 elements are possible.

XML Element:

```
<device>
  <data-set-list>
    <data-set attributes >
      <element attributes />
      :
    </data-set>
    :
  </data-set-list>
</device>
```

XML Attributes:

All attributes that can be specified for an element tag are described in Table C.28.

Table C.28 – Attributes for element tag

Name	Data Type	Optional/ Required	Description
name	STRING	Optional	Required for tooling and debugging
type	STRING	Required	Data type of data variable (BOOL8, BITSET8, ..., see basic data types in chapter C.7.1) or dataset identifier (1..)
array-size	UINT32	Optional	0 = Array with dynamic size 1 = Single >1 Number of instances in array Default = 1
unit	STRING	Optional	Unit text for visualisation purposes.
scale	FLOAT	Optional	Factor for visualisation purposes ($val = scale \times x + offset$).
offset	INT32	Optional	Offset for visualisation purposes ($val = scale \times x + offset$).

C.7.2.2 DataSet Element Type

If the type is another DataSet, the *Type* is the ID of this dataset.

For a DataVariable the *Type* tells the marshalling software the size of the data but also how to treat it. A 4-byte string is treated differently as 4-byte integer or real. The basic Datatype constants currently available are described in C.7.1.

The table describes all data types that can be used on the TCN. For IEC 61131-3 software only a sub-set of types with fixed size can be used.

NOTE The C++ type "BOOL" is often not resolved into a UINT8 variable by the compiler because of that here BOOL8 is defined and used.

C.7.2.3 DataSet Element array-size

The *array-size* describes the number of instances for the data item as listed in Table C.29.

Table C.29 – Use of element array size

Array-size	Instances
1	Single
2..n	Array, fixed size 2..n
0	Array, dynamic size

A dynamic sized array is preceded by a UINT8, UNIT16 or UINT32, containing current size of the array, i.e. number of items in the array (not number of bytes). See example in C.7.3.5.

Dynamic sized arrays can only be used for message data and not for process data.

C.7.3 Examples of DataSets

C.7.3.1 General

The examples below show different types of datasets with example of included DataVariables and DataSets, single and multiple instances, fixed or variable.

C.7.3.2 DataSet with DataVariables

This dataset contains single instances of basic data types.

C Declaration	Dataset Formatting Table		
	Type	Number	Comment
<pre>struct DS57 { UINT32 a; UINT8 b; INT16 c; REAL32 d; };</pre>			
	UINT32	1	Single
	UINT8	1	Single
	INT16	1	Single
	REAL32	1	Single

XML configuration example:

```
<device>
  <data-set-list>
    <data-set id="57" >
      <element type="UINT32" array-size="1" name="a" />
      <element type="UINT8" array-size="1" name="b" />
      <element type="INT16" array-size="1" name="c" />
      <element type="REAL32" array-size="1" name="d" />
    </data-set>
  :
</data-set-list>
</device>
```

C.7.3.3 DataSet with other DataSet

In this example a dataset DS3 contains another dataset DS2.

To specify that a dataset contain another dataset, the column for Type is set to the ID of that dataset. The marshalling software will recognize this by the positive value. Codes for basic data types are negative.

C Declaration	Dataset Formatting Table		
	Type	Number	Comment
<pre>struct DS2 { UINT32 a1; INT32 b1; INT32 c1; };</pre>			
	UINT32	1	Single
	INT32	1	Single
	INT32	1	Single
<pre>struct DS3 { UINT32 a; INT32 b; struct DS2 c; };</pre>			
	UINT32	1	Single
	INT32	1	Single
	ID of DS2	1	Single

XML configuration example:

```
<device>
  <data-set-list>
    <data-set id="2" >
      <element type="UINT32" array-size="1" name="a1" />
      <element type="INT32" array-size="1" name="b1" />
      <element type="INT32" array-size="1" name="c1" />
    </data-set>
    <data-set id="3" >
      <element type="UINT32" array-size="1" name="a" />
      <element type="INT32" array-size="1" name="b" />
      <element type="2" array-size="1" name="c" />
    </data-set>
  :
</data-set-list>
</device>
```

C.7.3.4 DataSet with Fixed Sized Arrays

In this example, a dataset contains multiple instances of DataVariables and DataSets, i.e. arrays of data.

C Declaration	Dataset Formatting Table		
	Type	Number	Comment
<pre>struct DS4 { INT32 a[17]; struct DS2 b[3]; };</pre>			
	INT32	17	Array of integers
	ID of DS2	3	Array of datasets

The first data item is an array of 17 instances of INT32;

The second data item is an array of 3 instances of dataset DS2.

XML configuration example:

```
<device>
  <data-set-list>
    <data-set id="2" >
      <element type="UINT32" array-size="1" name="a1" />
      <element type="INT32" array-size="1" name="b1" />
    </data-set>
  :
</data-set-list>
</device>
```

```

        <element type="INT32" array-size="1" name="c1" />
    </data-set>
    <data-set id="4" >
        <element type="INT32" array-size="17" name="a" />
        <element type="2" array-size="3" name="b" />
    </data-set>
    :
</data-set-list>
</device>

```

C.7.3.5 DataSet with Dynamic sized arrays

In this example a dataset contains instances of arrays, but the size of the arrays is not predefined in the configuration database but dynamically specified at runtime.

A dynamically sized array is preceded with a size of type UINT8, UINT16 or UINT32 that contains the current size of the array. These shall be loaded with current size at run-time. Size should be set to number of items in the array.

Dynamic size of arrays means that there is no information in the configuration database about the size of the array.

NOTE Dynamic sized datasets can only be used in MD communication, not in PD communication.

Declaration	Dataset Formatting Table		
	Type	Number	Comment
<pre> struct DS5 { UINT32 aSize; INT32 b[5]; }; </pre>			
	UINT32	1	Size of array b (=5)
	INT32	0	

Declaration	Dataset Formatting Table		
	Type	Number	Comment
<pre> struct DS6 { UINT32 aSize; INT32 b[5]; UINT16 cSize; struct DS2 d[9]; }; </pre>			
	UINT32	1	Size of array b (=5)
	INT32	0	
	UINT16	1	Size of array d (=9)
	ID of DS2	0	

XML configuration example:

```

<device>
  <data-set-list>
    <data-set id="2" >
      <element type="UINT32" array-size="1" name="a1" />
      <element type="INT32" array-size="1" name="b1" />
      <element type="INT32" array-size="1" name="c1" />
    </data-set>
    <data-set id="5" >
      <element type="UINT32" array-size="1" name="a" />
      <element type="INT32" array-size="0" name="b" />
    </data-set>
    <data-set id="6" >
      <element type="UINT32" array-size="1" name="a" />
      <element type="INT32" array-size="0" name="b" />
    </data-set>
  </data-set-list>
</device>

```

```

        <element type="UINT32" array-size="1" name="c" />
        <element type="2" array-size="0" name="d" />
    </data-set>
    :
</data-set-list>
</device>

```

The example of structure for data set 6 in the figure above is not very useful unless there are several similar structures used for sending with the same dataset ID. When this type of structure is used where a dynamic array is followed by any other variable type the size variables has to be set to exactly the size of the corresponding array. See the code example below for a more useful way to use this type of dynamic dataset. Another possibility, as for data set 5, is to only use one dynamic array in the end of the dataset then the size variable can be set to the current used size of the array. Code example for dataset 6:

```

char buf[2000];
UINT16 *p;
UINT32 len;

p = (UINT16 *) buf;          /* Set pointer to start of buffer */
*p++ = currASize;           /* Store current size of array a */
memcpy(p, aBuffer, currASize * sizeof(INT32));
p = (char *) p + currASize * sizeof(INT32); /* Move pointer */

*p++ = currBSize;           /* Store current size of array b */
memcpy(p, bBuffer, currBSize * sizeof(struct DS2));
p = (char *) p + currBSize * sizeof(struct DS2);

len = (char *) p - buf;      /* length of buffer, no. of bytes */

```

C.7.3.6 DataSet for Transparent Communication

In some cases applications want to exchange data without any interference of marshalling software, e.g for uploading of recorded binary data. This can be achieved by sending a dataset that contains only one data item: a dynamic sized array of bytes, unsigned 8 bit integers or by sending a comId without any specified dataset. As mentioned before dynamic data set can only be sent as message data.

Declaration	Dataset Formatting Table		
	Type	Number	Comment
<pre> struct DS5 { UINT32 bufSize; UINT8 buf[n]; }; </pre>			
	UINT32	1	Size of array buf (=n)
	UINT8	0	Array of bytes

The sending buffer `buf` shall be preceded with the size of the buffer to send, loaded with the number of items in `buf`, i.e. 'n'.

Annex D (informative)

Access to End Device (ED) statistics

D.1 General

This annex describes the access to statistics data of end devices implementing TRDP.

D.2 Structures

D.2.1 General

TRDP memory statistic and configuration information is defined as follows:

```
TRDP_MEM_STATISTICS_T := RECORD
{
    total            UINT32      -- total memory size
    free            UINT32      -- free memory
    minFree         UINT32      -- minimal free memory in statistics
                           interval
    allocBlocks     UINT32      -- number of allocated blocks
    allocErr        UINT32      -- number of allocation errors
    freeErr         UINT32      -- number of freeing errors
    blockSize       UINT32[15]  -- if used, predefined memory block sizes
    usedBlockSize   UINT32[15]  -- if used, number of used memory blocks
                           for each block size
}
```

TRDP process data statistic and configuration information is defined as follows:

```
TRDP_PD_STATISTICS_T := RECORD
{
    defQos          UINT32      -- default QoS for PD
    defTtl          UINT32      -- default TTL for PD
    defTimeout      UINT32      -- Default timeout for PD
    numSubs         UINT32      -- Number of subscribed ComId's
    numPub          UINT32      -- Number of published ComId's
    numRcv          UINT32      -- number of received PD packets
    numCrcErr       UINT32      -- number of received PD packets with
                           CRC err
    numProtErr      UINT32      -- number of received PD packets with
                           protocol err
    numTopoErr      UINT32      -- number of received PD packets with
                           wrong topo count
    numNoSubs       UINT32      -- number of received PD push packets
                           without subscription
    numNoPub        UINT32      -- Number of received PD pull packets
                           without publisher
    numTimeout      UINT32      -- number of PD timeouts
    numSend         UINT32      -- number of sent PD packets
}
```

TRDP message data statistic and configuration information is defined as follows:

```
TRDP_MD_STATISTICS_T ::= RECORD
{
    defQos                UINT32    -- default QoS for MD
    defTtl                UINT32    -- default TTL for MD
    defReplyTimeout       UINT32    -- Default reply timeout for MD
    defConfirmTimeout     UINT32    -- Default confirm timeout for MD
    numList               UINT32    -- Number of Listeners
    numRcv                UINT32    -- number of received MD packets
    numCrcErr             UINT32    -- number of received MD packets with
                                CRC err
    numProtErr            UINT32    -- number of received MD packets with
                                protocol err
    numTopoErr            UINT32    -- number of received MD packets with
                                wrong topo count
    numNoListener         UINT32    -- number of received MD packets without
                                listener
    numReplyTimeout       UINT32    -- number of MD reply timeouts
    numConfirmTimeout     UINT32    -- number of MD confirm timeouts
    numSend               UINT32    -- number of sent MD packets
}
```

TRDP general statistic and configuration information is defined as follows:

```
TRDP_STATISTICS_T ::= RECORD
{
    version               UINT32    -- TRDP version
    timestamp             UINT64    -- actual time stamp
    uptime                UINT32    -- time in seconds since last
                                initialization
    statisticTime         UINT32    -- time in seconds since last reset
                                of statistics
    hostname              CHAR[16]  -- own host name
    leaderName            CHAR[16]  -- leader host name
    ownIpAddr             UINT32    -- own IP address
    leaderIpAddr          UINT32    -- leader IP address
    processPrio           UINT32    -- priority of TRDP process
    processCycle          UINT32    -- cycle time of TRDP process
                                (unit: 10-6 s (µs))
    numJoin               UINT32    -- number of joins
    numRed                UINT32    -- number of redundancy groups
    mem                   TRDP_MEM_STATISTICS_T -- memory statistics
    pd                    TRDP_PD_STATISTICS_T  -- PD statistics
    udpMd                 TRDP_MD_STATISTICS_T  -- MD statistics
    tcpMd                 TRDP_MD_STATISTICS_T  -- MD statistics
}
```


D.2.2 tlc_getSubsStatistics

TRDP process data subscription statistic information is defined as follows:

```
TRDP_SUBS_STATISTICS_T := RECORD
{
  comId          UINT32    -- subscribed ComId
  joinedAddr     UINT32    -- joined IP address
  filterAddr     UINT32    -- filter IP address, i.e IP address of
                           the sender for this subscription,
                           0.0.0.0 in case all senders.
  callBack       UINT32    -- reference for call back function if used
  timeout        UINT32    -- time-out value (unit: 10-6 s (µs)).
                           0 = No time-out supervision
  status         UINT32    -- receive status information TRDP_NO_ERR,
                           TRDP_TIMEOUT_ERR
  toBehav        UINT32    -- behaviour at time-out.
                           set data to zero / keep last value
  numRecv        UINT32    -- number of packets received for this
                           subscription
}
```

D.2.3 tlc_getPubStatistics

TRDP process data publish statistic information is defined as follows:

```
TRDP_PUB_STATISTICS_T := RECORD
{
  comId          UINT32    -- subscribed ComId
  destAddr       UINT32    -- IP address of destination for this
                           publishing
  cycle          UINT32    -- publishing cycle (unit: 10-6 s (µs))
  redId          UINT32    -- redundancy group id
  redState       UINT32    -- redundant state:
                           !=0 = Leader or 0 = Follower
  numPut         UINT32    -- number of packet updates
  numSend        UINT32    -- number of packets sent for this publish
}
```

D.2.4 tlc_getUdpListStatistics, tlc_getTcpListStatistics

TRDP message data listener statistic information is defined as follows:

```
TRDP_LIST_STATISTICS_T := RECORD
{
  comId          UINT32    -- subscribed ComId
  uri            CHAR[32]  -- URI to listen to (user part)
  joinedAddr     UINT32    -- joined IP address
  callback       UINT32    -- call back function reference if used
  queue          UINT32    -- queue reference if used
  userRef        UINT32    -- user reference if used
  numRecv        UINT32    -- number of received packets
}
```

D.2.5 tlc_getRedStatistics

TRDP redundancy statistic information is defined as follows:

```
TRDP_RED_STATISTICS_T ::= RECORD
{
  id          UINT32    -- redundancy id
  state       UINT32    -- leader/follower
}
```

D.3 ED interface for statistic data access

D.3.1 General

TRDP statistic data can be retrieved either by:

- SNMP, as defined in Clause G.3
- TRDP interface as defined below

D.3.2 TRDP interface

The TRDP interface for retrieving statistic data from EDs is shown in Figure D.1.

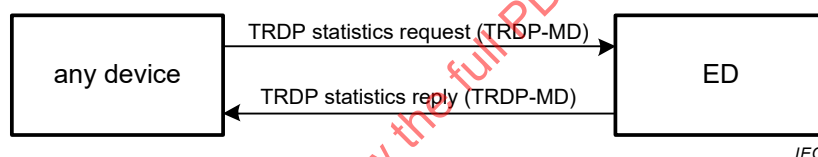


Figure D.1 – TRDP statistics data telegrams

TRDP MD Request telegram parameters:

MsgType:	'Mr'
ComId:	As defined below
SourceURI (user part):	user defined
DestinationURI (user part):	void
DestinationURI (host part):	selected ED
SourceURI (host part):	user defined
Dataset:	TRDP_STATISTICS_REQUEST
Reply timeout:	3,0 s

TRDP MD Reply telegram parameters:

MsgType: 'Mp'

ComId: As defined below

Dataset: TRDP_STATISTICS_REPLY

TRDP-MD telegram payload:

```
TRDP_STATISTICS_REQUEST ::= RECORD
{
    -- payload empty
}

TRDP_STATISTICS_REPLY ::= ONE_OF
{
    [GLOBAL]      TRDP_STATISTICS_T           -- comId = 30 (request) 31 (reply)
    [SUBS]        TRDP_SUBS_STATISTICS_T      -- comId = 32 (request) 33 (reply)
    [PUB]         TRDP_PUB_STATISTICS_T       -- comId = 34 (request) 35 (reply)
    [RED]         TRDP_RED_STATISTICS_T       -- comId = 36 (request) 37 (reply)
    [LIST_UDP]    TRDP_LIST_STATISTICS_T      -- comId = 38 (request) 39 (reply)
    [LIST_TCP]    TRDP_LIST_STATISTICS_T      -- comId = 40 (request) 41 (reply)
}
```

Annex E (informative)

Service interface

E.1 General

This annex defines a communication interface between the services providers defined in this part of the standard (DNS Server, TTDB Manager, ECSP and ETBN) and its clients.

An overview of the different interfaces is given in Figure E.1. Those interfaces are:

- ECSP interface. This interface is provided by the ECSP and supports the ETB control service. There exists only one ECSP per consist and the interface is between the ECSP and a dedicated client (ECSC) which is entitled to control the communication on the operational ETB (ETB0).
- TTDB Manager interface. The TTDB manager provides this interface and supports read access to the TTDB. The TTDB manager should be available within each consist network.
- DNS server interface. This interface is provided by the DNS server and supports the TCN-DNS service. A DNS server should be available within each consist network.
- ETBN control interface. This interface is provided by the ETBN and supports ETBN control and access to ETB status information, refer to IEC 61375-2-5. An ETBN is available within each consist network.

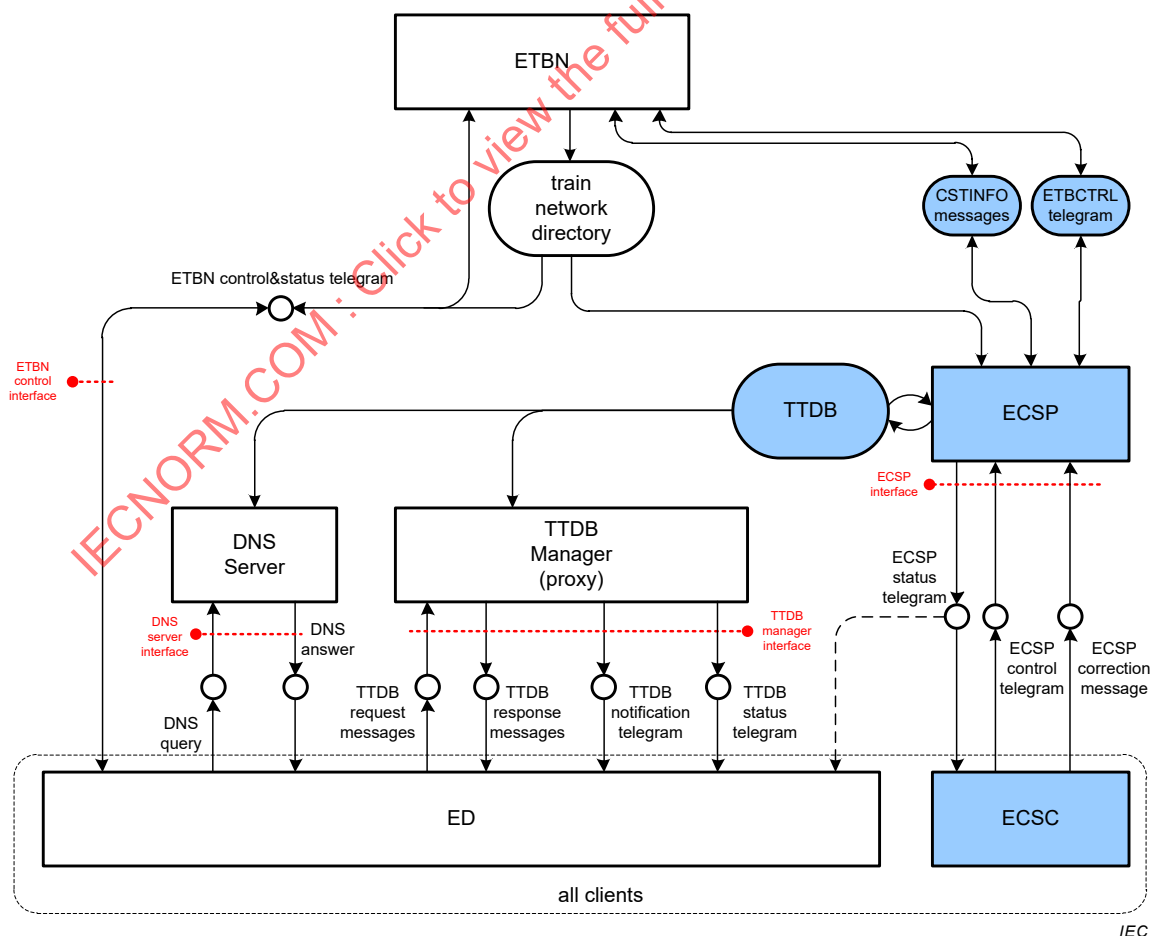


Figure E.1 – Service interfaces block diagram

E.2 Service provider

E.2.1 Proxies

In order to prevent IP routing of service requests in consists with multiple consist networks, there is a possibility to install local instances of a service provider or local proxies to a central service provider. The implementation of local instances or local proxies is a choice of the consist designer.

NOTE Services provided by the ETBN do not require a proxy as there is an ETBN connected to each CN.

E.2.2 Performance

A user of a service based on a TRDP MD request-reply pattern needs to know how fast the server will reply in order to set the TRDP parameter value 'ReplyTimeOut' correspondingly. This reply time not only depends on the server itself, but also on the performance of the network. Therefore it is recommended that the consist designer defines the minimum values.

EXAMPLE For retrieving the operational train directory from the local TTDB manager, the consist designer defines a minimum reply time of 1,5 s.

E.3 ECSP interface

E.3.1 General

The ECSP interface defines the interaction between the ECSC device and the ECSP for the functions which are defined in Clause 6.

The telegrams exchanged between ECSC and ECSP are shown in Figure E.2.

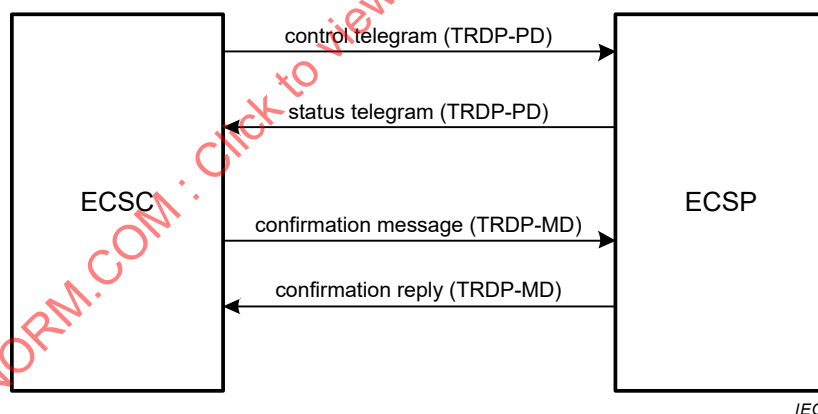


Figure E.2 – ECSP interface telegrams

E.3.2 ECSP control telegram

TRDP PD telegram parameters:

MsgType:	'Pd'
ComId:	120
SMI:	120 (only needed if SDTv2 is used)
UserDataVersion:	'0100'H (only needed if SDTv2 is used)

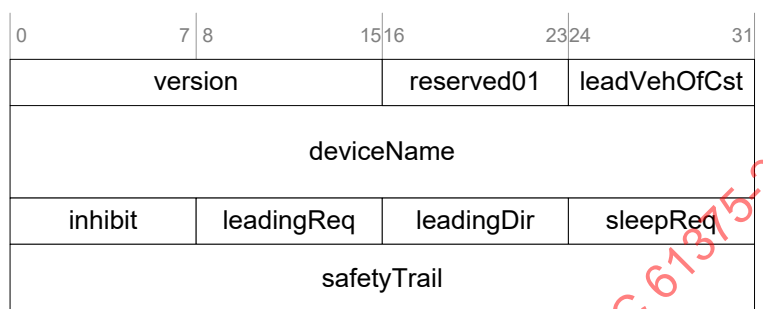
DestinationURI (host part): devECSP.anyVeh.ICst.ICITrn.ITrn

Dataset: ECSP_CTRL

Cycle time: $(1,0 \pm 0,1)$ s

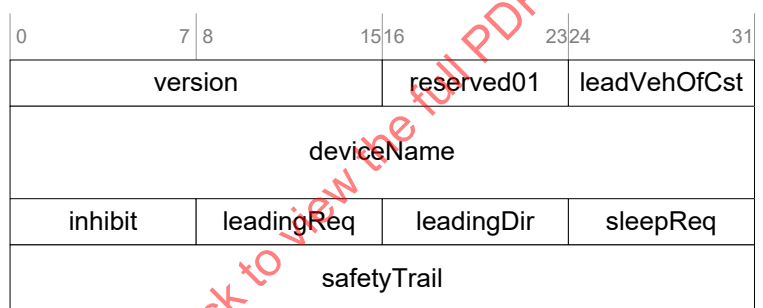
Timeout: 5,0 s

The TRDP telegram is defined as shown in



IEC

Figure E.3.



IEC

Figure E.3 – ECSP control data

The payload of the TRDP telegram is defined as:

```

ECSP_CTRL ::= RECORD
{
    version          UINT16          -- telegram version information
                                   (vvrr, v-version, r-release)
                                   version = 1
                                   release = 0

    reserved01       UINT8           -- reserved (= 0)

    leadVehOfCst     UINT8           -- position of leading vehicle in
                                   consist value range 0..32
                                   0 = not defined
                                   (1 = first vehicle in consist in
                                   Direction 1, 2 = second vehicle,
                                   etc.)

    deviceName       LABEL           -- function device of ECSC which sends
                                   the telegram

    inhibit          UINT8           -- inauguration inhibit
                                   0 = no inhibit request
                                   1 = inhibit request

    leadingReq       UINT8           -- leading request
    
```

```

                                0 = no leading request
                                1 = leading request
leadingDir      UINT8      -- leading direction
                                0 = no leading request
                                1 = leading request direction 1
                                2 = leading request direction 2
sleepReq        UINT8      -- sleep request
                                0 = no sleep request
                                1 = sleep request
safetyTrail     ETBCTRL_VDP -- ETB-VDP trailer
                                completely set to 0 = SDTv2 not used
}

```

NOTE The control telegram contains also parameters related to the ETBN, like for instance parameter 'inhibit'. For this interface, it is assumed that the ECSC interfaces only with the ECSP and that ETBN related information is transferred from the ECSP to the ETBN (ECSP acting as proxy). However, it is also possible to use the direct interface to the ETBN instead (see Clause E.6).

E.3.3 ECSP status telegram

TRDP PD telegram parameters:

MsgType: 'Pd'

ComId: 121

SMI: 121 (only needed if SDTv2 is used)

UserDataVersion: '0100'H (only needed if SDTv2 is used)

DestinationURI (host part): devECSC.anyVeh.ICst.ICITrn.ITrn

Dataset: ECSP_STATUS

Cycle time: $(1,0 \pm 0,1)$ s

Timeout: User defined

If the receiving ECSC detects a timeout, the ECSC shall inform the consist control application that the ECSP failed.

The TRDP telegram is defined as shown in Figure E.4.

0	7	8	15	16	23	24	31
version				reserved			
lifesign				ecspState		etbInhibit	
etbLength		etbShort		reserved			
etbLeadState		etbLeadDir		ttdbSrvState		dnsSrvState	
trnDirState		opTrnDirState		sleepCtrlState		slRequCnt	
opTrnTopoCnt							
safetyTrail							

IEC

Figure E.4 – ECSP status data

The payload of the TRDP telegram is defined as:

IECNORM.COM : Click to view the full PDF of IEC 61375-2-3:2015

ECSP_STATUS ::= RECORD

```

{
  version          UINT16      -- telegram version information
                                (vvrr, v-version, r-release)
                                version = 1
                                release = 0

  reserved01       UINT16      -- reserved (= 0)

  lifesign          UINT16      -- wrap-around counter, incremented with
                                each produced telegram.

  ecspState         UINT8       -- ECSP state indication
                                0 = ECSP not operational(initial value)
                                1 = ECSP in operation

  etbInhibit        UINT8       -- inauguration inhibit indication
                                0 = n/a (default)
                                1 = inhibit not requested on ETB
                                2 = inhibit set on local ETBN
                                3 = inhibit set on remote ETBN
                                4 = inhibit set on local and remote
                                  ETBN

  etbLength         UINT8       -- indicates train lengthening in case
                                train inauguration is inhibit
                                0 = no lengthening (default)
                                1 = lengthening detected

  etbShort          UINT8       -- indicates train shortening in case
                                train inauguration is inhibit
                                0 = no shortening (default)
                                1 = shortening detected

  reserved02       UINT16      -- reserved (= 0)

  etbLeadState      UINT8       -- indication of local consist leadership
                                5 = consist not leading (initial value)
                                6 = consist is leading requesting
                                9 = consist is leading
                                10 = leading conflict
                                other values are not allowed

  etbLeadDir        UINT8       -- direction of the leading end vehicle
                                in the local consist
                                0 = not relevant
                                1 = TCN direction 1
                                2 = TCN direction 2
                                other values are not allowed

  ttDbSrvState      UINT8       -- TTDB server state indication
                                0 = n/a (initial value)
                                1 = Leader (default)
                                2 = Follower
                                3 = Error

  dnsSrvState       UINT8       -- DNS server state indication
                                0 = n/a (initial value)
                                1 = Leader (default)
                                2 = Follower
                                3 = Error

  trnDirState       UINT8       -- train directory state
                                1 = UNCONFIRMED
                                2 = CONFIRMED
                                other values are not allowed

  opTrnDirState     UINT8       -- operational train directory state
                                1 = INVALID

```

```

                2 = VALID
                4 = SHARED
                other values are not allowed
sleepCtrlState  UINT8      -- sleep control state (option)
                        0 = option not available
                        1 = RegularOperation
                        2 = WaitForSleepMode
                        3 = PrepareForSleepMode
sleepReqCnt     UINT8      -- number of sleep requests (option)
                        value range: 0..63
                        not used = 0
opTrnTopoCnt    UINT32     -- operational train topography counter
safetyTrail     ETBCTRL_VDP -- ETB-VDP trailer
                        completely set to 0 = SDTv2 not used
}

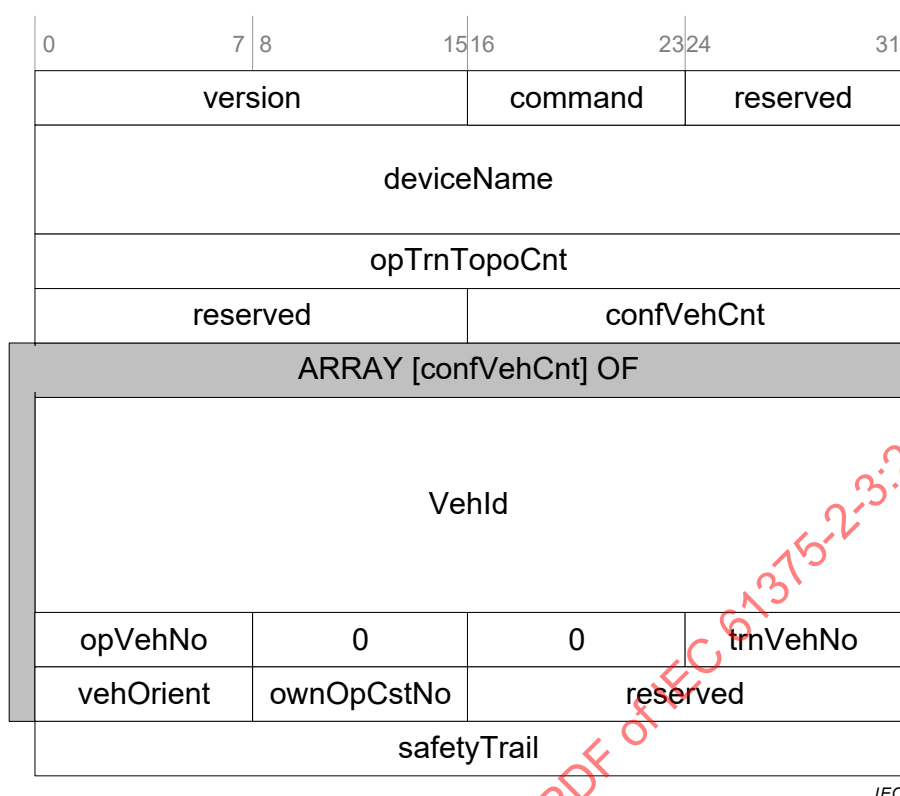
```

E.3.4 ECSP Confirmation/Correction Request

TRDP MD Request message telegram parameters:

MsgType:	'Mr'
ComId:	122
SMI:	122 (only needed if SDTv2 is used)
UserDataVersion:	'0100'H (only needed if SDTv2 is used)
SourceURI (user part):	user defined
DestinationURI (user part):	void
DestinationURI (host part):	devECSP.anyVeh.ICst.ICITrn.ITrn
SourceURI (host part):	user defined
Dataset:	ECSP_CONF_REQUEST
Reply timeout:	3,0 s

The TRDP message is defined as shown in Figure E.5.

**Figure E.5 – ECSP confirm/correction request data**

The payload of the TRDP message is defined as:

```

ECSP_CONF_REQUEST ::= RECORD
{
  version          VERSION          -- telegram version information
                                   main version = 1
                                   subversion = 0

  command          UINT8            -- confirmation order
                                   1 = confirmation/correction request
                                   2 = un-confirmation request

  reserved01       UINT8            -- reserved (= 0)

  deviceName       LABEL            -- function device of ECSC which sends
                                   the telegram

  opTrnTopoCnt     UINT32           -- operational train topocounter value
                                   of the operational train directory the
                                   correction is based on

  reserved02       UINT16           -- reserved (= 0)

  confVehCnt       UINT16           -- number of confirmed vehicles in the
                                   train (1..63).
                                   if set to 0: no correction

  confVehList      ARRAY [confVehCnt] OF OP_VEHICLE
                                   -- ordered list of
                                   confirmed vehicles in the train,
                                   starting with vehicle at train head,
                                   see chapter 5.3.3.2.10.
                                   Parameters 'isLead' and 'leadDir'
                                   to be set to 0.

  safetyTrail      ETBCTRL_VDP     -- ETB-VDP trailer
                                   completely set to 0 = SDTv2 not used
}

```

TRDP MD Reply telegram parameters:

MsgType:	'Mp'
Protocol:	UDP
ComId:	123
SMI:	123 (only needed if SDTv2 is used)
UserDataVersion	'0100'H (only needed if SDTv2 is used)
Dataset:	ECSP_CONF_REPLY

The TRDP message is defined as shown in Figure E.6.

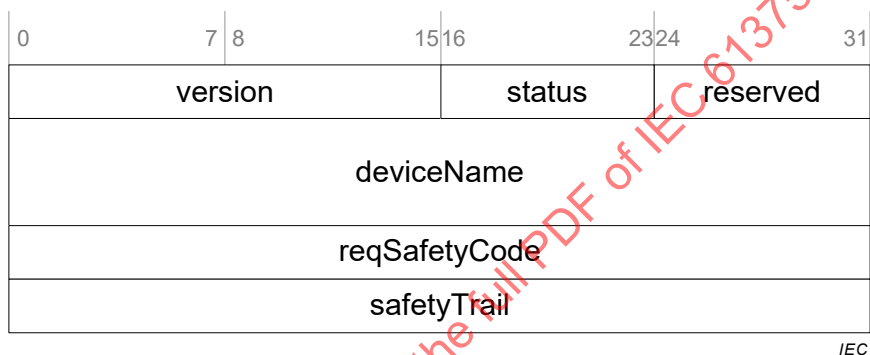


Figure E.6 – ECSP confirm/correction reply data

The payload of the TRDP message is defined as:

```

ECSP_CONF_REPLY ::= RECORD
{
    version          VERSION          -- telegram version information
                                   main version = 1
                                   subversion = 0

    status           UINT8            -- status of storing correction info
                                   0 = correctly stored
                                   1 = not stored

    reserved01       UINT8            -- reserved (= 0)

    deviceName       LABEL            -- function device of ECSC which sends
                                   the telegram

    reqSafetyCode    UINT32           -- SC-32 value of the request message

    safetyTrail      ETBCTRL_VDP      -- ETB-VDP trailer
                                   completely set to 0 = SDTv2 not used
}
    
```

E.4 TTDB manager interface

E.4.1 General

The TTDB manager interface defines the TRDP telegrams, which are used to retrieve data from the TTDB. A TRDP-PD telegram is used to inform about the actual operational train directory status, a notification message is sent after each change of the operational train directory. In addition, a set of TRDP-MD telegrams can be used to retrieve data from the TTDB as it is shown in Figure E.7.

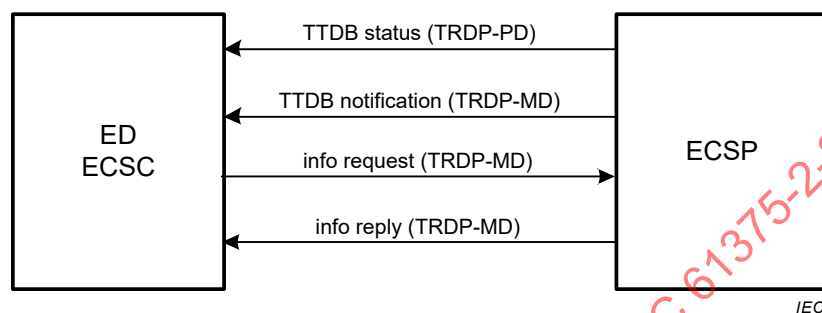


Figure E.7 – TTDB manager interface telegrams

E.4.2 TTDB status information

PD push telegram with operational train status information, pushed from TTDB manager (or a proxy of thereof) to all devices of the consist.

TRDP PD telegram parameters:

MsgType:	'Pd'
ComId:	100
SMI:	100 (only needed if SDTv2 is used)
UserDataVersion:	'0100'H (only needed if SDTv2 is used)
DestinationURI (host part):	grpAll.aVeh.ICst.ICITrn.ITrn
Dataset:	TTDB_OP_TRAIN_DIRECTORY_STATUS_INFO
Cycle time	(1,0 ± 0,1) s
Timeout:	5,0 s

The payload of the TRDP telegram is defined as:

```

TTDB_OP_TRAIN_DIRECTORY_STATUS_INFO ::= RECORD
{
    state                OP_TRAIN_DIRECTORY_STATE
                        -- operational train directory state
    etbTopoCnt           UINT32
                        -- ETB topography counter value of the ETB
                        the sender of this telegram is
                        connected to (e.g. ETB0)
}

```

```

ownOpCstNo      UINT8      -- own operational consist number
                        value range: 1..63
ownTrnCstNo     UINT8      -- own train consist number
                        value range: 1..63
reserved02      UINT16     -- reserved for future use (= 0)
safetyTrail     ETBCTRL_VDP -- ETB-VDP trailer
                        completely set to 0 = SDTv2 not used
}

```

E.4.3 TTDB notification

MD notification message with dynamic train information, pushed from ECSP to all devices of the consist after each TCN inauguration.

TRDP MD Notification telegram parameters:

```

MsgType:          'Mn'
Protocol:          UDP
ComId:             101
SourceURI (user part): user defined
DestinationURI (user part): void
DestinationURI (host part): grpAll.aVeh.Cst.ICITrn.ITrn
SourceURI (host part): user defined
Dataset:           TTDB_OP_TRAIN_DIRECTORY_INFO

```

The payload of the TRDP message is defined as:

```

TTDB_OP_TRAIN_DIRECTORY_INFO ::= RECORD
{
    opTrnDir      OP_TRAIN_DIRECTORY
                        -- operational train directory
}

```

E.4.4 TTDB information – train directory

TRDP-MD telegram to retrieve train directory information of a given ETB.

TRDP MD Request telegram parameters:

```

MsgType:          'Mr'
Protocol:          UDP
ComId:             102
SourceURI (user part): user defined

```

DestinationURI (user part): void

DestinationURI (host part): devECSP.anyVeh.ICst.ICITrn.ITrn

SourceURI (host part): user defined

Dataset: TTDB_TRAIN_DIRECTORY_INFO_REQUEST

Reply timeout: 3,0 s

The payload of the TRDP message is defined as:

```
TTDB_TRAIN_DIRECTORY_INFO_REQUEST ::= RECORD
{
    etbId      UINT8          -- identification of the ETB the train
                               directory information is requested for
                               0 = ETB0 (operational network)
                               1 = ETB1 (multimedia network)
                               2 = ETB2 (other network)
                               3 = ETB3 (other network)
}
```

TRDP MD Reply telegram parameters:

MsgType: 'Mp'

ComId: 103

Dataset: TTDB_TRAIN_DIRECTORY_INFO_REPLY

The payload of the TRDP message is defined as:

```
TTDB_TRAIN_DIRECTORY_INFO_REPLY ::= RECORD
{
    trnDir      TRAIN_DIRECTORY -- dynamic train info
}
```

E.4.5 TTDB information – static consist information

TRDP-MD telegram to retrieve static consist information of a given consist. Because of the possible large quantity of this information it is recommended to use TRDP-MD on TCP for this transmission.

TRDP MD Request message telegram parameters:

MsgType: 'Mr'

Protocol: UDP

ComId: 104

SourceURI (user part): user defined

DestinationURI (user part): void

DestinationURI (host part): devECSP.anyVeh.ICst

SourceURI (host part): user defined

Dataset: TTDB_STATIC_CONSIST_INFO_REQUEST

Reply timeout: 3,0 s

The payload of the TRDP message is defined as:

```
TTDB_STATIC_CONSIST_INFO_REQUEST ::= RECORD
{
    cstUUID      UINT8[16]          -- consist unique identification
}
```

TRDP MD Reply message telegram parameters:

MsgType: 'Mp'

ComId: 105

Dataset: TTDB_STATIC_CONSIST_INFO_REPLY

The payload of the TRDP message is defined as:

```
TTDB_STATIC_CONSIST_INFO_REPLY ::= RECORD
{
    cstInfo      CONSIST_INFO        -- consist information
}
```

E.4.6 TTDB information – train network directory information

TRDP-MD telegram to retrieve train network directory information of a given ETB.

NOTE It is also possible to read the train network directory directly from the local ETBN.

TRDP MD Request message telegram parameters:

MsgType: 'Mr'

Protocol: UDP

ComId: 106

SourceURI (user part): user defined

DestinationURI (user part): void

DestinationURI (host part): devECSP.anyVeh.ICst

SourceURI (host part): user defined

Dataset: TTDB_TRAIN_NETWORK_DIRECTORY_INFO_REQUEST

Reply timeout: 3,0 s

The payload of the TRDP message is defined as:

```
TTDB_TRAIN_NETWORK_DIRECTORY_INFO_REQUEST ::= RECORD
{
    etbId      UINT8          -- identification of the ETB the train
                                network directory information is
                                requested for
                                0 = ETB0 (operational network)
                                1 = ETB1 (multimedia network)
                                2 = ETB2 (other network)
                                3 = ETB3 (other network)
}
```

TRDP MD Reply message telegram parameters:

MsgType: 'Mp'

ComId: 107

Dataset: TTDB_TRAIN_NETWORK_DIRECTORY_INFO_REPLY

The payload of the TRDP message is defined as:

```
TTDB_TRAIN_NETWORK_DIRECTORY_INFO_REPLY ::= RECORD
{
    trnNetDir  TRAIN_NETWORK_DIRECTORY  -- dynamic ETB info
}
```

E.4.7 Operational train directory information

TRDP-MD telegram to retrieve operational train directory information of a given ETB.

TRDP MD Request message telegram parameters:

MsgType: 'Mr'

Protocol: UDP

ComId: 108

SourceURI (user part): user defined

DestinationURI (user part): void

DestinationURI (host part): devECSP.anyVeh.ICst

}

TRDP MD Reply message telegram parameters:

MsgType: 'Mp'

ComId: 111

Dataset: TTDB_READ_COMPLETE_REPLY

The payload of the TRDP message is defined as:

TTDB_READ_COMPLETE_REPLY ::= RECORD

```

{
  state          OP_TRAIN_DIRECTORY_STATE
                                     -- operational train directory state

  opTrnDir       OP_TRAIN_DIRECTORY   -- operational train directory
  trnDir         TRAIN_DIRECTORY      -- train directory
  trnNetDir      TRAIN_NETWORK_DIRECTORY -- train network directory
}

```

E.5 DNS server interface**E.5.1 DNS standard interface**

The DNS server standard interface is specified in RFC 1034 and RFC 1035.

E.5.2 DNS TCN interface**E.5.2.1 General**

Instead of using the DNS standard interface, the DNS TCN interface can be used which is more efficient when multiple TCN-URIs need to be resolved.

TCN-URI address resolution is requested by sending a DNS resolving request message. The TCN-DNS will resolve the TCN-URIs contained within the DNS resolving request message and will return a list with related IP addresses.

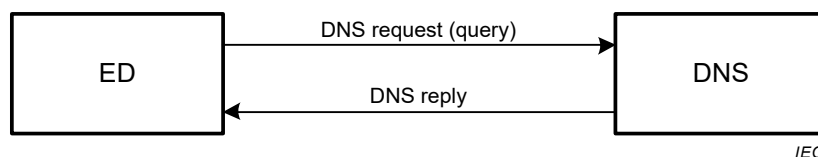


Figure E.8 – TCN-URI resolving

E.5.2.2 TCN_URI Record

The host part of a TCN-URI is defined in accordance to 5.4.4 as a string of the following form:

```
fctdevLabel.vehLabel.cstLabel.cltrnLabel.trnLabel
```

Hence the maximal length of host part character string is $(5 \times 15) + 4 + \text{'0'} = 80$ character (ASCII NUL terminated string).

For the DNS TCN interface a simplified resource record, called TCN_URI record is defined:

```
TCN_URI ::= RECORD
{
    tcnUriStr          CHAR[80]          -- Host part of TCN-URI, terminated with
                                         ASCII character NUL
    reserved01         INT16             -- reserved (= 0)
    resolvState        INT16             -- request message: reserved (= 0)
                                         reply message:
                                         0 = OK
                                         -1 = unknown TCN-URI
    tcnUriIpAddr       UINT32            -- IP address related to query
                                         unicast address (source and destination
                                         scope, address range) or
                                         MC group address (destination scope)
                                         value = 0 if 'resolvState' < 0
    tcnUriIpAddr2      UINT32            -- 2nd IP address in case the query
                                         produces an IP address range
                                         (see 5.4.4.6.2).
                                         Defines the higher IP address of the
                                         range.
                                         To be set to 0 if not resolved to an
                                         IP address range
}
```

This record is filled in the DNS query (DNS resolving request message) with the TCN-URI string and will be complemented with the related IP address by the DNS server (resolver) in the reply (DNS resolving reply message).

E.5.2.3 DNS resolving request message

The request for resolving TCN-URI is defined as:

TRDP MD Request message telegram parameters:

MsgType:	'Mr'
Protocol:	UDP
ComId:	140
SMI:	140 (only needed if SDTv2 is used)
UserDataVersion:	'0100'H (only needed if SDTv2 is used)
SourceURI (user part):	user defined

DestinationURI (user part): Void

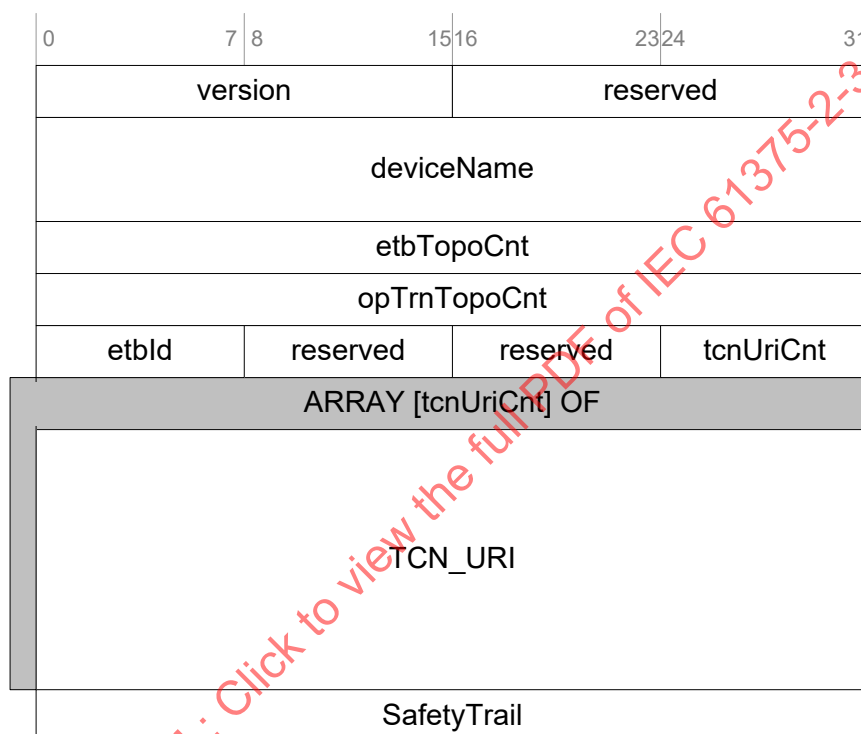
DestinationURI (host part): devDNS.anyVeh.ICst.ICITrn.ITrn

SourceURI (host part): user defined

Dataset: DNS_REQUEST

Reply timeout: 3,0 s

The TRDP message is defined as shown in



IEC

Figure E.9.

}

TRDP MD Reply telegram parameters:

MsgType:	'Mp'
ComId:	141
SMI:	141 (only needed if SDTv2 is used)
UserDataVersion	'0100'H (only needed if SDTv2 is used)
Dataset:	DNS_REPLY

The TRDP message is defined as shown in

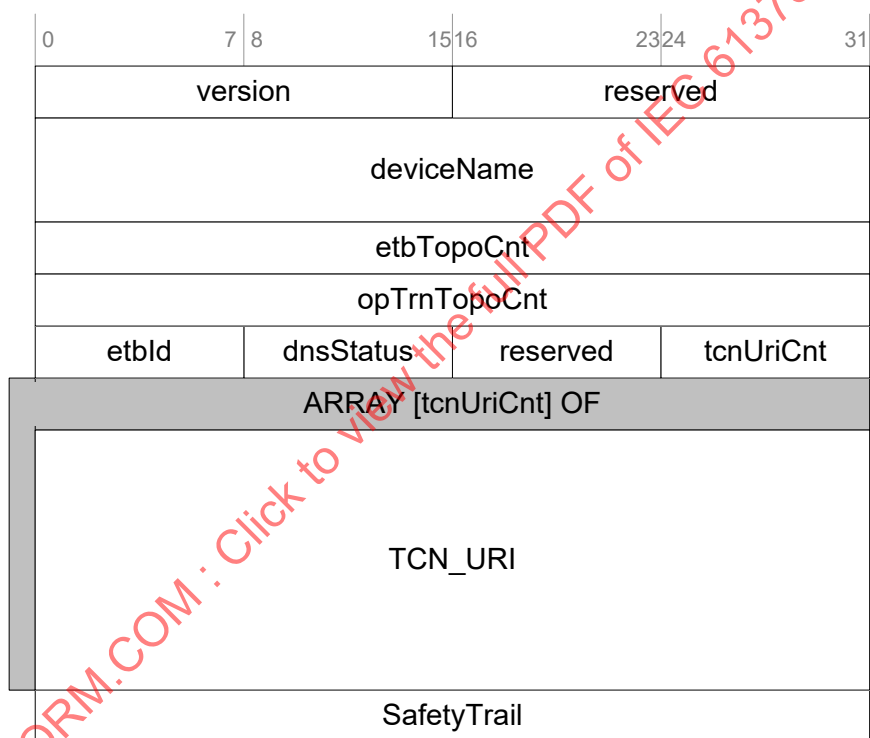
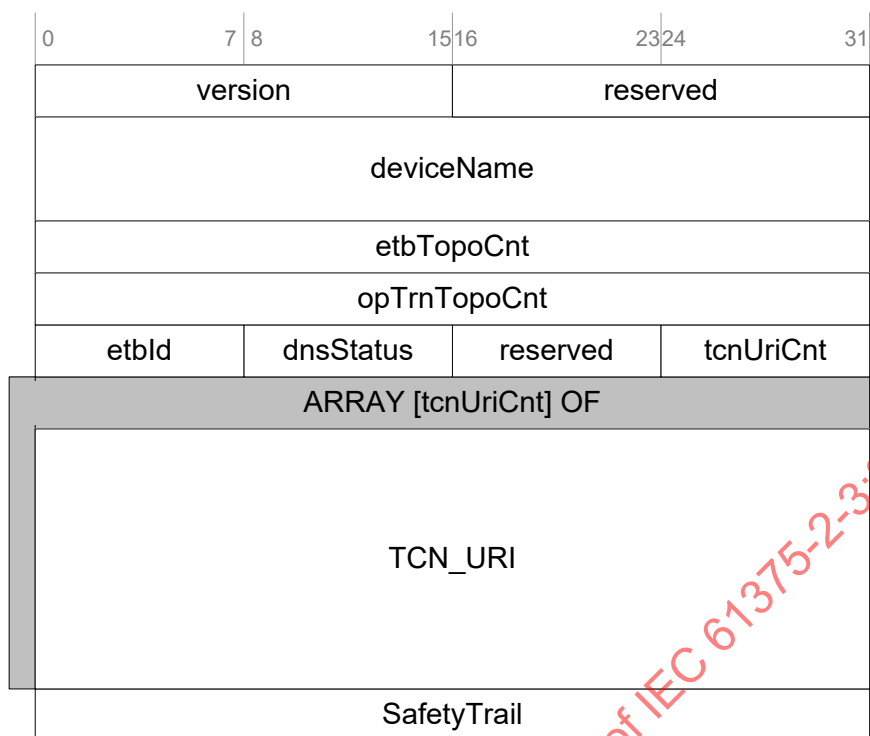


Figure E.10.

IEC



IEC

Figure E.10 – DNS resolving reply data

IECNORM.COM : Click to view the full PDF of IEC 61375-2-3:2015

The payload of the TRDP message is defined as:

```

DNS_REPLY ::= RECORD
{
  version          VERSION          -- telegram version information
                                   mainVersion = 1
                                   subVersion = 0

  reserved01       UINT16           -- reserved (= 0)

  deviceName       LABEL            -- function device of DNS which sends
                                   the telegram

  etbTopoCnt       UINT32           -- ETB topography counter

  opTrnTopoCnt     UINT32           -- operational train topography counter
                                   = 0 if not known

  • etbId          UINT8            -- identification of the related ETB
                                   0 = ETB0 (operational network)
                                   1 = ETB1 (multimedia network)
                                   2 = ETB2 (other network)
                                   3 = ETB3 (other network)

  dnsStatus        INT8            -- status of reply:
                                   0 = OK
                                   -1 = DNS Server not ready
                                   -2 = Inauguration in progress

  reserved02       UINT8           -- reserved (= 0)

  tcnUriCnt        UINT8           -- number of resolved TCN-URIs
                                   value range: 0 .. 255

  tcnUriList       ARRAY[tcnUriCnt] OF TCN_URI
                                   -- List of TCN_URI records

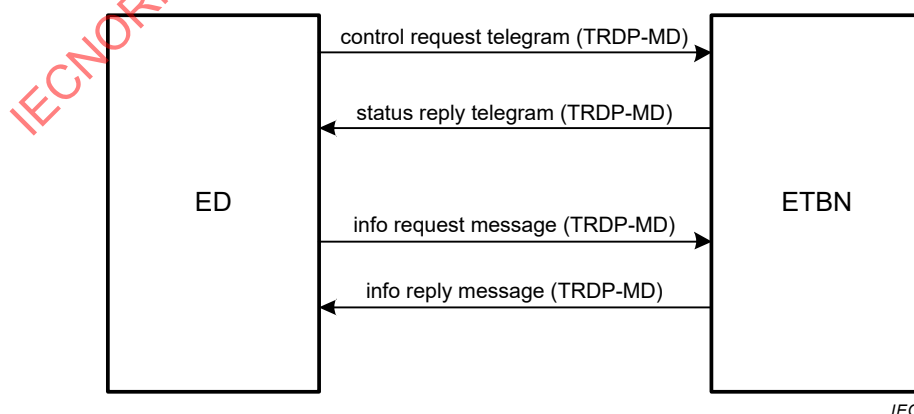
  safetyTrail      ETBCTRL_VDP     -- ETB-VDP trailer
                                   completely set to 0 = SDTv2 not used
}

```

E.6 ETBN control interface

E.6.1 General

The ETBN control interface defines the TRDP telegrams which are used to control the ETBN and to read status information from the ETBN. Additionally it provides telegrams for reading information from the ETBN as it is shown in Figure E.11.



IEC

Figure E.11 – ETBN control interface telegrams

E.6.2 ETBN control and status data

TRDP MD Request telegram parameters:

MsgType:	'Mr'
Protocol	UDP
ComId:	130
SMI:	130 (only needed if SDTv2 is used)
UserDataVersion:	'0100'H (only needed if SDTv2 is used)
SourceURI (user part)	void
DestinationURI (user part)	void
DestinationURI (host part):	a specific ETBN in the consist
Dataset:	ETBN_CTRL
Reply timeout:	3,0 s

The TRDP message is defined as shown in Figure E.12.

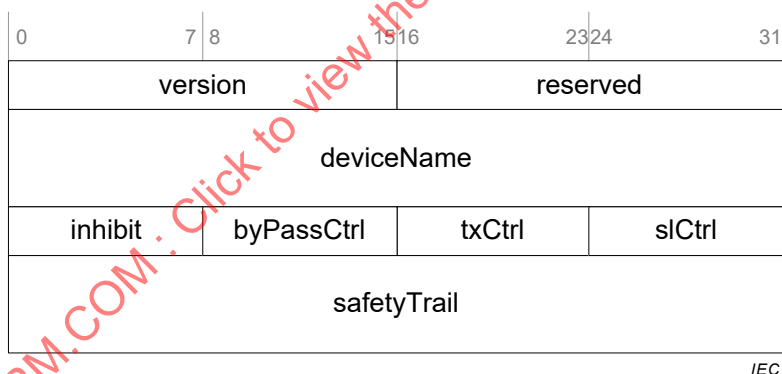


Figure E.12 – ETBN control request data

The payload of the TRDP message is defined as:

```

ETBN_CTRL ::= RECORD
{
    version          VERSION          -- telegram version information
                                   mainVersion = 1
                                   subVersion = 0

    reserved01       UINT16           -- reserved (= 0)

    deviceName       LABEL            -- function device of ED which sends
                                   the telegram

    inhibit          UINT8            -- ETBN inhibit
                                   0 = no action (keep old state)
                                   1 = no inhibit request
                                   2 = inhibit request

    byPassCtrl       UINT8            -- ETBN bypass control
    
```

```

                                0 = no action (keep old state)
                                1 = no bypass
                                2 = activate bypass
txCtrl          UINT8          -- ETBN transmission control
                                0 = no action (keep old state)
                                1 = activate sending on ETB (default)
                                2 = stop sending on ETB
slCtrl          UINT8          -- sleep mode control (option)
                                0 = no action (keep old state)
                                1 = deactivate sleep mode
                                2 = activate sleep mode (line
                                activity sensing)
safetyTrail     ETBCTRL_VDP    -- ETB-VDP trailer
                                completely set to 0 = SDTv2 not used
}

```

TRDP MD Reply telegram parameters:

MsgType: 'Mp'

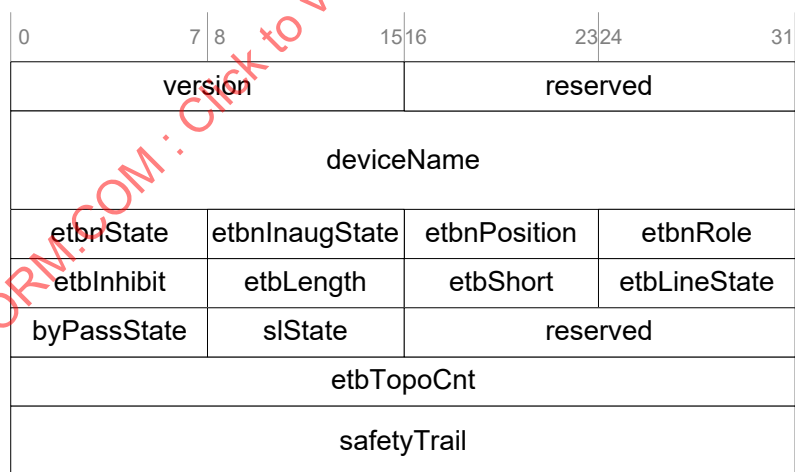
ComId: 131

SMI: 131 (only needed if SDTv2 is used)

UserDataVersion: '0100'H (only needed if SDTv2 is used)

Dataset: ETBN_STATUS

The TRDP message is defined as shown in Figure E.13.



IEC

Figure E.13 – ETBN status reply data

The payload of the TRDP message is defined as:

```

ETBN_STATUS ::= RECORD
{
  version          VERSION          -- telegram version information
                                version = 1
                                release = 0
  reserved01       UINT16          -- reserved (= 0)
}

```

deviceName	LABEL	-- function name of ETBN
etbnState	UINT8	-- state indication of the (active) ETBN 0 = ETBN not operational(initial value) 1 = ETBN in operation
etbnInaugState	UINT8	-- ETBN inauguration state as defined in IEC 61375-2-5 0 = init 1 = not inaugurated 2 = inaugurated 3 = ready for inauguration
etbnPosition	UINT8	-- position of the ETBN 0 = unknown (default) 1 = single node 2 = middle node 3 = end node TCN direction 1 4 = end node TCN direction 2
etbnRole	UINT8	-- ETBN node role as defined in IEC 61375-2-5 0 = undefined 1 = master (redundancy leader) 2 = backup (redundancy follower) 3 = not redundant
etbInhibit	UINT8	-- inauguration inhibit indication 0 = n/a (default) 1 = inhibit not requested on ETB 2 = inhibit set on local ETBN 3 = inhibit set on remote ETBN 4 = inhibit set on local and remote ETBN
etbLength	UINT8	-- indicates train lengthening in case train inauguration is inhibit 0 = no lengthening (default) 1 = lengthening detected
etbShort	UINT8	-- indicates train shortening in case train inauguration is inhibit 0 = no shortening (default) 1 = shortening detected
etbLineState	BITSET8	-- indication of ETB line status (FALSE == not trusted, TRUE == trusted) bit0 = line A ETBN direction 1 bit1 = line B ETBN direction 1 bit2 = line C ETBN direction 1 bit3 = line D ETBN direction 1 bit4 = line A ETBN direction 2 bit5 = line B ETBN direction 2 bit6 = line C ETBN direction 2 bit7 = line D ETBN direction 2
byPassState	UINT8	-- state of bypass function 0 = bypass disabled 1 = bypass enabled
slState	UINT8	-- sleep mode state (option) 0 = no sleep mode 1 = sleep mode active (line activity sensing)
reserved02	UINT16	-- reserved (= 0)
etbTopoCnt	UINT32	-- ETB topography counter

```

safetyTrail      ETBCTRL_VDP    -- ETB-VDP trailer
                                completely set to 0 = SDTv2 not used
}

```

NOTE As defined in IEC 61375-2-5 ETB lines 'C' and 'D' are optional.

E.6.3 ETBN train network directory

TRDP-MD telegram to retrieve train network directory information from ETBN.

TRDP MD Request message telegram parameters:

MsgType:	'Mr'
Protocol	UDP
ComId:	132
SMI:	132 (only needed if SDTv2 is used)
UserDataVersion:	'0100'H (only needed if SDTv2 is used)
SourceURI (user part):	user defined
DestinationURI (user part):	void
DestinationURI (host part):	a specific ETBN in the consist
SourceURI (host part):	user defined
Dataset:	void
Reply timeout:	3,0 s

TRDP MD Reply message telegram parameters:

MsgType:	'Mp'
ComId:	133
SMI	133 (only needed if SDTv2 is used)
UserDataVersion:	'0100'H (only needed if SDTv2 is used)
Dataset:	ETBN_TRAIN_NETWORK_DIRECTORY_INFO_REPLY

The payload of the TRDP message is defined as:

```

ETBN_TRAIN_NETWORK_DIRECTORY_INFO_REPLY ::= RECORD
{
    trnNetDir    TRAIN_NETWORK_DIRECTORY    -- dynamic train info
}

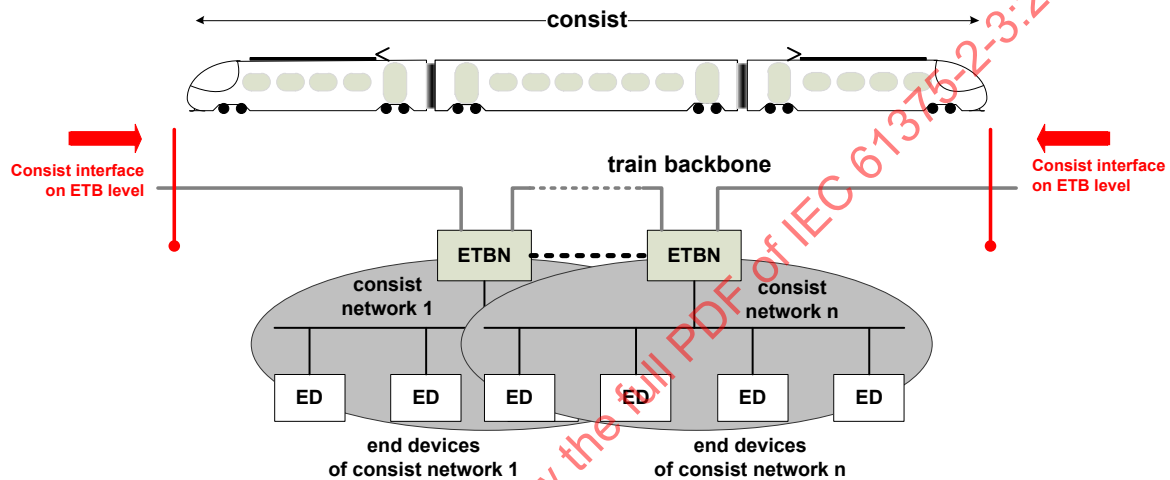
```

Annex F (normative)

Communication profile conformance test guideline

F.1 General

This annex provides a methodological framework for conformance and interoperability test of consists connected on ETB. The conformance test checks the conformity to the protocols specified in this part of IEC 61375 by stimulating the consist interface on ETB level as shown in Figure F.1. Conformance to the communication profile is one of the building blocks necessary to achieve interoperability between consists on ETB level.



IEC

Figure F.1 – Consist interface on ETB level

More precisely, the conformity test objective is to test:

- Conformity with respect to the services of the communication profile as defined within this part of the standard.
- Conformity to TRDP as a protocol used on ETB for data exchange between abstract functions.

NOTE 1 The difference between “conformance test” and “interoperability test” is that with the conformance test a UuT is tested against the protocol specification, while for interoperability testing two devices implementing the protocol are connected in order to test whether these devices are able to properly exchange and interpret data. Hence, a properly performed conformance test will supersede an interoperability test, while an interoperability test alone will not be sufficient to prove conformity to a protocol.

NOTE 2 It is not in the scope of this standard to provide a guideline of testing conformity of end devices to functions or services specified in this part of the standard, e.g. for testing TRDP conformity of end devices.

F.2 Scope of conformance test

A complete conformance test of consists connected on ETB encompasses both communication functions as defined in this part of the standard as well as functions defined in IEC 61375-2-4 (application profile). There is however an overlapping with respect to end-to-end communication protocols like TRDP and SDTv2, which can specifically only be tested in conjunction with the application functions using them (Figure F.2). Performing a generic conformance test for those protocols without involvement of the specific application can therefore never provide complete conformance test coverage.

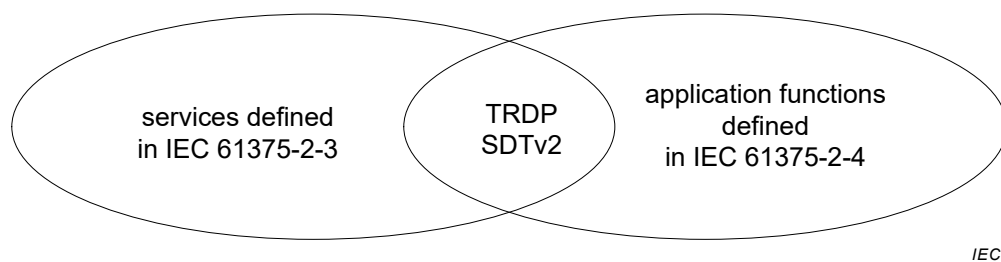


Figure F.2 – Scope of conformance test

Consequently, performing a generic conformance test for those protocols as specified in this part of the standard will be a necessary, but not sufficient condition for a complete conformance test.

F.3 Conformance test overview

Subject of conformance testing are:

- ETB control service, as defined in Clause 6
- TCN-DNS service, as defined in 5.5
- Train info service, as defined in 5.8
- TRDP, as defined in Annex A
- SDTv2, as defined in Annex B

F.4 Test laboratory

F.4.1 General

Conformance tests shall only be executed by a laboratory operating in accordance to ISO/IEC 17025.

An accreditation of the test laboratory executed by an independent accreditation body that operates in accordance with ISO/IEC 17011 is recommended.

F.4.2 Tasks

The test laboratory shall take responsibility for the following activities:

- Establish a process for TCN communication profile conformance testing.
- Provide a conformance test specification.
- Provide a test bed for conformance testing.
- Provide instructions for UuT conformance test preparation.
- Execute conformance test sessions on agreement between test laboratory and the UuT provider.
- Prepare a detailed conformance test report.
- In case of a positive test result: provide a certificate stating conformity of the UuT to this part of the standard.

F.5 Guideline for writing conformance test specifications

F.5.1 Overview of the main components

The main components of a test specification include:

- an identification of the base standard(s) to which the test specification applies;
- an identification of the relevant Protocol Implementation Conformance Statement (PICS) proforma which describes standards requirements and the supplier's support for these requirements, e.g. with respect to supported options;
- a specification of the abstract test architecture;
- a Protocol Implementation eXtra Information for Testing (PIXIT) proforma which provides additional information regarding parameters and values or their ranges;
- a test suite structure identifying each test that is to be defined and a grouping of the tests into a structure which is appropriate to the particular UuT;
- a test purpose for each identified test. A test purpose consists of:
 - a unique test purpose identifier;
 - a status specifying whether the test purpose is mandatory or optional according to the associated PICS;
 - references (usually by (sub)section number and paragraph number) to the requirements in the base standard(s) covered by the test purpose;
 - a description of the purpose of the test in plain language;
- for each test a test description which comprises:
 - a unique test identifier;
 - a brief description of the objective of the test, making reference to the test purpose included;
 - references to the requirements in the base standard(s) covered by the test (taken directly from the associated test purpose);
 - references to the individual PICS and PIXIT items covered by the test;
 - pre-conditions describing the static configuration of the test scenario;
 - the test procedure in a series of steps written in plain or structured language. This should include any preamble to establish the UuT into a known initial condition prior to the test and postamble to return it to a specific quiescent state after the completion of the test;
 - for each step, a description of the results which constitute a PASS condition.

As their name implies, guidelines are only for guidance and the actual process followed should use and adapt whichever of these guidelines are most applicable in each particular situation. In some cases this may mean the application of all aspects.

F.5.2 Protocol Implementation Conformance Statement (PICS)

The purpose of a PICS is to identify those standardized functions which an UuT shall support, those which are optional and those which are conditional on the presence of other functions. Although not strictly part of the TCN communication profile test suite, the PICS helps to provide a means for selection of the suite of tests which will subsequently be developed.

In addition, the PICS can be used as a proforma by a TCN communication profile implementation supplier in identifying which functions an UuT will support when interoperating with similar equipment from other suppliers.

The information in a PICS is usually presented in tabular form as recommended in ISO/IEC 9646-7.

The PICS can be considered as a set of “switches” which specify the capability of supporting the requirement in base standards to be tested. It is possible that with different choices in a PICS proforma, several different set of TPs will be necessary.

F.5.3 Abstract test architecture

An abstract test architecture provides a general framework within specific test arrangements which must fit in order to perform the specified suite of tests. It describes the entities involved with the protocol being tested and their means of communication, for example, the definition and placement of UuT, test components and test drivers.

Abstract test architectures can be depicted through diagrams or equivalent tables. An example abstract test architecture is provided in Clause F.6.

F.5.4 Protocol Implementation eXtra Information for Testing (PIXIT)

A Protocol Implementation eXtra Information for Testing (PIXIT) proforma identifies which PICS items are to be tested and which parameters to be instantiated for the TSS&TP being developed.

The production of a PIXIT Proforma is specified in ISO/IEC 9646-6. A supplier, providing an UuT for conformance testing, is required to complete a PIXIT proforma, which contains additional questions that need to be answered in order to turn on/off the “switches” and identify means of testing for a particular UuT.

F.5.5 Test suite structure

There is no hard and fast rule that can be used to determine how a test suite should be divided up into test groups other than to say that there should be a logical basis to the choice of groups. In many cases, the division will be rather arbitrary and based on the preferences of the author(s). However, the following categorizations should be considered when identifying appropriate test groups within a test suite structure:

- Abstract test configuration: A test group for each valid configuration specified. For example:
 - UuT as end node in varying train compositions;
 - UuT as intermediate node in varying train compositions.
- Functionality: A test group for each of the major functions (and sub-functions, when necessary) supported. For example:
 - CSTINFO telegram exchange;
 - Computation of TTDB;
 - ETBCTRL packet exchange;
 - ETB control – function leading;
 - ETB control – function sleep mode;
 - ETB control – function confirmation/correction;
 - TCN-DNS service.
- Success, inclusive or failure: test groups for normal behavior, inclusive behavior and exceptional behavior.

F.6 Abstract test architecture (option)

F.6.1 General

The following test setups are examples only, but it is recommended to follow these examples in real implementations because they define a standard way of TCN communication profile conformance testing.

F.6.2 Test architecture with one ETB

A principal abstract architecture is shown in Figure F.3. Here, the UuT is connected to the tester only via one ETB and a power connector, which allows to switch the UuT on/off. This principal abstract architecture is suited for an automated test execution.

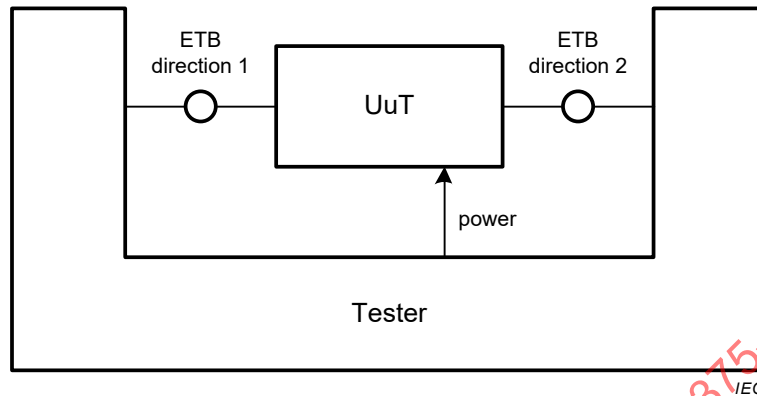


Figure F.3 – Abstract test architecture (1 ETB)

F.6.3 Test architecture for multiple ETB

The test architecture described in F.6.2 can be extended for supporting multiple ETBs, as in the example of Figure F.4 shown for two ETBs.

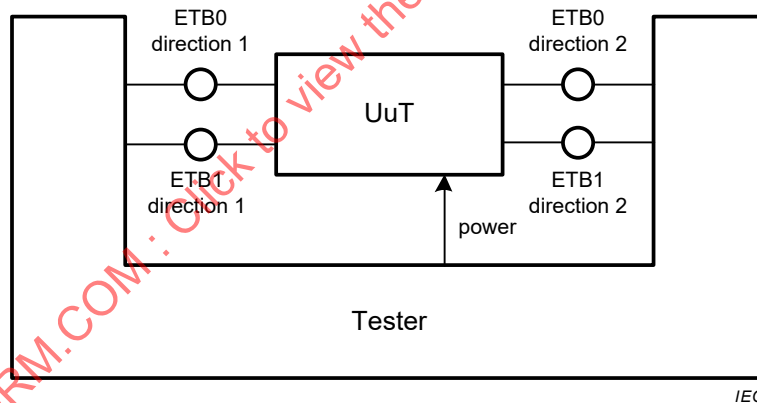
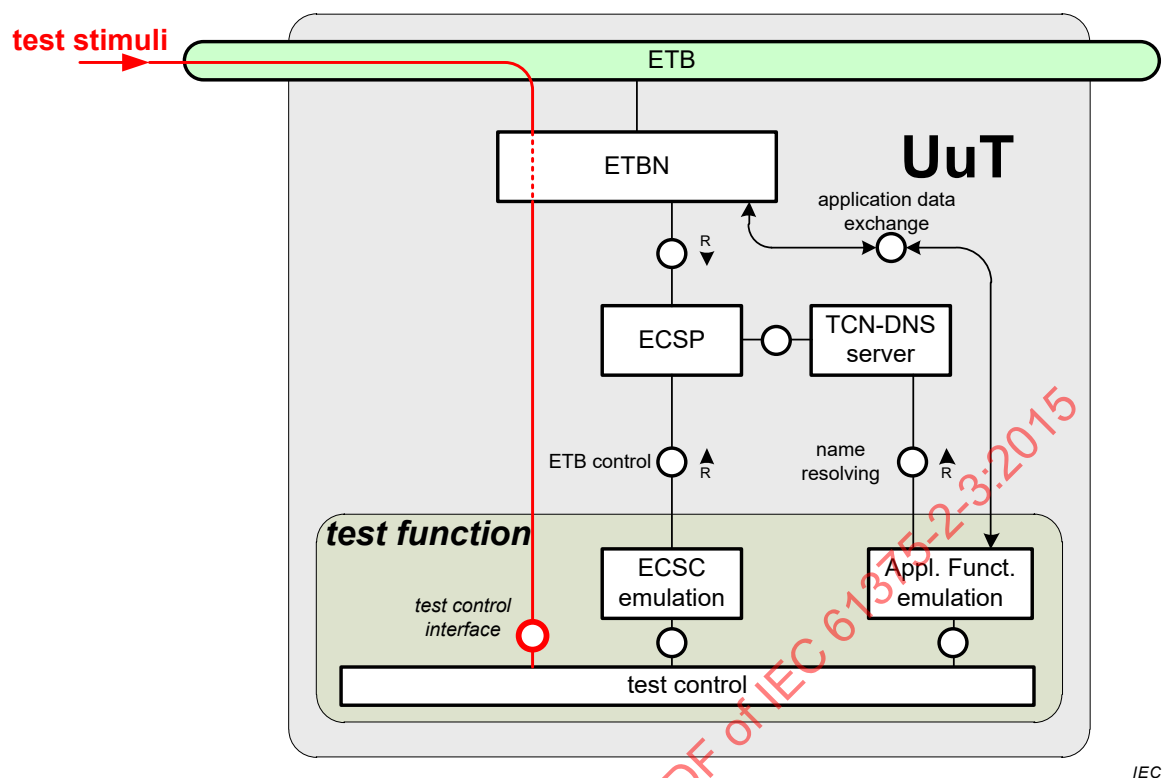


Figure F.4 – Abstract test architecture (2 ETBs)

F.6.4 Set-up for automatic test

An automatic test requires that the UuT is instrumented with a test function which receives and executes test stimuli generated by the tester. The principal architecture of an instrumented UuT is shown in Figure F.5. The UuT contains the ECSP, the TCN-DNS server and the ETBN for connection to the ETB. In addition, it provides a test function which allows to emulate application functions and the ECSC to an extend which is required for the test. The test control implements the interface to the tester. Via this interface, test control commands containing test stimuli are received and, in reply, test status reports are returned.



IEC

Figure F.5 – Unit under test abstract architecture

In closed trains supporting multiple consists, multiple instances of the test functions may be implemented.

F.7 Test of conformity to the common ETB framework

F.7.1 General

The objective of this clause is to give guidance for testing conformity of communication devices (end devices or network devices) to the common ETB framework as defined in Clause 5. Depending on the device type some functions and definitions of the common ETB framework may have to be considered or not. This will be explicitly mentioned in the following subclauses.

F.7.2 Test of CSTINFO telegram

A functional test of the CSTINFO telegram exchange is only affecting ECSP devices and is implicitly done with the test of the ETB control services as defined in Clause F.8.

F.7.3 Test of TTDB

A test of the TTDB is only affecting ECSP devices and is implicitly done with the test of the ETB control services as defined in Clause F.8.

Additional tests may be executed for testing the consist local access to the TTDB, as for instance defined within Annex E.

F.7.4 Test of service addressing and TCN-DNS server

Service addressing uses abstract functional addresses based on TCN-URI schemes as defined in 5.4. However, most protocols used for train wide communication (including TRDP) are solely based on IP addressing. The process of resolving TCN-URI to IP addresses is a consist

internal function, which has no direct impact on the interoperability on ETB level. Therefore it is in the responsibility of the consist manufacturer to perform related tests. Only in cases where the TCN-URI is part of the user data (e.g. used within the payload of TRDP-PD or TRDP-MD telegrams), ETB interoperability will be affected. For those cases a conformity test of TCN-URI resolution shall be executed, which may be restricted to use cases defined on application level, e.g. defined by IEC 61375-2-4 or IEC 62580.

The communication test control interface defined in Clause F.8 provides a possibility to test the TCN-URI address resolution.

The support and test of TCN-URI shall be stated in the PICS.

F.7.5 Test of data exchange

F.7.5.1 TRDP protocol compliance

TRDP (Annex A) defines a generic end-to-end protocol with implementations on all end devices which are involved in process data and message data communication over ETB. TRDP protocol compliance shall be proven by a dedicated TRDP conformance test.

It shall be within the scope of the test laboratory offering a TRDP conformance test to define the necessary test setup and the test cases to be executed. The test set-up may deviate from the test architecture defined in Clause F.6.

NOTE Most often a simple computer is used for tester to which the UuT is directly connected.

F.7.5.2 SDTv2 protocol compliance

SDTv2 (Annex B) defines a generic end-to-end protocol with implementations on all end devices which are involved in safe process data communication over ETB. SDTv2 protocol compliance shall be proven by a dedicated SDTv2 conformance test.

It shall be within the scope of the test laboratory offering a SDTv2 conformance test to define the necessary test setup and the test cases to be executed. The test set-up may deviate from the test architecture defined in Clause F.6.

NOTE Most often a simple computer is used for tester and the UuT is directly connected to it.

F.7.5.3 End-to-end data communication

The objectives of an end-to-end data communication test are:

- Test the proper setup of the IP routers within the ETBN.
- Test the proper configuration of the ECN for correctly passing telegrams to the intended destination.
- Test the proper configuration of the ED (test application) for correctly accepting telegrams intended for the test application.
- Test the proper data marshalling by the destination.
- Test the TCN-URI address resolution of the UuT.

The end-to-end data communication test does not have the objective to test compliance to TRDP and SDTv2 protocols.

Testing end-to-end data communication within a train has to deal with different aspects:

- Transmission protocol, which can be any IP based protocol (e.g. TRDP, streaming protocols, etc.).
- Addressing (IP addresses, or if TCN-URI is used, also the resolving process).

- Train composition changes (inaugurations).
- Transversal communication, e.g. communication between devices connected to the operational network and devices connected to the multimedia network.

The communication profile conformance test shall include tests for generic communication szenarios considering these aspects.

These tests are necessary, but not sufficient. They shall be supplemented by tests on application level, see Clause F.2.

The echo function defined in Clause F.9 provides a possibility to test generic communication patterns.

F.7.6 Test of service discovery

Service discovery as defined in 5.7 relies on the availability of service information within the CSTINFO telegrams. Service information content of CSTINFO is specified in IEC 61375-2-4, therefore a test of service discovery will be in the scope of IEC 61375-2-4.

F.7.7 Test of train info service

The train info service as defined in 5.8 is a consist local function and has no effect on ETB interoperability. Therefore it shall be in the responsibility of the consist manufacturer to execute those tests. The execution of those tests shall be stated in the PICS.

F.8 ETB Control Service conformity test

F.8.1 General

The objective of this clause is to give guidance for testing conformity to the ETB control services as specified in Clause 6. As those services are related to consist and train operation, a stronger focus is set on the test environment in order to generate a common standard for conformance testing. This includes the adoption of the test setup as defined in Clause F.6 and the definition of a common test control interface.

The definition of a test control interface aims to define a standard way of conformance test execution. It intends to avoid a situation that each test laboratory defines its own interface, making it difficult to change the test laboratory once this would be necessary.

In addition, a common set of test cases shall be defined.

NOTE A new standard part will be proposed for the definition of the conformance test cases.

The test control interface is defined by a set of TRDP telegrams which are exchanged between the Tester and the UuT's test function over ETB. Besides this interface, there is no other interface required.

F.8.2 Test control interface for the test of ETB control services

F.8.2.1 Test scope

The test of communication services aims to check the following functions:

- Proper detection of (different) train topologies.
- Handling of leading vehicle requests.
- Handling of confirmation and correction.
- Handling of sleep request.

F.8.2.2 Test control telegram

The Test Control Telegram is cyclically sent point-to-point from the tester to the UuT (ECSC emulation) (Figure F.6). Upon reception of a command from the tester, the UuT (ECSC emulation) shall forward this command to its ECSP for execution.

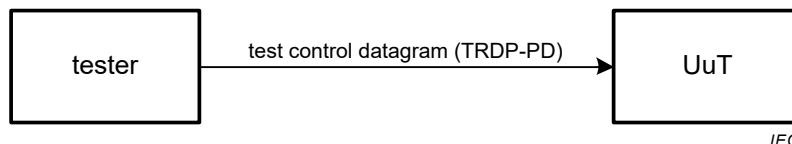


Figure F.6 – Conformance test control telegram

TRDP PD telegram parameters:

Message type:	'Pd'
ComId:	80
Destination:	IP address of UuT device executing the test function (unicast)
Dataset:	CONFTEST_CTRL
Cycle time:	$(1,0 \pm 0,1)$ s
Timeout:	5,0 s

If the receiving UuT detects a timeout, the UuT shall reset waiting for the start of a new test run.

The TRDP telegram is defined as shown in Figure F.7.

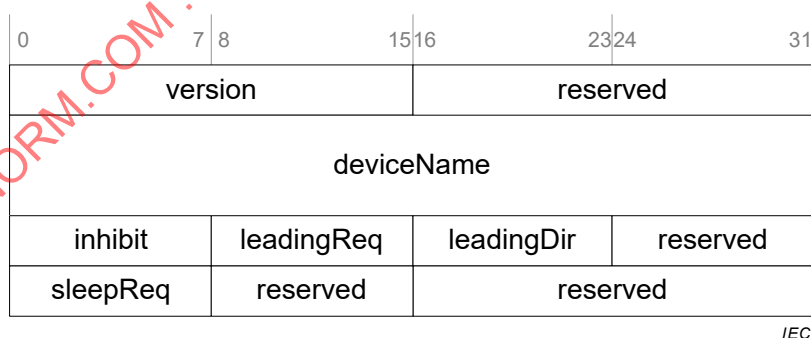


Figure F.7 – Conformance test control telegram data

The payload of the TRDP process data is defined as follows:

```

CONFTEST_CTRL ::= RECORD
{
    version          VERSION          -- telegram version information
                                   main version = 1
                                   subversion = 0
    reserved01       UINT16          -- reserved (= 0)
    
```

deviceName	LABEL	-- function device of tester which sends the telegram
inhibit	UINT8	-- inauguration inhibit 0 = no inhibit request 1 = inhibit request
leadingReq	UINT8	-- leading request 0 = no leading request 1 = leading request
leadingDir	UINT8	-- leading direction 0 = no leading request 1 = leading request direction 1 2 = leading request direction 2
reserved02	UINT8	-- reserved (= 0)
sleepReq	UINT8	-- sleep request 0 = no sleep request 1 = sleep request
reserved03	UINT8	-- reserved (= 0)
reserved04	UINT16	-- reserved (= 0)
}		

F.8.2.3 Test status telegram

The Test Control Data Telegram is cyclically sent point-to-point from the UuT to the tester (Figure F.8).

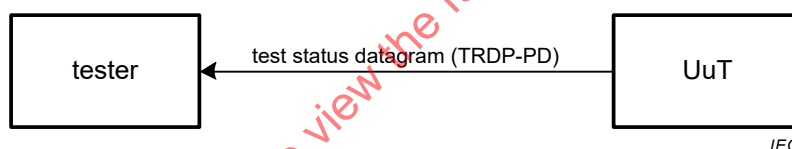


Figure F.8 – Conformance test status telegram

TRDP PD telegram parameters:

Message type:	'Pd'
ComId:	81
Destination:	test function of the tester
Dataset:	CONFTEST_STATUS
Cycle time:	(1,0 ± 0,1) s
Timeout:	5,0 s

The TRDP telegram is defined as shown in Figure F.9.

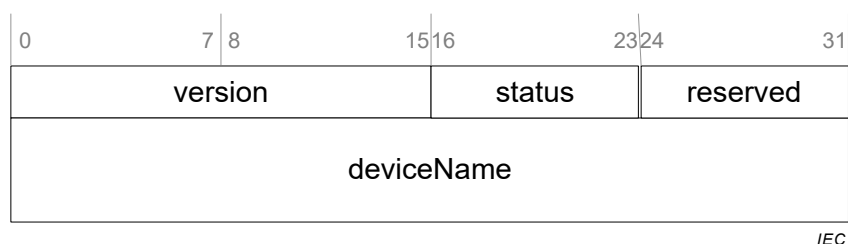


Figure F.9 – Conformance test status telegram data

The payload of the TRDP process data message is defined as follows:

```
CONFTEST_STATUS ::= RECORD
{
  version          VERSION      -- telegram version information
                                main version = 1
                                subversion = 0
  status           UINT8        -- status of the UuT
                                0 = ready
                                > 1 = error detected (UuT specific)
  reserved01       UINT8        -- reserved for future use (= 0)
  deviceName       LABEL        -- function device of UuT which sends
                                the telegram
}
```

F.8.2.4 Confirmation request

F.8.2.4.1 General

This message is sent by the tester to request (un-)confirmation and to inform the UuT about corrections rules. This is done by sending a corrected sequence of vehicles. Vehicle gaps are identified by parameter 'trnVehNo' set to 0, see Figure F.10.

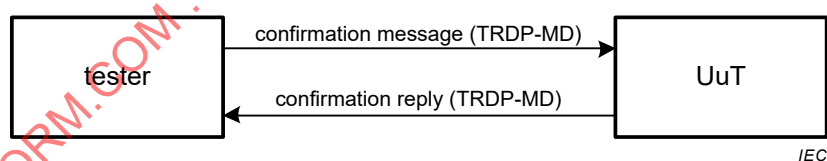


Figure F.10 – (Un-)confirmation request

F.8.2.4.2 Request message

TRDP MD request message parameters:

MsgType:	'Mr'
Protocol:	UDP
ComId:	82
SourceURI (user part):	"ComProfTester"
DestinationURI (user part):	"ComProfTestAppl"

confirmed vehicles in the train,
starting with vehicle at train head,
see chapter 5.3.3.2.10.
Parameters 'isLead', 'leadDir'
and 'ownOpCstNo' to be set to 0.

}

F.8.2.4.3 Reply message

TRDP MD reply message parameters:

ComId: 83

Dataset: CONFTEST_CONF_REPLY

The TRDP message is defined as shown in Figure F.12.

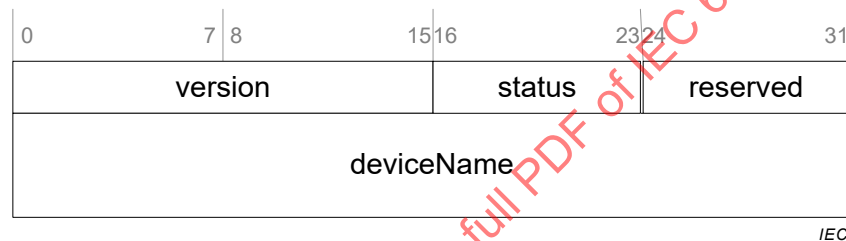


Figure F.12 – Conformance test confirmation/correction reply data

The payload of the TRDP reply message is defined as follows:

```
CONFTEST_CONF_REPLY ::= RECORD
{
    version          VERSION          -- telegram version information
                                   main version = 1
                                   subversion = 0
    status           UINT8            -- status of storing correction info
                                   0 = correctly stored
                                   1 = not stored
    reserved01       UINT8            -- reserved (= 0)
    deviceName       LABEL            -- function device of ECSC which sends
                                   the telegram
}
```

F.8.2.5 Operational train directory request

The operational train directory request is sent as a TRDP-MD request expecting a TRDP-MD reply containing a copy of the operational train directory as computed by the UuT (Figure F.13). This request provides a possibility to check the correct computation of the TTDB.

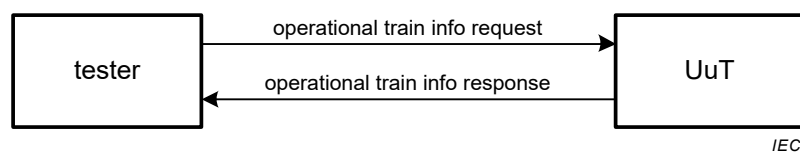


Figure F.13 – Conformance test operational train directory request

TRDP MD request message parameters:

Message type:	'Mr'
Protocol:	UDP
ComId:	84
SourceURI (user part):	"ComProfTestAppl"
DestinationURI (user part):	"ComProfTester"
DestinationURI (host part):	test function of the UuT
SourceURI (host part):	user defined
Dataset:	CONFTEST_OPTRAIN_REQUEST
ReplyTimeOut:	Defined by tester. Default value: 2,5 s

The TRDP message is defined as shown in Figure F.14.



Figure F.14 – Conformance test operational train directory request data

The payload of the TRDP request message is defined as follows:

```

CONFTEST_OPTRAIN_REQUEST ::= RECORD
{
  version          UINT16      -- telegram version information
                                main version = 1
                                subversion = 0
  command          UINT8       -- command to UuT
                                0 = no command (message to be ignored)
                                3 = respond operational train directory
  reserved01       UINT8       -- reserved (= 0)
  deviceName       LABEL       -- function device of tester which sends
                                the telegram
}
  
```


F.9 Echo function

F.9.1 General

The echo function allows a simple communication test between sources and sinks connected to TCN. The echo function is divided into two echo communication patterns:

- Echo test, as specified in F.9.2.
- Reverse-Echo test, as specified in F.9.3.

F.9.2 TRDP echo test

The tester sends an echo request message to the UuT and expects a reply with echoing the user data payload of the request as shown in Figure F.16.

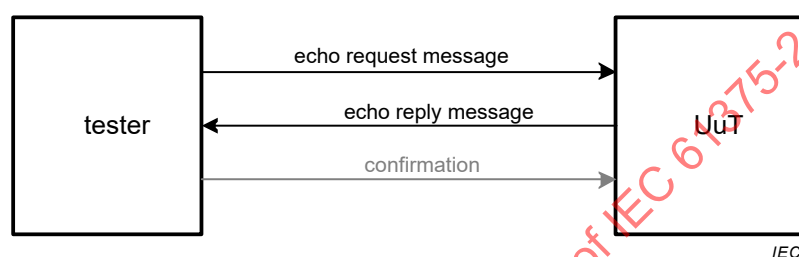


Figure F.16 – Echo test

TRDP MD request message parameters:

MsgType:	'Mr'
Protocol:	UDP or TCP
ComId:	86
NoOfRepliers:	0
SourceURI (user part):	"ComProfTester" or UuT defined (Reverse-Echo test)
DestinationURI (user part):	"ComProfTestAppl" or "ComProfTester" (Reverse-Echo test)
DestinationURI (host part):	-
SourceURI (host part):	user defined
DestinationIpAddress:	IP unicast address of ED (application function emulation), computed from the fctId (see 5.3.3.2.4) or computed from the Reverse-Echo notification message.
	IP multicast address of the multicast group the ED (application function emulation) belongs to, computed from the fctId (see 5.3.3.2.4) or computed from the Reverse-Echo notification message.
Dataset:	Defined by tester or UuT (Reverse-Echo test)

ReplyTimeOut: Defined by tester or UuT (Reverse-Echo test).

Default value: 2,5 s

The tester can run multiple message transfers in parallel.

TRDP MD reply message parameters:

MsgType: 'Mp' or 'Mq'

ComId: 87

SMI: -

UserDataVersion: -

Dataset: Taken from echo request

It is an UuT option to implement an echo reply with or without confirmation.

F.9.3 Reverse-Echo test

The tester sends a reverse echo notification message to the UuT triggering an Echo test initiated by the UuT as shown in Figure F.17.



Figure F.17 – Reverse-Echo test

TRDP MD notification message parameters:

MsgType: 'Mn'

Protocol: UDP

ComId: 88

SourceURI (user part): "ComProfTester"

DestinationURI (user part): "ComProfTestAppl"

DestinationURI (host part): -

SourceURI (host part): user defined

DestinationIpAddress: IP unicast address of ED (application function emulation), computed from the fctId (see 5.3.3.2.4)

Dataset: CONFTEST_MDTRIG_MD

The TRDP message is defined as shown in Figure F.18.

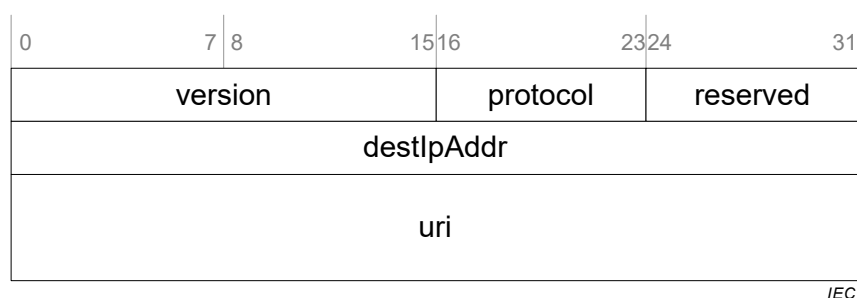


Figure F.18 – Conformance test message data telegram data

The payload of the TRDP notification message is defined as follows:

```

CONFTEST_MDTRIG_MD ::= RECORD
{
  version          VERSION          -- telegram version information
                                     mainVersion = 1
                                     subVersion = 0

  protocol          UINT8           -- protocol for echo test
                                     0 = UDP
                                     1 = TCP

  reserved01        UINT8           -- reserved (= 0)

  destIpAddr        UINT32          -- The destination IP address the UuT
                                     shall send the request to
                                     0 = not used

  uri               CHAR[128]       -- TCN-URI (host part), the UuT shall
                                     send the notification or request to
                                     0 = not used
}

```

It is an option to choose either an IP address (parameter 'destIpAddr') or an URI (Parameter 'uri') as destination of the echo request message.

Upon reception of a reverse echo notification message, the UuT shall initiate an echo test as defined in F.9.2 latest after a time of 2,0 s.

F.10 Statement of conformity

Complete conformity to this part of IEC 61375 is demonstrated when all mandatory conformance tests have been successfully executed.

Table F.1 provides a list of all mandatory and optional conformance tests which have been discussed more deeply in the above subclauses.

Table F.1 – Conformance testing summary

Function/Service	Affected device	Mandatory(M) Optional(O)	Conformance test specification
Common ETB Framework – Service addressing	DNS server (e.g. one per consist network)	M	General test based on Reverse-Echo test. Application specific test in the scope of IEC 61375-2-4
Common ETB Framework – TRDP	End Devices	M	Dedicated specification, e.g. provided by a test laboratory or a user organization
Common ETB Framework – SDTv2	End Devices	O	
Common ETB Framework – Service discovery	ECSP, End Devices	O	In the scope of IEC 61375-2-4
Common ETB Framework – Train info service	ECSP, End Devices	O	Not in the scope of this part of IEC 61375
Common ETB Framework – TTDB	Consist (ECSP)	M	Conformance test interface specified in this part of IEC 61375 Conformance test cases to be generally specified (e.g. in new standard part)
Common ETB Framework – CSTINFO telegram	Consist (ECSP)	M	
ETB control service	Consist (ECSP)	M	
End-to-end data communication	End Devices, ETBN (router), ECN	M	

IECNORM.COM : Click to view the full PDF of IEC 61375-2-3:2015

Annex G (informative)

SNMP Management Information Base (MIB)

G.1 General

This annex defines SNMP MIBs for the following data groups:

- TTDB-MIB for train topology database operational train directory, as defined in 5.3.
- TRDP-MIB for the TRDP protocol statistic data, as defined in Annex D.

NOTE IEC 61375-2-5 defines an MIB for the train topology discovery protocol (TTDP).

The OID tree is defined in accordance to ISO/IEC 9834-1.

G.2 TTDB-MIB

The TTDB-MIB should be implemented on each ECSP. The MIB is defined as follows:

```
--
--          TTDB-MIB DEFINITIONS ::= BEGIN
--
-- This MIB defines objects of the Train Topology Database (TTDB) which
-- is specified in standard IEC 61375-2-3.
--
-- The objects defined in this MIB are located under:
--
--          iso(1).std(0)
--          |
--          stdx61375(61375)
--          |
--          IEC 61375p2(2)
--          |
--          -----
--          |         |         |
--          (trdp(1))  (ttdb(3))  (ttdp(5))
--
--
-- trdp: train realtime data protocol, MIB defined in IEC 61375-2-3
-- ttdb: train topology database, this MIB
-- tt dp: train topology discovery protocol, MIB defined in IEC 61375-2-5
--
--
-- IMPORTS
--      OBJECT-GROUP, MODULE-COMPLIANCE
--      FROM SNMPv2-CONF
--      Unsigned32, OBJECT-TYPE, MODULE-IDENTITY
--      FROM SNMPv2-SMI
--      TEXTUAL-CONVENTION
--      FROM SNMPv2-TC;
--
-- *****
-- Root OID
-- *****
--
-- 1.0.61375.2
-- IEC 61375p2 MODULE-IDENTITY
--   LAST-UPDATED "201405220000Z"
--   ORGANIZATION
--     -- May 22, 2014 00:00 GMT
```

```

        "IEC"
CONTACT-INFO
    "International Electrotechnical Commission
    IEC Central Office
    3, rue de Varembe
    P.O. Box 131
    CH - 1211 GENEVA 20
    Switzerland
    Phone: +41 22 919 02 11
    Fax: +41 22 919 03 00
    email: info@iec.ch"
DESCRIPTION
    "This MIB module defines the Network Management interfaces
    for the Train Topology Database (TTDB) defined by the
    IEC standard 61375-2-3.
    This definition specifies a pure monitoring variant of a
SNMP entity."
REVISION "201405220000Z"
DESCRIPTION "First Release"
::= { stdx61375 2 }

--
-- Textual conventions
--

TtdbOrient ::= TEXTUAL-CONVENTION
    STATUS current
    DESCRIPTION
        "Represents orientation of a vehicle or a Consist"
    SYNTAX INTEGER
        {
            direct(1),
            inverse(2),
            undefined(3)
        }

TtdbValidity ::= TEXTUAL-CONVENTION
    STATUS current
    DESCRIPTION
        "defines validity of the operational train directory"
    SYNTAX INTEGER
        {
            invalid(1),
            valid(2),
            shared(3)
        }

TtdbConfirmation ::= TEXTUAL-CONVENTION
    STATUS current
    DESCRIPTION
        "defines confirmation status of the operational train
directory"
    SYNTAX INTEGER
        {
            unconfirmed(1),
            confirmed(2)
        }

--
-- Node definitions
--

-- 1.0
std OBJECT IDENTIFIER ::= { iso 0 }

-- 1.0.61375
stdx61375 OBJECT IDENTIFIER ::= { std 61375 }

-- *****
-- Train Topology Database (TTDB)

```

```

-- *****
-- 1.0.61375.2.3
ttdb OBJECT IDENTIFIER ::= { IEC 61375p2 3 }

-- *****
-- objects groups of TTDB object identifiers
-- *****
-- 1.0.61375.2.3.1
ttdbObjects OBJECT IDENTIFIER ::= { ttdb 1 }

-- General information
-- 1.0.61375.2.3.1.1
ttdbGenInfo OBJECT IDENTIFIER ::= { ttdbObjects 1 }

-- 1.0.61375.2.3.1.1.1
ttdbEtbId OBJECT-TYPE
    SYNTAX Unsigned32 (1..4)
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "ETB Identifier"
    ::= { ttdbGenInfo 1 }

-- 1.0.61375.2.3.1.1.2
ttdbValidityState OBJECT-TYPE
    SYNTAX TtdbValidity
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Validity of operational train directory"
    ::= { ttdbGenInfo 2 }

-- 1.0.61375.2.3.1.1.3
ttdbConfirmationState OBJECT-TYPE
    SYNTAX TtdbConfirmation
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Confirmation state of operational train directory"
    ::= { ttdbGenInfo 3 }

-- 1.0.61375.2.3.1.1.4
ttdbTrainId OBJECT-TYPE
    SYNTAX OCTET STRING (SIZE (16))
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Train Identifier"
    ::= { ttdbGenInfo 4 }

-- 1.0.61375.2.3.1.1.5
ttdbOpTrnTopoCnt OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "operational train topocounter"
    ::= { ttdbGenInfo 5 }

-- Operational vehicle list

-- 1.0.61375.2.3.1.2
ttdbOpVehList OBJECT IDENTIFIER ::= { ttdbObjects 2 }

-- 1.0.61375.2.3.1.2.1
ttdbOpVehCnt OBJECT-TYPE
    SYNTAX Unsigned32 (1..63)
    MAX-ACCESS read-only
    STATUS current

```

```

DESCRIPTION
    "Number of vehicles in train"
    ::= { ttDbOpVehList 1 }

-- 1.0.61375.2.3.1.2.2
ttDbOpVehTable OBJECT-TYPE
    SYNTAX SEQUENCE OF TtdbOpVehEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "Operational vehicle table"
    ::= { ttDbOpVehList 2 }

-- 1.0.61375.2.3.1.2.2.1
ttDbOpVehEntry OBJECT-TYPE
    SYNTAX TtdbOpVehEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "An antry in the operational vehicle table"
    INDEX { ttDbOpVehIdx }
    ::= { ttDbOpVehTable 1 }

TtdbOpVehEntry ::=
    SEQUENCE {
        ttDbOpVehIdx      Unsigned32,
        ttDbOpVehId       OCTET STRING,
        ttDbOpVehNo       Unsigned32,
        ttDbOpVehIsLead   INTEGER,
        ttDbOpVehLeadDir  INTEGER,
        ttDbOpVehOrient   TtdbOrient }

-- 1.0.61375.2.3.1.2.2.1.1
ttDbOpVehIdx OBJECT-TYPE
    SYNTAX Unsigned32 (1..63)
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "Operational vehicle table index"
    ::= { ttDbOpVehEntry 1 }

-- 1.0.61375.2.3.1.2.2.1.2
ttDbOpVehId OBJECT-TYPE
    SYNTAX OCTET STRING (SIZE(0..16))
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Operational vehicle identifier (UIC vehicle number)"
    ::= { ttDbOpVehEntry 2 }

-- 1.0.61375.2.3.1.2.2.1.3
ttDbOpVehNo OBJECT-TYPE
    SYNTAX Unsigned32 (1..63)
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Operational vehicle sequence number"
    ::= { ttDbOpVehEntry 3 }

-- 1.0.61375.2.3.1.2.2.1.4
ttDbOpVehIsLead OBJECT-TYPE
    SYNTAX INTEGER {notLeading(1),leading(2)}
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Operational vehicle leading property"
    ::= { ttDbOpVehEntry 4 }

-- 1.0.61375.2.3.1.2.2.1.5
ttDbOpVehLeadDir OBJECT-TYPE
    SYNTAX INTEGER {dir1(1),dir2(2)}
    MAX-ACCESS read-only
    STATUS current

```

```

        DESCRIPTION
            "Operational vehicle leading direction"
        ::= { ttldbOpVehEntry 5 }

-- 1.0.61375.2.3.1.2.2.1.6
ttldbOpVehOrient OBJECT-TYPE
    SYNTAX TtdbOrient
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Operational vehicle orientation"
    ::= { ttldbOpVehEntry 6 }

-- *****
-- conformance statements
-- *****
-- 1.0.61375.2.3.2
ttldbConformance OBJECT IDENTIFIER ::= { ttldb 2 }

-- 1.0.61375.2.3.2.2
ttldbBasicGroup OBJECT-GROUP
    OBJECTS { ttldbEtbId,
               ttldbValidityState,
               ttldbConfirmationState,
               ttldbTrainId,
               ttldbOpVehCnt,
               ttldbOpTrnTopoCnt}
    STATUS current
    DESCRIPTION
        "basic group of TTDB parameters"
    ::= { ttldbConformance 2 }

-- 1.0.61375.2.3.2.3
ttldbOpVehListGroup OBJECT-GROUP
    OBJECTS { ttldbOpVehId,
               ttldbOpVehNo,
               ttldbOpVehIsLead,
               ttldbOpVehLeadDir,
               ttldbOpVehOrient}
    STATUS current
    DESCRIPTION
        "operational vehicle list parameters for TTDB monitoring"
    ::= { ttldbConformance 3 }

-- 1.0.61375.2.3.2.4
ttldbBasicCompliance MODULE-COMPLIANCE
    STATUS current
    DESCRIPTION
        "basic implementation requirements for TTDB monitoring"
    MODULE -- this module
    MANDATORY-GROUPS {ttldbBasicGroup,ttldbOpVehListGroup}
    ::= { ttldbConformance 4 }

```

END

G.3 TRDP-MIB

The TRDP-MIB should be implemented on each end device which is running TRDP for train wide communication. The MIB is defined as follows:

```
--
--      TTDB-MIB DEFINITIONS ::= BEGIN
--
-- This MIB defines objects of the Train Realtime Data Protocol (TRDP) which
-- is specified in standard IEC 61375-2-3.
--
-- The objects defined in this MIB are located under:
--
--       iso(1).std(0)
--         |
--         +-- stdx61375(61375)
--           |
--           +-- IEC 61375p2(2)
--             |
--             +-----+-----+-----+
--             |         |         |
--             +-- trdp(1)   (ttdb(3))   (ttdp(5))
--
-- trdp: train realtime data protocol, this MIB
-- ttdb: train topology database, MIB defined in IEC 61375-2-3
-- ttdp: train topology discovery protocol, MIB defined in IEC 61375-2-5
--
--
IMPORTS
    OBJECT-GROUP, MODULE-COMPLIANCE
        FROM SNMPv2-CONF
    IpAddress, Unsigned32, OBJECT-TYPE,          MODULE-IDENTITY
        FROM SNMPv2-SMI
    ;

-- *****
-- Root OID
-- *****

-- 1.0.61375.2
IEC 61375p2 MODULE-IDENTITY
    LAST-UPDATED "201405220000Z"          -- May 22, 2014 00:00 GMT
    ORGANIZATION
        "IEC"
    CONTACT-INFO
        "International Electrotechnical Commission
        IEC Central Office
        3, rue de Varembe
        P.O. Box 131
        CH - 1211 GENEVA 20
        Switzerland
        Phone: +41 22 919 02 11
        Fax: +41 22 919 03 00
        email: info@iec.ch"
    DESCRIPTION
        "This MIB module defines the Network Management interfaces
        for the Train Topology Database (TTDB) defined by the
        IEC standard 61375-2-3.
        This definition specifies a pure monitoring variant of a
        SNMP entity."
    REVISION "201405220000Z"
    DESCRIPTION "First Release"
    ::= { stdx61375 2 }
```

```

--
-- Node definitions
--

    -- 1.0
    std OBJECT IDENTIFIER ::= { iso 0 }

    -- 1.0.61375
    stdx61375 OBJECT IDENTIFIER ::= { std 61375 }

-- *****
-- Train Realtime Data Protocol (TRDP)
-- *****
    -- 1.0.61375.2.1
    trdp OBJECT IDENTIFIER ::= { IEC 61375p2 1 }

-- *****
-- objects groups of TTDB object identifiers
-- *****
    -- 1.0.61375.2.1.1
    trdpObjects OBJECT IDENTIFIER ::= { trdp 1 }

-- General information
    -- 1.0.61375.2.1.1.1
    trdpGenInfo OBJECT IDENTIFIER ::= { trdpObjects 1 }

    -- 1.0.61375.2.1.1.1.1
    trdpGenVers OBJECT-TYPE
        SYNTAX Unsigned32
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
            "TRDP version"
        ::= { trdpGenInfo 1 }

    -- 1.0.61375.2.1.1.1.2
    trdpGenUpTime OBJECT-TYPE
        SYNTAX Unsigned32
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
            "time in seconds since last initialization"
        ::= { trdpGenInfo 2 }

    -- 1.0.61375.2.1.1.1.3
    trdpGenStatTime OBJECT-TYPE
        SYNTAX Unsigned32
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
            "time in seconds since last reset of statistics"
        ::= { trdpGenInfo 3 }

    -- 1.0.61375.2.1.1.1.4
    trdpGenHostName OBJECT-TYPE
        SYNTAX OCTET STRING (SIZE(0..16))
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
            "Own host name"
        ::= { trdpGenInfo 4 }

    -- 1.0.61375.2.1.1.1.5
    trdpGenLeadName OBJECT-TYPE
        SYNTAX OCTET STRING (SIZE(0..16))
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
            "Leader host name"

```

```

        ::= { trdpGenInfo 5 }

-- 1.0.61375.2.1.1.1.6
trdpGenOwnIp OBJECT-TYPE
    SYNTAX IpAddress
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Own IP address"
    ::= { trdpGenInfo 6 }

-- 1.0.61375.2.1.1.1.7
trdpGenLeadIp OBJECT-TYPE
    SYNTAX IpAddress
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Leader IP address"
    ::= { trdpGenInfo 7 }

-- 1.0.61375.2.1.1.1.8
trdpGenProcPrio OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "priority of TRDP process"
    ::= { trdpGenInfo 8 }

-- 1.0.61375.2.1.1.1.9
trdpGenProcCycle OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Cycle time of TRDP process in microseconds"
    ::= { trdpGenInfo 9 }

-- 1.0.61375.2.1.1.1.10
trdpGenNumJoin OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of IGMP joins"
    ::= { trdpGenInfo 10 }

-- 1.0.61375.2.1.1.1.11
trdpGenNumRed OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of redundancy groups"
    ::= { trdpGenInfo 11 }

-- Memory statistics
-- 1.0.61375.2.1.1.2
trdpMemStat OBJECT IDENTIFIER ::= { trdpObjects 2 }

-- 1.0.61375.2.1.1.2.1
trdpMemTotal OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Total memory size"
    ::= { trdpMemStat 1 }

-- 1.0.61375.2.1.1.2.2
trdpMemFree OBJECT-TYPE
    SYNTAX Unsigned32

```



```

        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
            "Free memory"
        ::= { trdpMemStat 2 }

-- 1.0.61375.2.1.1.2.3
trdpMemMinFree OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Minimal free memory in statistics interval"
    ::= { trdpMemStat 3 }

-- 1.0.61375.2.1.1.2.4
trdpMemAllocBlocks OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of allocated blocks"
    ::= { trdpMemStat 4 }

-- 1.0.61375.2.1.1.2.5
trdpMemAllocErr OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of allocation errors"
    ::= { trdpMemStat 5 }

-- 1.0.61375.2.1.1.2.6
trdpMemFreeErr OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of freeing errors"
    ::= { trdpMemStat 6 }

-- Process data statistics
-- 1.0.61375.2.1.1.3
trdpPdStat OBJECT IDENTIFIER ::= { trdpObjects 3 }

-- 1.0.61375.2.1.1.3.1
trdpPdDefQos OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Default QoS for PD"
    ::= { trdpPdStat 1 }

-- 1.0.61375.2.1.1.3.2
trdpPdDefTtl OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Default TTL for PD"
    ::= { trdpPdStat 2 }

-- 1.0.61375.2.1.1.3.3
trdpPdDefTo OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Default timeout for PD"

```

```

        ::= { trdpPdStat 3 }

-- 1.0.61375.2.1.1.3.4
trdpPdNumSubs OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of subscribed ComIds"
    ::= { trdpPdStat 4 }

-- 1.0.61375.2.1.1.3.5
trdpPdNumPub OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of published ComIds"
    ::= { trdpPdStat 5 }

-- 1.0.61375.2.1.1.3.6
trdpPdNumRcv OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "number of received PD packets"
    ::= { trdpPdStat 6 }

-- 1.0.61375.2.1.1.3.7
trdpPdNumCrcErr OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of received PD packets with CRC error"
    ::= { trdpPdStat 7 }

-- 1.0.61375.2.1.1.3.8
trdpPdNumProtErr OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of received PD packets with protocol error"
    ::= { trdpPdStat 8 }

-- 1.0.61375.2.1.1.3.9
trdpPdNumTopoErr OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of received PD packets with wrong
        topo counter value"
    ::= { trdpPdStat 9 }

-- 1.0.61375.2.1.1.3.10
trdpPdNumNoSubs OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of received PD packets without subscription"
    ::= { trdpPdStat 10 }

-- 1.0.61375.2.1.1.3.11
trdpPdNumNoPub OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of received PD packets without publisher"
    ::= { trdpPdStat 11 }

```

```
-- 1.0.61375.2.1.1.3.12
trdpPdNumTo OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of PD timeouts"
    ::= { trdpPdStat 12 }

-- 1.0.61375.2.1.1.3.13
trdpPdNumSend OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of sent PD packets"
    ::= { trdpPdStat 13 }

-- Message data statistics (UDP)
-- 1.0.61375.2.1.1.4
trdpMduStat OBJECT IDENTIFIER ::= { trdpObjects 4 }

-- 1.0.61375.2.1.1.4.1
trdpMduDefQos OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Default QoS for MD"
    ::= { trdpMduStat 1 }

-- 1.0.61375.2.1.1.4.2
trdpMduDefTtl OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "default TTL for MD"
    ::= { trdpMduStat 2 }

-- 1.0.61375.2.1.1.4.3
trdpMduDefReplyTo OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Default reply timeout for MD"
    ::= { trdpMduStat 3 }

-- 1.0.61375.2.1.1.4.4
trdpMduDefConfTo OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Default confirm timeout for MD"
    ::= { trdpMduStat 4 }

-- 1.0.61375.2.1.1.4.5
trdpMduNumList OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of Listeners"
    ::= { trdpMduStat 5 }

-- 1.0.61375.2.1.1.4.6
trdpMduNumRcv OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
```

```

        "Number of received MD packets"
        ::= { trdpMduStat 6 }

-- 1.0.61375.2.1.1.4.7
trdpMduNumCrcErr OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of received MD packets with CRC error"
        ::= { trdpMduStat 7 }

-- 1.0.61375.2.1.1.4.8
trdpMduNumProtErr OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of received MD packets with protocol
        error"
        ::= { trdpMduStat 8 }

-- 1.0.61375.2.1.1.4.9
trdpMduNumTopoErr OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of received MD packets with wrong
        topo counter value"
        ::= { trdpMduStat 9 }

-- 1.0.61375.2.1.1.4.10
trdpMduNumNoList OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of received MD packets without
        Listener"
        ::= { trdpMduStat 10 }

-- 1.0.61375.2.1.1.4.11
trdpMduNumReplyTo OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of MD reply timeouts"
        ::= { trdpMduStat 11 }

-- 1.0.61375.2.1.1.4.12
trdpMduNumConfTo OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of MD confirm timeouts"
        ::= { trdpMduStat 12 }

-- 1.0.61375.2.1.1.4.13
trdpMduNumSend OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "number of sent MD packets"
        ::= { trdpMduStat 13 }

-- Message data statistics (TCP)
-- 1.0.61375.2.1.1.5
trdpMdtStat OBJECT IDENTIFIER ::= { trdpObjects 5 }

-- 1.0.61375.2.1.1.5.1

```

```
trdpMdtDefQos OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Default QoS for MD"
    ::= { trdpMdtStat 1 }

-- 1.0.61375.2.1.1.5.2
trdpMdtDefTtl OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "default TTL for MD"
    ::= { trdpMdtStat 2 }

-- 1.0.61375.2.1.1.5.3
trdpMdtDefReplyTo OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Default reply timeout for MD"
    ::= { trdpMdtStat 3 }

-- 1.0.61375.2.1.1.5.4
trdpMdtDefConfTo OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Default confirm timeout for MD"
    ::= { trdpMdtStat 4 }

-- 1.0.61375.2.1.1.5.5
trdpMdtNumList OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of Listeners"
    ::= { trdpMdtStat 5 }

-- 1.0.61375.2.1.1.5.6
trdpMdtNumRcv OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of received MD packets"
    ::= { trdpMdtStat 6 }

-- 1.0.61375.2.1.1.5.7
trdpMdtNumCrcErr OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of received MD packets with CRC error"
    ::= { trdpMdtStat 7 }

-- 1.0.61375.2.1.1.5.8
trdpMdtNumProtErr OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of received MD packets with protocol
        error"
    ::= { trdpMdtStat 8 }

-- 1.0.61375.2.1.1.5.9
trdpMdtNumTopoErr OBJECT-TYPE
```

```

SYNTAX Unsigned32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
    "Number of received MD packets with wrong
    topo counter value"
 ::= { trdpMdtStat 9 }

-- 1.0.61375.2.1.1.5.10
trdpMdtNumNoList OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of received MD packets without
        Listener"
    ::= { trdpMdtStat 10 }

-- 1.0.61375.2.1.1.5.11
trdpMdtNumReplyTo OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of MD reply timeouts"
    ::= { trdpMdtStat 11 }

-- 1.0.61375.2.1.1.5.12
trdpMdtNumConfTo OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Number of MD confirm timeouts"
    ::= { trdpMdtStat 12 }

-- 1.0.61375.2.1.1.5.13
trdpMdtNumSend OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "number of sent MD packets"
    ::= { trdpMdtStat 13 }

-- Redundancy statistics
-- 1.0.61375.2.1.1.6
trdpRedStat OBJECT IDENTIFIER ::= { trdpObjects 6 }

-- 1.0.61375.2.1.1.6.1
trdpRedId OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Redundancy Id"
    ::= { trdpRedStat 1 }

-- 1.0.61375.2.1.1.6.2
trdpRedState OBJECT-TYPE
    SYNTAX Unsigned32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "State: leader/follower"
    ::= { trdpRedStat 2 }

-- *****
-- conformance statements
-- *****
-- 1.0.61375.2.1.2
trdpConformance OBJECT IDENTIFIER ::= { trdp 2 }

```

```
-- 1.0.61375.2.1.2.2
trdpGenGroup OBJECT-GROUP
  OBJECTS { trdpGenVers,
            trdpGenUpTime,
            trdpGenStatTime,
            trdpGenHostName,
            trdpGenLeadName,
            trdpGenOwnIp,
            trdpGenLeadIp,
            trdpGenProcPrio,
            trdpGenProcCycle,
            trdpGenNumJoin,
            trdpGenNumRed}

  STATUS current
  DESCRIPTION
    "General statistics and configuration data"
  ::= { trdpConformance 2 }

-- 1.0.61375.2.1.2.3
trdpMemGroup OBJECT-GROUP
  OBJECTS { trdpMemTotal,
            trdpMemFree,
            trdpMemMinFree,
            trdpMemAllocBlocks,
            trdpMemAllocErr,
            trdpMemFreeErr,
            trdpMemFreeErr}

  STATUS current
  DESCRIPTION
    "Memory statistics and configuration data"
  ::= { trdpConformance 3 }

-- 1.0.61375.2.1.2.4
trdpPdGroup OBJECT-GROUP
  OBJECTS { trdpPdDefQos,
            trdpPdDefTtl,
            trdpPdDefTo,
            trdpPdNumSubs,
            trdpPdNumPub,
            trdpPdNumRcv,
            trdpPdNumCrcErr,
            trdpPdNumProtErr,
            trdpPdNumTopoErr,
            trdpPdNumNoSubs,
            trdpPdNumNoPub,
            trdpPdNumTo,
            trdpPdNumSend}

  STATUS current
  DESCRIPTION
    "PD statistis and configuration data"
  ::= { trdpConformance 4 }

-- 1.0.61375.2.1.2.5
trdpMduGroup OBJECT-GROUP
  OBJECTS { trdpMduDefQos,
            trdpMduDefTtl,
            trdpMduDefReplyTo,
            trdpMduDefConfTo,
            trdpMduNumList,
            trdpMduNumRcv,
            trdpMduNumCrcErr,
            trdpMduNumProtErr,
            trdpMduNumTopoErr,
            trdpMduNumReplyTo,
            trdpMduNumNoList,
            trdpMduNumConfTo,
            trdpMduNumSend}

  STATUS current
  DESCRIPTION
    "MD statistis and configuration data"
  ::= { trdpConformance 5 }
```

```
-- 1.0.61375.2.1.2.6
trdpMdtGroup OBJECT-GROUP
  OBJECTS { trdpMdtDefQos,
            trdpMdtDefTtl,
            trdpMdtDefReplyTo,
            trdpMdtDefConfTo,
            trdpMdtNumList,
            trdpMdtNumRcv,
            trdpMdtNumCrcErr,
            trdpMdtNumProtErr,
            trdpMdtNumTopoErr,
            trdpMdtNumReplyTo,
            trdpMdtNumNoList,
            trdpMdtNumConfTo,
            trdpMdtNumSend}
  STATUS current
  DESCRIPTION
    "MD statistis and configuration data"
  ::= { trdpConformance 6 }

-- 1.0.61375.2.1.2.7
trdpRedGroup OBJECT-GROUP
  OBJECTS { trdpRedId,
            trdpRedState }
  STATUS current
  DESCRIPTION
    "General statistis and configuration data"
  ::= { trdpConformance 7 }

-- 1.0.61375.2.1.2.8
trdpBasicCompliance MODULE-COMPLIANCE
  STATUS current
  DESCRIPTION
    "basic implementation requirements for TRDP monitoring"
  MODULE -- this module
  MANDATORY-GROUPS {trdpGenGroup,
                    trdpMemGroup,
                    trdpPdGroup,
                    trdpMduGroup,
                    trdpMdtGroup,
                    trdpRedGroup }
  ::= { trdpConformance 8 }
```

END

Bibliography

IEC 60571, *Railway applications – Electronic equipment used on rolling stock*

IEC 61131-3:2013, *Programmable controllers – Part 3: Programming languages*

IEC 61375-3-4, *Electronic railway equipment – Train Communication Network (TCN) – Part 3-4: Ethernet Consist Network (ECN)*

IEC 61508-1:2010, *Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 1: General requirements*

IEC 61784-3-3:2010, *Industrial communication networks – Profiles – Part 3-3: Functional safety fieldbuses – Additional specifications for CPF 3*

IEC 62580 (all parts), *Electronic railway equipment – Onboard multimedia and telematic subsystems for railways*

ISO/IEC 7498-1, *Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model*

ISO/IEC 9834-1:2012, *Information technology – Procedures for the operation of object identifier registration authorities: General procedures and top arcs of the international object identifier tree*

EN 15380-4, *Railway applications – Classification system for railway vehicles – Part 4: Function groups*

UIC Leaflet 550, *Power supply installations for passenger stock, 11th edition, April 2005*

UIC Leaflet 550-1, *Electrical switch cabinets on passenger stock, 1st edition of 1.1.90*

UIC Leaflet 556, *Information transmission in the train (train bus), 5th edition 2010*

IETF Requests For Comments:

RFC 768, *User Datagram Protocol*

RFC 791, *Internet Protocol*

RFC 793, *Transmission Control Protocol*

RFC 956, *Algorithms for synchronizing network clocks*

RFC 1034, *Domain names – concepts and facilities*

RFC 1035, *Domain names – implementation and specification*

RFC 2326, *Real Time Streaming Protocol (RTSP)*

RFC 2616, *Hypertext Transfer Protocol – HTTP/1.1.*

RFC 2663, *IP Network Address Translator (NAT) Terminology and Considerations*

RFC 2766, *Network Address Translation – Protocol Translation (NAT-PT)*

RFC 3022, *Traditional IP Network Address Translator (Traditional NAT)*

RFC 3261, *SIP: Session Initiation Protocol*

RFC 3550, *RTP: A Transport Protocol for Real-Time Applications*

RFC 3605, *Real Time Control Protocol (RTCP) attribute in Session Description Protocol (SDP)*

RFC 3711, *The Secure Real-time Transport Protocol (SRTP)*

RFC 3986, *Uniform Resource Identifier (URI): Generic Syntax*

RFC 4122, *A Universally Unique Identifier (UUID) URN Namespace*

RFC 5246, *The Transport Layer Security (TLS) Protocol Version 1.2.*

SOMMAIRE

AVANT-PROPOS	292
INTRODUCTION.....	294
1 Domaine d'application	295
2 Références normatives	296
3 Termes, définitions, abréviations, acronymes et conventions	297
3.1 Termes et définitions	297
3.2 Abréviations et acronymes	305
3.3 Conventions	308
3.3.1 Base des valeurs numériques	308
3.3.2 Chaînes de caractères et citations	308
3.3.3 Conventions d'appellation	308
3.3.4 Conventions pour les diagrammes	309
3.3.5 Annotation des structures de données	309
4 Architecture	310
4.1 Généralités	310
4.2 Architecture physique du train (décomposition du système)	310
4.2.1 Généralités	310
4.2.2 Architectures de réseau de train	311
4.2.3 Ensembles de rames insécables	315
4.2.4 Directions	317
4.2.5 Propriétés de base des rames et des véhicules	318
4.3 Architecture logique de train (décomposition fonctionnelle)	319
4.3.1 Généralités	319
4.3.2 Classification des services	319
4.3.3 Présentation des services opérationnels	321
4.3.4 Fournisseur de service	321
5 Cadre ETB commun	321
5.1 Généralités	321
5.1.1 Vue d'ensemble	321
5.1.2 Interopérabilité	322
5.2 Télégramme CSTINFO	322
5.2.1 Généralités	322
5.2.2 Prise en charge des ensembles de rames insécables (option)	322
5.2.3 Protocole	322
5.2.4 Classes CSTINFO	323
5.2.5 Message de notification CSTINFO	323
5.2.6 Demande CSTINFO	324
5.3 Base de données topologiques de train	326
5.3.1 Généralités	326
5.3.2 Calcul de la TTDB	329
5.3.3 Structure des données	335
5.3.4 Base de données topologiques de train pour plusieurs ETB (option)	344
5.4 Adressage des services	346
5.4.1 Généralités	346
5.4.2 Système de noms de domaine TCN (DNS TCN)	347
5.4.3 Noms de domaine TCN	348

5.4.4	Schéma de l'URI TCN	348
5.4.5	Mapping d'URI TCN en adresses IP	355
5.4.6	Prise en charge d'autres schémas d'URI	359
5.5	Serveur TCN-DNS	359
5.5.1	Généralités	359
5.5.2	Architecture	359
5.5.3	Résolution d'adresse fonctionnelle	359
5.5.4	Protocole	362
5.5.5	ETB multiples	363
5.6	Echange de données	363
5.6.1	Généralités	363
5.6.2	Réseau opérationnel de communication	363
5.6.3	Réseau OMTS de communication	364
5.6.4	Qualité de service (QoS)	364
5.7	Découverte de service	365
5.8	Service d'informations de train	365
6	Services du profil de communication – Service de contrôle ETB	365
6.1	Généralités	365
6.2	Modèle de communication	365
6.3	Supervision ECSP	366
6.4	Interconnexion d'ECSP	366
6.4.1	Généralités	366
6.4.2	Sélection d'échange de télégrammes ETBCTRL	367
6.4.3	Transmission de télégramme ETBCTRL	367
6.4.4	Structure du télégramme ETBCTRL	367
6.4.5	Processus de calcul du répertoire opérationnel de train	370
6.5	Fonction "véhicule de tête"	374
6.5.1	Généralités	374
6.5.2	Primitives de fonction	374
6.5.3	Protocole d'ECSP à ECSP	375
6.6	Fonction confirmation/correction	380
6.6.1	Généralités	380
6.6.2	Primitives de fonction	381
6.6.3	Protocole d'ECSP à ECSP	382
6.6.4	Diagramme d'états	386
6.6.5	Défaillance de l'ECSC	388
6.7	Calcul du répertoire opérationnel de train	389
6.7.1	Généralités	389
6.7.2	Action setCorrInfo	389
6.7.3	Action computeOpTrnDir	394
6.8	Fonction mode sommeil (option)	397
6.8.1	Généralités	397
6.8.2	Cas d'utilisation du mode sommeil (informel)	397
6.8.3	Exclusivité	399
6.8.4	Primitives de fonction	399
6.8.5	Protocole d'ECSP à ECSP	400
Annexe A (normative)	Protocole temps réel de données de train (TRDP)	405
A.1	Généralités	405
A.2	Couches inférieures	406

A.2.1	Couche liaison de données	406
A.2.2	Couche réseau	406
A.2.3	Couche transport	406
A.3	Calcul du FCS TRDP	407
A.4	Interaction entre utilisateur TRDP et couche TRDP	409
A.5	Identificateur de communication (ComId)	409
A.6	Données de processus	411
A.6.1	Modèle de communication	411
A.6.2	Rôles	411
A.6.3	Modèle de communication	412
A.6.4	Adressage	417
A.6.5	PD-PDU	418
A.6.6	Interaction entre application et couche de protocole TRDP	420
A.6.7	Contrôle du compteur de topographie	428
A.6.8	Diagramme d'états	429
A.7	Données de messagerie	434
A.7.1	Modèle de communication	434
A.7.2	Rôles	435
A.7.3	Modèle de communication	436
A.7.4	Adressage	436
A.7.5	MD-PDU	436
A.7.6	Interaction entre l'application et la couche TRDP	440
A.7.7	Contrôle du compteur de topographie	446
A.7.8	Diagramme d'états de protocole de MD	447
A.7.9	Gestion des connexions TCP	457
A.8	Serveur de données de messagerie d'écho (option)	458
Annexe B (normative)	Transmission de données sécurisées (SDTv2)	459
B.1	Généralités	459
B.2	Vue d'ensemble de la SDTv2 (informelle)	459
B.3	Exigences fonctionnelles de sécurité	460
B.4	Mesures de sécurité	461
B.5	Modes opérationnels du canal SDTv2	461
B.6	Présentation des données	462
B.7	SC-32	463
B.8	SID	465
B.9	Paquets de données stratégiques	466
B.10	Exclusivité	467
B.11	Paramètres temporels de configuration	467
B.12	Source de données sécurisées (SDSRC)	468
B.12.1	Généralités	468
B.12.2	Préparation des données sécurisées (application)	468
B.12.3	Envoi de données sécurisées	468
B.13	Destinataire de données sécurisées (SDSINK)	470
B.13.1	Généralités	470
B.13.2	Définitions	470
B.13.3	Etats de l'élément SDSINK	472
B.13.4	Echantillonnage du VDP	473
B.13.5	Contrôle d'intégrité des VDP	474
B.13.6	Supervision du temps de transmission	475

B.13.7	Contrôle de l'intervalle de garde	476
B.13.8	Surveillance de la latence	477
B.13.9	Surveillance du canal	480
B.13.10	Interface de l'application SDTv2	481
B.13.11	Modification de la composition du train fonctionnel	481
B.14	Diagnostic et statistiques	482
B.15	Transmission de données sécurisées sur le MVB (informative)	482
B.15.1	Généralités	482
B.15.2	VDP MVB	482
B.15.3	Ecarts du protocole SDTv2 spécifiques au MVB	484
B.16	SDTv2 prenant en charge les données de messagerie TRDP	484
Annexe C (informative)	Configuration du protocole temps réel de données de train (TRDP)	485
C.1	Généralités	485
C.2	Paramètres de l'appareil	487
C.3	Paramètres de configuration de l'appareil	487
C.4	Liste relative à l'interface de bus	488
C.4.1	Généralités	488
C.4.2	Configuration de l'interface de bus	488
C.5	Paramètres de l'appareil mappé	500
C.5.1	Généralités	500
C.5.2	Paramètres de l'interface de bus mappée	501
C.6	Paramètres de communication (ComPar)	504
C.6.1	Généralités	504
C.6.2	Paramètres de communication par défaut	504
C.7	Paramètres de l'objet DataSet	505
C.7.1	Généralités	505
C.7.2	Élément de l'objet DataSet	507
C.7.3	Exemples d'objets DataSet	508
Annexe D (informative)	Accès aux statistiques de l'appareil terminal (ED)	513
D.1	Généralités	513
D.2	Structures	513
D.2.1	Généralités	513
D.2.2	tlc_getSubsStatistics	515
D.2.3	tlc_getPubStatistics	515
D.2.4	tlc_getUdpListStatistics, tlc_getTcpListStatistics	515
D.2.5	tlc_getRedStatistics	516
D.3	Interface des appareils terminaux pour l'accès aux données statistiques	516
D.3.1	Généralités	516
D.3.2	Interface du TRDP	516
Annexe E (informative)	Interface de service	518
E.1	Généralités	518
E.2	Fournisseur de service	520
E.2.1	Proxys	520
E.2.2	Performances	520
E.3	Interface ECSP	520
E.3.1	Généralités	520
E.3.2	Télégramme de contrôle ECSP	521
E.3.3	Télégramme des états de l'ECSP	522

E.3.4	Demande de confirmation/correction de l'ECSP	524
E.4	Interface du gestionnaire TTDB.....	527
E.4.1	Généralités	527
E.4.2	Informations sur l'état de la TTDB.....	527
E.4.3	notification TTDB	528
E.4.4	Informations sur la TTDB – répertoire de train	529
E.4.5	Informations sur la TTDB – informations sur la rame statique	530
E.4.6	Informations sur la TTDB – informations sur le répertoire du réseau de trains	531
E.4.7	Informations sur le répertoire opérationnel de train	532
E.4.8	Lecture de la TTDB.....	533
E.5	Interface serveur DNS.....	534
E.5.1	Interface DNS normalisée	534
E.5.2	Interface DNS TCN	534
E.6	Interface de contrôle ETBN	538
E.6.1	Généralités	538
E.6.2	Données sur le contrôle et l'état de l'ETBN	539
E.6.3	Répertoire de réseau de trains de l'ETBN	542
Annexe F (normative)	Directives d'essai de conformité du profil de communication.....	544
F.1	Généralités	544
F.2	Domaine d'application de l'essai de conformité	545
F.3	Vue d'ensemble de l'essai de conformité	545
F.4	Laboratoire d'essai	545
F.4.1	Généralités	545
F.4.2	Tâches	546
F.5	Directives pour la rédaction des spécifications relatives à l'essai de conformité.....	546
F.5.1	Vue d'ensemble des principaux composants	546
F.5.2	Déclaration de conformité de mise en œuvre du protocole (PICS)	547
F.5.3	Architecture d'essai abstraite.....	547
F.5.4	Informations complémentaires de mise en œuvre de protocole nécessaires aux essais de conformité (PIXIT).....	547
F.5.5	Structure de la suite d'essais	547
F.6	Architecture d'essai abstraite (facultative).....	548
F.6.1	Généralités	548
F.6.2	Architecture d'essai avec un ETB	548
F.6.3	Architecture d'essai avec plusieurs ETB	549
F.6.4	Configuration pour essais automatiques	549
F.7	Essai de conformité du cadre ETB commun	550
F.7.1	Généralités	550
F.7.2	Essai du télégramme CSTINFO	551
F.7.3	Essai de la TTDB.....	551
F.7.4	Essai d'adressage de service et du serveur DNS TCN	551
F.7.5	Essai d'échange des données.....	551
F.7.6	Essai de découverte de service	552
F.7.7	Essai de service d'information sur le train.....	552
F.8	Essai de conformité du service de contrôle ETB	553
F.8.1	Généralités	553
F.8.2	Interface de contrôle d'essai pour l'essai des services de contrôle ETB	553

F.9	Fonction d'écho	561
F.9.1	Généralités	561
F.9.2	Essai d'écho du protocole TRDP	561
F.9.3	Essai d'écho inversé	562
F.10	Déclaration de conformité	564
Annexe G (informative)	Base d'informations de gestion (MIB) du protocole SNMP	566
G.1	Généralités	566
G.2	TTDB-MIB	566
G.3	TRDP-MIB	571
Bibliographie		583

Figure 1	– L'IEC 61375-2-3 en tant qu'élément de connexion entre le réseau central de train et l'application embarquée	296
Figure 2	– Structure de train selon l'IEC 61375-1 (exemple)	311
Figure 3	– Structure de train considérée du point de vue du profil de communication (exemple)	311
Figure 4	– Réseau de train (exemple)	312
Figure 5	– Accouplements possibles du réseau opérationnel et du réseau multimédia	313
Figure 6	– Passerelle entre le réseau opérationnel et le réseau multimédia (exemple)	315
Figure 7	– Exemple: trois rames accouplées	316
Figure 8	– Exemple: Ensemble de rames insécables	316
Figure 9	– Classification des services	320
Figure 10	– Données de notification CSTINFO	324
Figure 11	– Télégramme CSTINFOCTRL	326
Figure 12	– Diagramme de blocs de gestion de la TTDB	327
Figure 13	– Contenu de la TTDB	328
Figure 14	– Diagramme de blocs du calcul de la TTDB	329
Figure 15	– Diagramme d'états de calcul du répertoire de train	331
Figure 16	– Diagramme de classes TTDB (exemple)	335
Figure 17	– Adoption de la TTDB (illustrée par rapport à la première rame dans cet exemple)	345
Figure 18	– Espace de noms DNS TCN avec division en zones	347
Figure 19	– Schéma d'URI TCN	349
Figure 20	– Directions, orientations et numéros dans le train	350
Figure 21	– Résolution d'URI TCN dans un train	360
Figure 22	– Protocole DNS (cas a sans, cas b avec interrogation TTDB)	363
Figure 23	– Modèle de service de contrôle ETB	366
Figure 24	– Echange de télégrammes ETBCTRL	367
Figure 25	– Télégramme CSTINFOCTRL	368
Figure 26	– Diagramme d'états de calcul du répertoire de train	371
Figure 27	– Diagramme d'états de traitement ETBCTRL	372
Figure 28	– Diagramme de séquences de la fonction véhicule de tête	374
Figure 29	– Diagramme de blocs du diagramme d'états de la fonction véhicule de tête	376
Figure 30	– Diagramme d'états de la fonction véhicule de tête	378
Figure 31	– Diagramme de séquences de la fonction confirmation	381

Figure 32 – Diagramme de blocs du diagramme d'états de la fonction confirmation/correction	383
Figure 33 – Diagramme de séquences du protocole de correction/confirmation (exemple)	385
Figure 34 – Diagramme de séquences du protocole d'information (exemple)	386
Figure 35 – Diagramme d'états de la fonction confirmation/correction	387
Figure 36 – Diagramme de blocs de l'action "setCorrInfo"	389
Figure 37 – Exemples de contrôle de cohérence de la composition du train	393
Figure 38 – Calcul du répertoire opérationnel de train.....	394
Figure 39 – Diagramme d'états computeOpTrnDir	395
Figure 40 – Diagramme d'états du cas d'utilisation "mode sommeil".....	398
Figure 41 – Diagramme de séquences de la fonction mode sommeil.....	399
Figure 42 – Diagramme de blocs du diagramme d'états de la fonction mode sommeil.....	401
Figure 43 – Diagramme de séquences du protocole de demande de passage en sommeil (exemple).....	402
Figure 44 – Diagramme d'états de la fonction mode sommeil	403
Figure A.1 – Vue d'ensemble de la pile de protocole	405
Figure A.2 – Calcul d'une FCS	407
Figure A.3 – Table FCS	408
Figure A.4 – Modèle de service TRDP	409
Figure A.5 – Modèle de type push de données de processus (point à point)	412
Figure A.6 – Modèle de type push de données de processus (point à multipoint)	413
Figure A.7 – Modèle de type pull de données de processus (point à point, le destinataire connaissant la source)	414
Figure A.8 – Modèle de type pull de données de processus (multipoint à point, le destinataire ne connaissant pas la source)	415
Figure A.9 – Modèle de type pull de données de processus (point à multipoint, le destinataire connaissant la source)	416
Figure A.10 – Modèle de type pull de données de processus (multipoint à multipoint, le destinataire ne connaissant pas la source)	417
Figure A.11 – PD-PDU	418
Figure A.12 – Diagramme de séquences d'interaction pour le modèle de type pull de données de processus	426
Figure A.13 – Diagramme de séquences d'interaction pour le modèle de type push de données de processus	427
Figure A.14 – Diagramme de séquences d'interaction pour la gestion des données de processus redondantes	428
Figure A.15 – Diagramme d'états de l'éditeur des PD.....	430
Figure A.16 – Demandeur du diagramme d'états des PD	431
Figure A.17 – Abonné au diagramme d'états des PD	433
Figure A.18 – Options de transfert de données de messagerie	435
Figure A.19 – MD-PDU	437
Figure A.20 – Diagramme de séquences d'interaction	445
Figure A.21 – Diagramme d'états de l'appelant de MD de la couche TRDP	449
Figure A.22 – Diagramme d'états du répondeur de MD de la couche TRDP	453
Figure A.23 – Réception du télégramme de MD de la couche TRDP	456

Figure B.1 – Canal SDTv2	460
Figure B.2 – Etats du canal SDTv2	462
Figure B.3 – Calcul du SC-32.....	463
Figure B.4 – Table du SC-32.....	464
Figure B.5 – Calcul du SID.....	465
Figure B.6 – VDP ETB	466
Figure B.7 – Format du VDP ETB	467
Figure B.8 – Groupe de redondance (exemple avec 2 SDSRC).....	469
Figure B.9 – Diagramme d'états de l'élément SDSINK	472
Figure B.10 – Fourchette de valeurs planifiées pour le SSC (exemple)	475
Figure B.11 – Non-respect de l'intervalle de garde (exemple)	476
Figure B.12 – Diagramme de séquences du non-respect de la latence (exemple).....	478
Figure B.13 – VDP MVB.....	483
Figure B.14 – Format du VDP MVB.....	484
Figure C.1 – Diagramme d'états de configuration du TRDP	485
Figure C.2 – Paramètres d'échange avec élément central ComId.....	494
Figure C.3 – Structure de l'objet DataSet.....	505
Figure D.1 – Télégrammes de données statistiques relatives au protocole TRDP	516
Figure E.1 – Diagramme d'états d'interfaces de service.....	519
Figure E.2 – Télégrammes de l'interface ECSP.....	520
Figure E.3 – Données de contrôle de l'ECSP	521
Figure E.4 – Données des états de l'ECSP	523
Figure E.5 – Données de demande de confirmation/correction de l'ECSP	525
Figure E.6 – Données de réponse de confirmation/correction de l'ECSP	526
Figure E.7 – Télégrammes de l'interface du gestionnaire TTDB	527
Figure E.8 – Résolution des URI TCN.....	534
Figure E.9 – Données de demande de résolution du DNS	536
Figure E.10 – Données de réponse de résolution du DNS.....	537
Figure E.11 – Télégrammes de l'interface de contrôle ETBN.....	539
Figure E.12 – Données de demande de contrôle ETBN.....	540
Figure E.13 – Données de réponse d'état ETBN	541
Figure F.1 – Interface de la rame au niveau de l'ETB.....	544
Figure F.2 – Domaine d'application de l'essai de conformité	545
Figure F.3 – Architecture d'essai abstraite (1 ETB)	548
Figure F.4 – Architecture d'essai abstraite (2 ETB)	549
Figure F.5 – Architecture de l'unité en essai abstraite.....	550
Figure F.6 – Télégramme de contrôle d'essai de conformité.....	553
Figure F.7 – Données du télégramme de contrôle d'essai de conformité	554
Figure F.8 – Télégramme des états d'essai de conformité.....	555
Figure F.9 – Données du télégramme des états d'essai de conformité	555
Figure F.10 – Demande de confirmation (ou de non confirmation).....	556
Figure F.11 – Données de demande de confirmation/correction d'essai de conformité	557
Figure F.12 – Données de réponse de confirmation/correction d'essai de conformité	558

Figure F.13 – Demande de répertoire opérationnel de train de l'essai de conformité.....	559
Figure F.14 – Données de la demande de répertoire opérationnel de train de l'essai de conformité.....	559
Figure F.15 – Données de la réponse de répertoire opérationnel de train de l'essai de conformité.....	560
Figure F.16 – Essai d'écho	561
Figure F.17 – Essai d'écho inversé	563
Figure F.18 – Données du télégramme de données de messagerie de l'essai de conformité.....	563
Tableau 1 – Mots-clés et notations des data types.....	310
Tableau 2 – Service de contrôle ETB	321
Tableau 3 – Calcul du répertoire de train – triggers.....	332
Tableau 4 – Calcul du répertoire de train – guards.....	332
Tableau 5 – Calcul du répertoire de train – actions	332
Tableau 6 – Syntaxe de base de l'URI TCN	348
Tableau 7 – Schéma général de la syntaxe.....	349
Tableau 8 – Syntaxe des étiquettes d'appareil	351
Tableau 9 – Définition des étiquettes d'appareil	351
Tableau 10 – Syntaxe des étiquettes de véhicule.....	352
Tableau 11 – Définition des étiquettes de véhicule (veh).....	352
Tableau 12 – Syntaxe des étiquettes de rame	353
Tableau 13 – Définition des étiquettes de rame	353
Tableau 14 – Syntaxe d'étiquette d'ensemble de rames insécables	354
Tableau 15 – Définition des étiquettes d'ensemble de rames insécables.....	354
Tableau 16 – Syntaxe d'étiquette de train	355
Tableau 17 – Définition des étiquettes de train	355
Tableau 18 – Décomposition générale des adresses de groupes de multidiffusion IP.....	355
Tableau 19 – Décomposition des groupes relatifs à tous les trains	356
Tableau 20 – Décomposition des groupes associés à un ETB.....	356
Tableau 21 – Décomposition des groupes limités à une rame	357
Tableau 22 – URI TCN prédéfinis	358
Tableau 23 – Résolution d'URI TCN – Exemple 1	360
Tableau 24 – Résolution d'URI TCN – Exemple 2	361
Tableau 25 – Résolution d'URI TCN – Exemple 3	362
Tableau 26 – Résolution d'URI TCN – Exemple 4	362
Tableau 27 – Priorités des classes de données	365
Tableau 28 – Traitement ETBCTRL: triggers.....	372
Tableau 29 – Traitement ETBCTRL: guards.....	372
Tableau 30 – Traitement ETBCTRL: actions	373
Tableau 31 – Primitives de la fonction véhicule de tête: F_leadingStatusRequest	374
Tableau 32 – Primitives de la fonction véhicule de tête: F_leadingSetRequest.....	375
Tableau 33 – Primitives de la fonction véhicule de tête: F_leadingResetRequest.....	375
Tableau 34 – Balises de contrôle de la fonction véhicule de tête.....	375

Tableau 35 – Fonction véhicule de tête: triggers	379
Tableau 36 – Fonction véhicule de tête: guards	379
Tableau 37 – Fonction véhicule de tête: actions	380
Tableau 38 – Primitives de la fonction confirmation: F_confirmStatusRequest	381
Tableau 39 – Primitives de la fonction confirmation: F_confirmRequest	382
Tableau 40 – Primitives de la fonction confirmation: F_unconfirmRequest	382
Tableau 41 – Balises de contrôle de la fonction confirmation/correction	382
Tableau 42 – Diagramme d'états de la fonction confirmation/correction: triggers	387
Tableau 43 – Diagramme d'états de la fonction confirmation/correction: guards	388
Tableau 44 – Diagramme d'états de la fonction confirmation/correction: actions	388
Tableau 45 – Règles de confirmation/correction	390
Tableau 46 – Diagramme d'états de calcul du répertoire opérationnel de train: trigger	395
Tableau 47 – Diagramme d'états de calcul du répertoire opérationnel de train: guards	396
Tableau 48 – Diagramme d'états de calcul du répertoire opérationnel de train: action	396
Tableau 49 – Exemple de répertoire opérationnel de train	396
Tableau 50 – Conditions de fonctionnement des ETBN	397
Tableau 51 – Primitives de la fonction mode sommeil: F_sleepStatus	400
Tableau 52 – Primitives de la fonction mode sommeil: F_sleepRequest	400
Tableau 53 – Primitives de la fonction mode sommeil: F_sleepCancel	400
Tableau 54 – Primitives de la fonction mode sommeil: F_nodeAwake	400
Tableau 55 – Balises de contrôle de la fonction mode sommeil	400
Tableau 56 – Diagramme d'états de la fonction mode sommeil: triggers	403
Tableau 57 – Diagramme d'états du destinataire de MD: guards	403
Tableau 58 – Diagramme d'états de la fonction mode sommeil: actions	404
Tableau A.1 – Affectations de ports UDP/TCP	406
Tableau A.2 – ComId réservés	410
Tableau A.3 – Paramètres des PD-PDU	419
Tableau A.4 – Primitives de service TRDP	421
Tableau A.5 – Contrôle du compteur de topographie	429
Tableau A.6 – Diagramme d'états de l'éditeur des PD: guards	430
Tableau A.7 – Diagramme d'états de l'éditeur des PD: triggers	430
Tableau A.8 – Diagramme d'états de l'éditeur des PD: actions	431
Tableau A.9 – Diagramme d'états de l'éditeur des PD: états	431
Tableau A.10 – Diagramme d'états de l'éditeur des PD: guards	432
Tableau A.11 – Diagramme d'états du demandeur des PD: triggers	432
Tableau A.12 – Diagramme d'états du demandeur des PD: actions	432
Tableau A.13 – Diagramme d'états du demandeur des PD: état	432
Tableau A.14 – Diagramme d'états de l'abonné aux PD: triggers	433
Tableau A.15 – Diagramme d'états de l'abonné aux PD: guards	434
Tableau A.16 – Diagramme d'états de l'abonné aux PD: actions	434
Tableau A.17 – Diagramme d'états de l'abonné aux PD: état	434
Tableau A.18 – Paramètres des MD-PDU	438
Tableau A.19 – Primitives de service TRDP: appelant	440

Tableau A.20 – Primitives de service TRDP: répondeur	442
Tableau A.21 – Contrôle du compteur de topographie.....	446
Tableau A.22 – Diagramme d'états de l'appelant de MD: triggers.....	449
Tableau A.23 – Diagramme d'états de l'appelant de MD: guards.....	449
Tableau A.24 – Diagramme d'états de l'appelant de MD: actions	450
Tableau A.25 – Diagramme d'états de l'appelant de MD: états.....	450
Tableau A.26 – Diagramme d'états du répondeur de MD: triggers.....	454
Tableau A.27 – Diagramme d'états du répondeur de MD: guards.....	454
Tableau A.28 – Diagramme d'états du répondeur de MD: actions	454
Tableau A.29 – Diagramme d'états du répondeur de MD: états.....	455
Tableau A.30 – Diagramme d'états du destinataire de MD: triggers	456
Tableau A.31 – Diagramme d'états du destinataire de MD: guards.....	456
Tableau A.32 – Diagramme d'états du destinataire de MD: actions	457
Tableau A.33 – Diagramme d'états du destinataire de MD: état	457
Tableau B.1 – Mesures adoptées face aux erreurs de communication.....	461
Tableau B.2 – Diagramme d'états de l'élément SDSINK: triggers.....	472
Tableau B.3 – Diagramme d'états de l'élément SDSINK: guards.....	473
Tableau B.4 – Diagramme d'états de l'élément SDSINK: actions.....	473
Tableau B.5 – Compteurs de statistiques SDTv2	482
Tableau C.1 – Attributs relatifs à la balise "device".....	487
Tableau C.2 – Attributs relatifs à la balise "device-configuration".....	488
Tableau C.3 – Attributs relatifs à la balise "bus-interface"	489
Tableau C.4 – Attributs relatifs à la balise "trdp-process".....	490
Tableau C.5 – Valeurs par défaut du thread/de la tâche.....	490
Tableau C.6 – Attributs relatifs à la balise "pd-com-parameter".....	491
Tableau C.7 – Valeurs par défaut de la balise "pd-com-parameter".....	491
Tableau C.8 – Attributs relatifs à la balise "md-com-parameter".....	492
Tableau C.9 – Valeurs par défaut de la balise "md-com-parameter".....	493
Tableau C.10 – Attributs relatifs à la balise "telegram".....	495
Tableau C.11 – Attributs relatifs à la balise "md-parameter".....	496
Tableau C.12 – Attributs relatifs à la balise "pd-parameter".....	497
Tableau C.13 – Attributs relatifs à la balise "source".....	498
Tableau C.14 – Attributs relatifs à la balise "destination"	499
Tableau C.15 – Attributs relatifs à la balise "sdt-parameter".....	500
Tableau C.16 – Valeurs par défaut de la balise "sdt-parameter".....	500
Tableau C.17 – Attributs relatifs à la balise "mapped-device".....	501
Tableau C.18 – Attributs relatifs à la balise "mapped-bus-interface".....	502
Tableau C.19 – Attributs relatifs à la balise "mapped-telegram"	502
Tableau C.20 – Attributs relatifs à la balise "mapped-pd-parameter".....	502
Tableau C.21 – Attributs relatifs à la balise "mapped-source"	502
Tableau C.22 – Attributs relatifs à la balise "mapped-destination".....	503
Tableau C.23 – Attributs relatifs à la balise "mapped-SDTv2-parameter"	504
Tableau C.24 – Attributs relatifs à la balise "com-parameter".....	504

Tableau C.25 – Paramètres de communication par défaut	505
Tableau C.26 – Types de données de base	505
Tableau C.27 – Attributs relatifs à la balise "data-set"	506
Tableau C.28 – Attributs relatifs à la balise "element"	507
Tableau C.29 – Utilisation de l'élément "array-size"	508
Tableau F.1 – Récapitulatif des essais de conformité.....	565

IECNORM.COM : Click to view the full PDF of IEC 61375-2-3:2015

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

MATÉRIEL ÉLECTRONIQUE FERROVIAIRE – RÉSEAU EMBARQUÉ DE TRAIN (TCN) –

Partie 2-3: Profil de communication TCN

AVANT-PROPOS

- 1) La Commission Electrotechnique Internationale (IEC) est une organisation mondiale de normalisation composée de l'ensemble des comités électrotechniques nationaux (Comités nationaux de l'IEC). L'IEC a pour objet de favoriser la coopération internationale pour toutes les questions de normalisation dans les domaines de l'électricité et de l'électronique. A cet effet, l'IEC – entre autres activités – publie des Normes internationales, des Spécifications techniques, des Rapports techniques, des Spécifications accessibles au public (PAS) et des Guides (ci-après dénommés "Publication(s) de l'IEC"). Leur élaboration est confiée à des comités d'études, aux travaux desquels tout Comité national intéressé par le sujet traité peut participer. Les organisations internationales, gouvernementales et non gouvernementales, en liaison avec l'IEC, participent également aux travaux. L'IEC collabore étroitement avec l'Organisation Internationale de Normalisation (ISO), selon des conditions fixées par accord entre les deux organisations.
- 2) Les décisions ou accords officiels de l'IEC concernant les questions techniques représentent, dans la mesure du possible, un accord international sur les sujets étudiés, étant donné que les Comités nationaux de l'IEC intéressés sont représentés dans chaque comité d'études.
- 3) Les Publications de l'IEC se présentent sous la forme de recommandations internationales et sont agréées comme telles par les Comités nationaux de l'IEC. Tous les efforts raisonnables sont entrepris afin que l'IEC s'assure de l'exactitude du contenu technique de ses publications; l'IEC ne peut pas être tenue responsable de l'éventuelle mauvaise utilisation ou interprétation qui en est faite par un quelconque utilisateur final.
- 4) Dans le but d'encourager l'uniformité internationale, les Comités nationaux de l'IEC s'engagent, dans toute la mesure possible, à appliquer de façon transparente les Publications de l'IEC dans leurs publications nationales et régionales. Toutes divergences entre toutes Publications de l'IEC et toutes publications nationales ou régionales correspondantes doivent être indiquées en termes clairs dans ces dernières.
- 5) L'IEC elle-même ne fournit aucune attestation de conformité. Des organismes de certification indépendants fournissent des services d'évaluation de conformité et, dans certains secteurs, accèdent aux marques de conformité de l'IEC. L'IEC n'est responsable d'aucun des services effectués par les organismes de certification indépendants.
- 6) Tous les utilisateurs doivent s'assurer qu'ils sont en possession de la dernière édition de cette publication.
- 7) Aucune responsabilité ne doit être imputée à l'IEC, à ses administrateurs, employés, auxiliaires ou mandataires, y compris ses experts particuliers et les membres de ses comités d'études et des Comités nationaux de l'IEC, pour tout préjudice causé en cas de dommages corporels et matériels, ou de tout autre dommage de quelque nature que ce soit, directe ou indirecte, ou pour supporter les coûts (y compris les frais de justice) et les dépenses découlant de la publication ou de l'utilisation de cette Publication de l'IEC ou de toute autre Publication de l'IEC, ou au crédit qui lui est accordé.
- 8) L'attention est attirée sur les références normatives citées dans cette publication. L'utilisation de publications référencées est obligatoire pour une application correcte de la présente publication.
- 9) L'attention est attirée sur le fait que certains des éléments de la présente Publication de l'IEC peuvent faire l'objet de droits de brevet. L'IEC ne saurait être tenue pour responsable de ne pas avoir identifié de tels droits de brevets et de ne pas avoir signalé leur existence.

La Norme internationale IEC 61375-2-3 a été établie par le comité d'études 9 de l'IEC: Matériels et systèmes électriques ferroviaires.

La présente version bilingue (2019-01) correspond à la version anglaise monolingue publiée en 2015-07.

Le texte anglais de cette norme est issu des documents 9/2029/FDIS et 9/2048/RVD.

Le rapport de vote 9/2048/RVD donne toute information sur le vote ayant abouti à l'approbation de cette norme.

La version française de cette norme n'a pas été soumise au vote.

Cette publication a été rédigée selon les Directives ISO/IEC, Partie 2.

Une liste de toutes les parties de la série IEC 61375, publiées sous le titre général *Matériel électronique ferroviaire – Réseau embarqué de train (TCN)*, peut être consultée sur le site web de l'IEC.

Le comité a décidé que le contenu de cette publication ne sera pas modifié avant la date de stabilité indiquée sur le site web de l'IEC sous "<http://webstore.iec.ch>" dans les données relatives à la publication recherchée. A cette date, la publication sera

- reconduite,
- supprimée,
- remplacée par une édition révisée, ou
- amendée.

Le contenu des corrigenda de décembre 2015 et d'octobre 2016 a été inclus dans cette version.

IMPORTANT – Le logo "*colour inside*" qui se trouve sur la page de couverture de cette publication indique qu'elle contient des couleurs qui sont considérées comme utiles à une bonne compréhension de son contenu. Les utilisateurs devraient, par conséquent, imprimer cette publication en utilisant une imprimante couleur.

IECNORM.COM : Click to view the full PDF of IEC 61375-2-3:2015

INTRODUCTION

La série IEC 61375 spécifie un réseau embarqué de train (TCN, Train Communication Network) à des fins d'utilisation dans des véhicules ferroviaires (trains), principalement destiné à l'échange d'informations relatives aux systèmes de contrôle et de gestion de train (TCMS, Train Control and Monitoring System), sans toutefois s'y limiter. La spécification s'étend de la couche physique à la couche application; elle implique différentes technologies de communication.

La présente partie de l'IEC 61375 (IEC 61375-2-3) définit le profil de communication du réseau embarqué de train afin de garantir l'interopérabilité entre les rames connectées par des réseaux centraux de trains Ethernet (ETB, Ethernet Consist Network) tels que définis dans l'IEC 61375-2-5.

Les raisons motivant l'élaboration de la présente partie de l'IEC 61375 sont:

- la définition des exigences nécessaires à l'interopérabilité de communication au niveau des réseaux centraux de trains Ethernet;
- la documentation complète des exigences de tous les utilisateurs, en les harmonisant et en les présentant sous un format normalisé;
- la fourniture de lignes directrices pour la solution technique adoptée pour la communication interopérable des réseaux centraux de trains;
- la définition de directives d'essai de conformité (Annexe F) qui fournissent des lignes directrices pour la vérification de la conformité des rames par rapport au profil de communication.

Les applications embarquées concrètes pour certaines fonctionnalités ne sont pas traitées dans la présente partie de l'IEC 61375. Elles sont définies dans l'IEC 61375-2-4.

IECNORM.COM : Click to view the full PDF of IEC 61375-2-3:2015

MATÉRIEL ÉLECTRONIQUE FERROVIAIRE – RÉSEAU EMBARQUÉ DE TRAIN (TCN) –

Partie 2-3: Profil de communication TCN

1 Domaine d'application

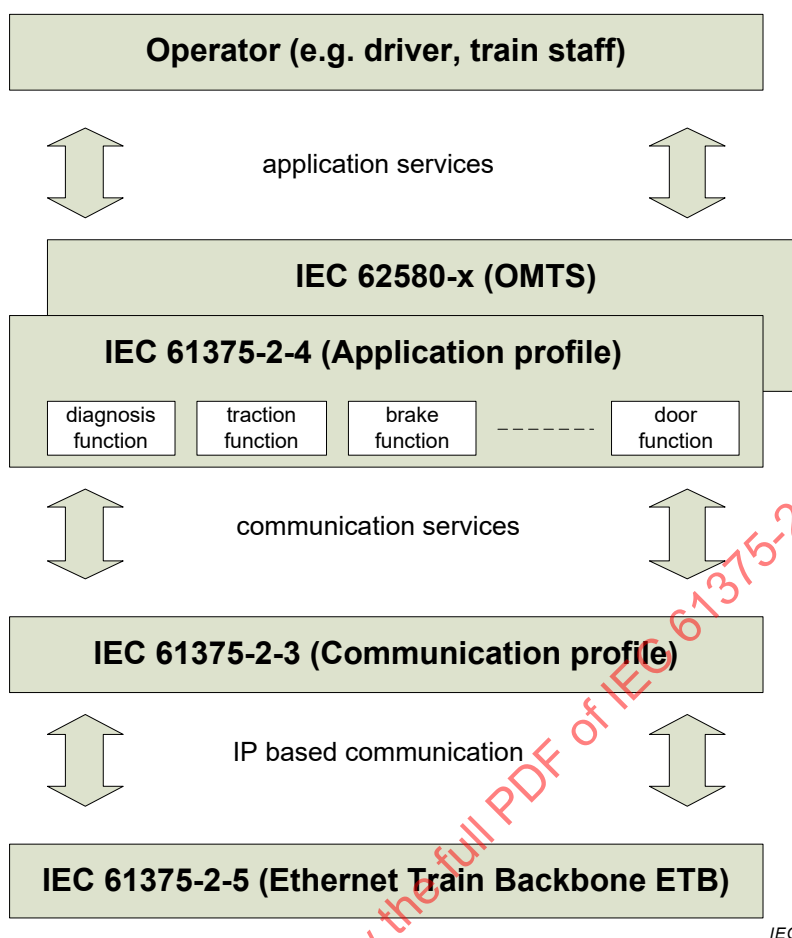
La présente partie de l'IEC 61375 spécifie les règles relatives à l'échange de données entre les rames de trains. L'agrégation de ces règles définit le profil de communication TCN.

L'objectif du profil de communication est d'assurer l'interopérabilité entre les rames desdits trains du point de vue de l'échange d'informations. Pour cela, il définit l'ensemble des éléments nécessaires à l'interopérabilité de communication:

- une architecture, avec des directions de train définies relatives à différentes vues du train;
- un concept d'adressage fonctionnel commun;
- un protocole de communication commun pour l'échange des données entre les fonctions;
- un ensemble de services pour le contrôle des communications de trains.

À titre restrictif, ce profil de communication est conforme à la technologie de réseau central de train Ethernet (ETB, Ethernet Consist Network) telle que définie dans l'IEC 61375-2-5. En ce qui concerne les réseaux de rames, une interface plus abstraite est définie sans restreindre l'application d'une quelconque technologie de réseau de rame, par exemple le bus de véhicule multifonctions (MVB, IEC 61375-3-1), le réseau de rame CANopen (CCN, IEC 61375-3-3) ou le réseau de rame Ethernet (ECN, IEC 61375-3-4).

Il n'entre pas dans les attributions du profil de communication de définir le contenu des données d'application ni leur signification (ex: syntaxe et sémantique). Cela relève de la responsabilité des profils d'application. Comme le montre la Figure 1, deux profils d'application sont explicitement pris en compte, à savoir: le profil d'application TCMS tel que défini dans l'IEC 61375-2-4 et les profils liés aux services multimédias et télématiques embarqués (OMTS, Onboard Multimedia and Telematic Services) tels que définis dans la série IEC 62580.



Anglais	Français
Operator (e.g. driver, train staff)	Opérateur (ex.: conducteur, personnel du train)
application services	services d'application
IEC 62580-x (OMTS)	IEC 62580-x (OMTS)
IEC 61375-2-4 (Application profile)	IEC 61375-2-4 (Profil d'application)
diagnosis function	fonction de diagnostic
traction function	fonction de traction
brake function	fonction de freinage
door function	fonction de porte
communication services	services de communication
IEC61375-2-3 (Communication profile)	IEC 61375-2-3 (Profil de communication)
IP based communication	communication basée sur IP
IEC61375-2-5 (Ethernet Train Backbone ETB)	IEC 61375-2-5 (Réseau central de train)

Figure 1 – L'IEC 61375-2-3 en tant qu'élément de connexion entre le réseau central de train et l'application embarquée

2 Références normatives

Les documents ci-après, dans leur intégralité ou non, sont des références normatives indispensables à l'application du présent document. Pour les références datées, seule l'édition citée s'applique. Pour les références non datées, la dernière édition du document de référence s'applique (y compris les éventuels amendements).

IEC 61375-1, *Matériel électronique ferroviaire – Réseau embarqué de train (TCN) – Partie 1: Architecture générale*

IEC 61375-2-1, *Matériel électronique ferroviaire – Réseau embarqué de train (TCN) – Partie 2-1: Bus de Train Filaire (WTB)*

IEC 61375-2-4, *Electronic railway equipment – Train communication network (TCN) – Part 2-4: Application Profile* (disponible seulement en anglais)

IEC 61375-2-5, *Matériel électronique ferroviaire – Réseau embarqué de train (TCN) – Partie 2-5: Réseau central de train Ethernet*

IEC 62280, *Applications ferroviaires – Systèmes de signalisation, de télécommunication et de traitement – Communication de sécurité dans le système de transmission*

ISO/IEC 9646-6:1994, *Technologies de l'information – Interconnexion de systèmes ouverts (OSI) – Cadre général et méthodologie des tests de conformité OSI – Partie 6: spécification de test pour les profils de protocoles*

ISO/IEC 9646-7:1995, *Information technology – Open Systems Interconnection – Conformance testing methodology and framework – Part 7: Implementation Conformance Statements* (disponible seulement en anglais)

ISO/IEC 17011:2004, *Evaluation de la conformité – Exigences générales pour les organismes d'accréditation procédant à l'accréditation d'organismes d'évaluation de la conformité*

ISO/IEC 17025:2005, *Exigences générales concernant la compétence des laboratoires d'étalonnages et d'essais*

3 Termes, définitions, abréviations, acronymes et conventions

3.1 Termes et définitions

Pour les besoins du présent document, les termes et définitions suivants s'appliquent.

3.1.1

couche application

couche supérieure du modèle OSI en interface directe avec l'application

3.1.2

interface de couche application

définition des services offerts par la couche application

3.1.3

processus d'application

élément d'un système ouvert réel qui effectue le traitement de l'information pour une application particulière

3.1.4

big-endian

gros-boutiste

définition selon laquelle l'octet de poids fort des données à octets multiples est stocké à l'emplacement mémoire ayant la plus petite adresse mémoire et envoyé en premier dans la séquence de transmissions sérialisées

3.1.5

pont

appareil qui stocke et retransmet les trames d'un bus à l'autre en fonction de leurs adresses de couche liaison

3.1.6

diffusion

transmission quasi simultanée des mêmes informations vers plusieurs destinataires

Note 1 à l'article: La diffusion dans le TCN n'est pas considérée comme fiable, c'est-à-dire que certains destinataires peuvent recevoir les informations et d'autres pas.

3.1.7

ensemble de rames insécables

train composé d'une rame ou d'un ensemble de rames, dont la composition ne change pas en exploitation normale

EXEMPLE métro, train suburbain ou rame à grande vitesse

3.1.8

rame insécable

rame d'un ensemble de rames insécables

3.1.9

appareils de communication

appareils reliés au réseau de rame ou au réseau central de train, capables de transporter, d'émettre ou de recevoir des données

3.1.10

composition

nombre, séquence, orientation et caractéristiques des rames composant un train

3.1.11

configuration

définition de la topologie d'un réseau, des appareils qui lui sont connectés, de leurs capacités et du trafic qu'ils génèrent

3.1.12

rame

véhicule isolé ou groupe de véhicules n'étant pas séparé en exploitation normale

Note 1 à l'article: Une rame peut comporter aucun, un ou plusieurs réseaux de rames.

3.1.13

réseau de rame

réseau de communication reliant des appareils de communication dans une rame

3.1.14

adresse réseau de rame

adresse réseau qui n'est pas modifiée après l'inauguration et qui est utilisée pour l'adressage de l'appareil de communication dans le propre réseau de rame

3.1.15

numéro de séquence de rame

numéro de séquence de la rame dans le train obtenu lors de l'inauguration du TCN

3.1.16

consommateur

destinataire d'un message au niveau de la couche transport

3.1.17**appareil destinataire**

destinataire d'un paquet de données

3.1.18**véhicule/rame découvert(e)**

véhicule/rame découvert(e) durant l'inauguration du train

3.1.19**appareil terminal**

entité reliée à un réseau de rame ou à un ensemble de réseaux de rames préparé pour des raisons de redondance

3.1.20**Réseau central de train Ethernet****Ethernet Train Backbone**

réseau central de train basé sur une technologie Ethernet commutée

Note 1 à l'article: Selon les spécifications de l'IEC 61375-2-5.

3.1.21**Nœud de réseau central de train Ethernet****Ethernet Train Backbone Node**

nœud relié au réseau central de train Ethernet, recevant un numéro séquentiel lors de l'inauguration et capable de transmettre des paquets de données utilisateurs entre le réseau de rame et le réseau central de train

3.1.22**fonction**

processus d'application qui échange des messages avec un autre processus d'application

3.1.23**passerelle**

liaison entre différentes technologies de communication

3.1.24**adresse de groupe**

adresse d'un groupe de multidiffusion auquel appartient un appareil

3.1.25**intégrité**

capacité d'un système à reconnaître et à éliminer des données erronées en cas de dysfonctionnement de ses éléments constitutifs

3.1.26**véhicule de tête**

véhicule qui contrôle les mouvements du train

3.1.27**topologie linéaire**

topologie de liaison en série des nœuds selon laquelle deux nœuds sont reliés chacun à un seul autre nœud, les nœuds restants étant tous reliés à deux autres nœuds (c'est-à-dire reliés sous la forme d'une ligne)

3.1.28**réseau local****LAN**

portion d'un réseau caractérisée par un mode d'accès et par un espace d'adressage communs

Note 1 à l'article: L'abréviation "LAN" est dérivée du terme anglais développé correspondant "Local Area Network".

3.1.29

contrôle d'accès au support

MAC

sous-couche de la couche liaison qui contrôle l'accès au support (arbitrage, transfert de maîtrise, interrogation)

Note 1 à l'article: L'abréviation "MAC" est dérivée du terme anglais développé correspondant "medium access control".

3.1.30

support

support physique de propagation du signal: câbles électriques, fibres optiques, etc.

3.1.31

message

donnée transmise en un ou plusieurs paquets

3.1.32

multidiffusion

transmission du même message à un groupe de destinataires identifiés par leur adresse de groupe

Note 1 à l'article: Le mot "multidiffusion" est également utilisé lorsque le groupe comprend tous les destinataires.

3.1.33

réseau multimédia

réseau d'un train servant à la transmission de données multimédias et télématiques

3.1.34

services multimédias

services servant à des fonctions multimédias

Note 1 à l'article: Les services multimédias et télématiques embarqués sont définis dans l'IEC 62580-1.

3.1.35

unité multiple

certain nombre de véhicules groupés pour former une unité servant au transport de voyageurs et/ou de fret

Note 1 à l'article: Les cabines de conduite de l'unité multiple peuvent se trouver dans des motrices spéciales (ex: locomotives monocabines) ou dans des véhicules d'extrémité équipés de cabines de conduite. Des groupes de véhicules de l'unité multiple peuvent être assemblés pour former des rames.

3.1.36

train à unités multiples

train composé d'un ensemble de rames insécable, la composition de cet ensemble pouvant évoluer en exploitation normale

3.1.37

réseau

ensemble de différents systèmes possibles de communication qui échangent de l'information selon un moyen communément accepté

3.1.38

adresse réseau

adresse qui identifie un appareil de communication sur une couche réseau

3.1.39**appareil réseau**

composants utilisés pour la configuration de réseaux de rames et de réseaux de trains

Note 1 à l'article: Il peut s'agir de composants passifs tels que des câbles ou des connecteurs, de composants actifs non gérés tels que des répéteurs, des convertisseurs de supports ou des commutateurs (non gérés), ou de composants actifs gérés tels que des passerelles, des routeurs et des commutateurs (gérés).

3.1.40**couche réseau**

couche du modèle OSI assurant le routage entre les différents bus

3.1.41**gestion de réseau**

opérations nécessaires à la configuration, la surveillance, le diagnostic et la maintenance à distance du réseau

3.1.42**nœud**

appareil placé sur le réseau central de train, pouvant servir de passerelle entre le réseau central de train et le réseau de rame

3.1.43**réseau opérationnel**

réseau d'un train servant à la transmission de données d'exploitation de train et de données critiques pour la sécurité

3.1.44**vision opérationnelle du train**

vision d'un train par le conducteur ou le personnel du train, perçue comme une séquence de véhicules

3.1.45**octet**

mot de 8 bits stocké dans la mémoire ou transmis sous forme d'unité

3.1.46**sous-système multimédia et télématique embarqué**

ensemble de services et de fonctions embarqués pour le multimédia et la télématique

Note 1 à l'article: Une définition de ces services est donnée dans l'IEC 62580.

3.1.47**services opérationnels**

tous les services nécessaires pour exploiter un train en toute sécurité

3.1.48**direction opérationnelle du train**

direction d'un train telle qu'elle est vue par le conducteur ou le personnel du train

3.1.49**répertoire opérationnel de train**

structure de données décrivant la composition du train telle qu'elle est vue par le conducteur ou le personnel du train, y compris les informations sur le véhicule de tête et les sauts de véhicules

Note 1 à l'article: Le répertoire opérationnel de train fait partie de la base de données topologiques du train.

3.1.50

opérateur

entreprise ou organisme dont l'activité consiste à exploiter des trains

3.1.51

paquet

unité de message (information, acquittement ou contrôle) transmise sur la couche réseau ou la couche transport par des protocoles

3.1.52

PD-PDU

unité de données pour la transmission des données de processus

Note 1 à l'article: Il existe deux types de PD-PDU. Tous les types de PD-PDU ont le même format de PDU. Une PD-PDU (données) est une PD-PDU contenant des données utilisateur et envoyée d'un éditeur vers un ou plusieurs abonnés. Cette procédure est utilisée dans les modèles de communication de types push et pull. Les PD-PDU (données) du modèle de type pull sont également appelées PD-PDU (réponse). Une PD-PDU (demande) est une PD-PDU envoyée par un demandeur vers un ou plusieurs éditeurs. Elle peut ou peut ne pas contenir des données utilisateur.

3.1.53

éditeur de données de processus

appareil qui envoie cycliquement des télégrammes de données de processus avec un ComId défini à une adresse ou une plage d'adresses définies

3.1.54

demandeur de données de processus

appareil qui demande qu'un simple télégramme de données de processus avec un ComId défini soit envoyé à une adresse ou une plage d'adresses spécifiées

3.1.55

abonné de données de processus

appareil qui reçoit des télégrammes de données de processus avec un ComId défini et un filtre d'adresse source éventuel

3.1.56

producteur

émetteur d'un message au niveau de la couche transport

3.1.57

éditeur

source d'un dataset en diffusion (voir abonné)

3.1.58

récepteur

appareil électronique qui peut recevoir des signaux en provenance du support physique

3.1.59

groupe de redondance

groupe d'appareils dans lequel seul un d'entre eux est actif et les autres en veille

Note 1 à l'article: Si l'appareil actif est défaillant, l'un des appareils en veille prend le relais. Un groupe de redondance comprend en général deux appareils.

3.1.60

transmission normale

transmission de données non relatives à la sécurité

3.1.61
répéteur

interconnexion entre des segments de bus au niveau de la couche physique permettant une extension du bus au-delà des limites admises par des composants passifs

Note 1 à l'article: Les segments interconnectés fonctionnent à la même vitesse et selon le même protocole. Le retard introduit par un répéteur est de l'ordre d'un temps de bit.

3.1.62
taux d'erreur résiduel

probabilité d'atteinte à l'intégrité (bit erroné non reconnu) par bit transmis

3.1.63
topologie en anneau

réseau actif dans lequel chaque nœud est connecté en série à deux autres nœuds

Note 1 à l'article: L'anneau peut également être désigné comme une boucle.

[SOURCE: IEC 61918:2013, 3.1.63]

3.1.64
routeur

interconnexion entre deux bus au niveau de la couche réseau, transférant des télégrammes d'un bus à l'autre sur la base de leurs adresses réseau

3.1.65
transmission de données de sécurité

transmission de données relatives à la sécurité

3.1.66
service

capacités et moyens d'un sous-système offerts à un utilisateur

3.1.67
données apériodiques

transmission de données suite à une demande

3.1.68
abonné

un des destinataires d'un dataset en diffusion (voir éditeur)

3.1.69
commutateur
pont MAC

Note 1 à l'article: Selon la définition de l'IEEE 802.1D.

3.1.70
topologie

interconnexion de câbles et de plusieurs appareils éventuels dans un réseau donné

3.1.71
compteur de topographie

identificateur unique d'une composition de train

3.1.72
train

composition d'une rame ou d'un ensemble de rames, dont la configuration peut changer en exploitation normale

3.1.73

réseau de communication de train

réseau de communication de données permettant de connecter des équipements électroniques programmables embarqués sur des véhicules ferroviaires

3.1.74

système de communication de train

réseau embarqué de train, plus son profil de communication et son profil d'application pour l'échange d'informations sur l'ensemble du train

3.1.75

système de contrôle et de gestion de train

ensemble de services et de fonctions embarqués de train pour l'exploitation d'un train

Note 1 à l'article: Une définition de ces services est donnée dans l'IEC 61375-2-4.

3.1.76

réseau central de train

bus connectant les rames d'un train

Note 1 à l'article: Conforme à l'IEC 61375-2-1 (WTB) ou l'IEC 61375-2-5 (ETB).

3.1.77

nœud de réseau central de train

appareil du réseau central de train qui reçoit un numéro de nœud de réseau central de train lors de l'inauguration

Note 1 à l'article: Un nœud de réseau central de train peut être utilisé pour relier un appareil terminal ou un réseau de rame au réseau central de train.

3.1.78

numéro de nœud de réseau central de train

adresse de nœud

numéro du nœud

numéro attribué à chaque nœud de réseau central de train lors de l'inauguration, qui indique la position du nœud de réseau central de train sur ledit réseau

3.1.79

vision du réseau central de train

train vu comme une séquence de réseaux de rames, déterminé par l'inauguration ETB

3.1.80

répertoire de train

structure de données décrivant la composition du train après inauguration du train

Note 1 à l'article: Le répertoire de train fait partie de la base de données topologiques du train.

3.1.81

adresse réseau de train

adresse réseau dynamique utilisée pour l'adressage d'appareils de communication dans d'autres réseaux de rames

Note 1 à l'article: Cette adresse peut changer après chaque inauguration.

3.1.82

gestion de réseau de train

services de gestion de réseau TCN

3.1.83**calcul du répertoire de train**

protocole permettant de déterminer la composition réelle d'un train comme un ensemble de rames contenant des véhicules constitués d'appareils de fonction

Note 1 à l'article: Les ensembles de rames insécables facultatifs représentés en tant que rames peuvent être détectés et interprétés comme un ensemble de rames. Le calcul du répertoire de train a pour résultat la structure de données du répertoire de train qui fait partie de la base de données topologiques du train.

3.1.84**direction de référence du train**

direction déterminée par l'inauguration

3.1.85**base de données topologiques de train**

base de données permettant de rassembler toutes les informations sur la composition réelle d'un train

Note 1 à l'article: Une instance de cette base de données est calculée pour chaque rame du train.

3.1.86**vision du train**

train vu comme une séquence de rames, déterminé par l'inauguration ETB

3.1.87**couche transport**

couche du modèle OSI responsable du contrôle de flux de bout en bout et de la reprise sur erreur

3.1.88**numéro d'identification de véhicule UIC**

numéro à 12 chiffres servant à l'identification unique des véhicules

Note 1 à l'article: Selon la définition de l'UIC.

3.1.89**PDU non spécifiée**

PDU TRDP avec un ComId=0

Note 1 à l'article: Le contenu des données utilisateur d'une PDU non spécifiée est indéterminé pour le profil de communication. Différentes PDU et leur contenu peuvent être distingués uniquement au niveau application.

3.1.90**véhicule**

voiture ou locomotive individuelle

3.1.91**saut de véhicule**

saut

véhicule figurant dans la liste des véhicules confirmés du répertoire opérationnel de train, ne pouvant pas être atteint sur l'ETB

Note 1 à l'article: Un saut de véhicule peut se trouver à l'une des extrémités ou au milieu d'un train. Il peut également y avoir plus d'un saut de véhicule dans un train.

Note 2 à l'article: Les sauts de véhicules peuvent être déterminés par le conducteur en comparant la composition du train non confirmée indiquée avec la composition réelle du train, conduisant à une correction de la composition du train.

3.2 Abréviations et acronymes

ALG Application Layer Gateway (Passerelle de couche application)

API	Application Programming Interface (Interface de programmation d'application)
ASCII	American Standard Code for Information Interchange (Code normalisé américain pour l'échange d'informations)
CN	Consist Network (Réseau de rame)
ComId	Communication Identifier (Identificateur de communication)
CRC	Cyclic Redundancy Check (Contrôle de redondance cyclique)
CST	Consist (Rame)
CSTINFO	Consist Information Telegram (Télégramme d'informations de rame)
CC	Courant Continu
DEV	Device (Appareil)
DNS	Domain Name System (Système de noms de domaine)
DoS	Denial of Service (Déni de service)
FCS	Frame Check Sequence (Séquence de contrôle de trame)
ECN	Ethernet Consist Network (Réseau de rame Ethernet)
ECSC	ETB Control Service Client (Client du service de contrôle du réseau central de train Ethernet)
ECSP	ETB Control Service Provider (Fournisseur de service de contrôle du réseau central de train Ethernet)
ED	End Device (Appareil terminal)
ETB	Ethernet Train Backbone (Réseau central de train Ethernet)
ETBCTRL	ETB Control packet (Paquet de contrôle du réseau central de train Ethernet)
ETBN	Ethernet Train Backbone Node (Nœud de réseau central de train Ethernet)
FCT	Fonction
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IP	Internet Protocol (Protocole Internet)
LEN	Length (Longueur)
MAC	Medium Access Control (Contrôle d'accès au support)
MC	Multicast (Multidiffusion)
MD	Message Data (Données de messagerie)
MD-PDU	Message Data Protocol Data Unit (Unité de données de protocole de données de message)
MIB	Management Information Base (Base d'informations de gestion)
MTU	Maximal Transmission Unit (Unité de transmission maximale)
MVB	Multifunction Vehicle Bus (Bus de véhicule multifonctions)

NAT	Network Address Translation (Traduction d'adresse réseau)
NOK	Not OK (Pas OK)
OID	Object Identifier (Identificateur d'objet)
OMTS	Onboard Multimedia and Telematic Subsystem (Sous-système multimédia et télématique embarqué)
OSI	Open Systems Interconnection (Interconnexion de systèmes ouverts)
PD	Process Data (Données de processus)
PD-PDU	Process Data Protocol Data Unit (Unité de données de protocole de données de processus)
PICS	Protocol Implementation Conformance Statement (Déclaration de conformité de mise en œuvre de protocole)
PROP	Propriétés
QoS	Quality of Service (Qualité de service)
RFC	Request For Comments (Demande de commentaires)
SDSINK	Safe Data Sink (Destinataire de données sécurisées)
SDSRC	Safe Data Source (Source de données sécurisées)
SDTv2	Safe Data Transmission version 2 (Transmission de données sécurisées version 2)
SID	Source Identifier (Identificateur source)
SIL	Safety Integrity Level (Niveau d'intégrité de la sécurité)
SMI	Safe Message Identifier (Identificateur de message sécurisé)
SNMP	Simple Network Management Protocol (Protocole simple de gestion de réseau)
SSC	Safety Sequence Counter (Compteur de séquence de sécurité)
STC	Safe Topography Counter (Compteur de topographie de sécurité)
TCN	Train Communication Network (Réseau embarqué de train)
TCMS	Train Control and Monitoring System (Système de contrôle et de gestion de train)
TCP	Transmission Control Protocol (Protocole de contrôle de transmission)
TP	Test Procedure (Procédure d'essai)
TSS&TP	Test Suite Structure and Test Purpose (Structure et objet de la suite d'essais)
TRDP	Train Real Time Data Protocol (Protocole temps réel de données de train)
TTDB	Train Topology Database (Base de données topologiques de train)
TTL	Time to Live (Durée de vie)
TTDP	Train Topology Discovery Protocol (Protocole de détermination de topologie de train)
UDP	User Datagram Protocol (Protocole de datagramme d'utilisateur)
UIC	Union Internationale des Chemins de Fer

UML	Unified Modeling Language (Langage de modélisation unifié)
URI	Uniform Resource Identifier (Identificateur uniforme de ressource)
URL	Uniform Resource Locator (Localisateur uniforme de ressource)
UUID	Universally Unique Identifier (Identificateur unique universel)
UuT	Unit under Test (Unité à l'essai)
VDP	Vital Data Packet (Paquet de données stratégiques)
VLAN	Virtual Local Area Network (Réseau local virtuel)
WTB	Wire Train Bus (Bus de train filaire)
XML	Extended Markup Language
XSD	XML Schema Definition (Définition de schémas XML)

3.3 Conventions

3.3.1 Base des valeurs numériques

La présente partie de l'IEC 61375 utilise une représentation décimale pour toutes les valeurs numériques, sauf mention contraire.

Les valeurs analogiques et fractionnaires comportent une virgule.

EXEMPLE La tension est de 20,0 V.

Les valeurs binaires et hexadécimales sont représentées selon la convention ASN.1 (ISO/IEC 8824).

EXEMPLE Le nombre décimal 20 est codé sur 8 bits = '0001 0100'B = '14'H.

3.3.2 Chaînes de caractères et citations

Les chaînes de caractères sont placées entre guillemets doubles (").

EXEMPLE "ComProfTestAppl".

Les guillemets simples (') sont utilisés pour les citations.

EXEMPLE paramètre 'etbTopoCnt'.

Parfois, les chaînes de caractères sont des valeurs de paramètres, auquel cas elles peuvent être écrites comme des citations.

EXEMPLE type de télégramme 'Mn'.

3.3.3 Conventions d'appellation

La première lettre des mots-clés est une majuscule.

Lorsque le nom d'un mot-clé est un nom composé, les différentes parties du nom composé sont séparées par un espace, la première lettre de toutes les parties étant en majuscule.

EXEMPLES "Réseau Central de Train", "Rame", "Réseau de Rame".

La première lettre des paramètres est une majuscule.

Lorsque le nom d'un paramètre est un nom composé, les différentes parties du nom ne sont pas séparées par un espace, la première lettre de toutes les parties, sauf la première partie, étant une majuscule.

EXEMPLE paramètre 'etbTopoCnt'.

3.3.4 Conventions pour les diagrammes

Partout où cela est approprié, des types de diagrammes UML sont utilisés. En particulier, les types de diagrammes UML suivants sont appliqués:

- Diagrammes d'états UML
- Diagramme de séquences UML

Les conditions utilisées dans ces types de diagrammes font appel aux conventions suivantes:

opérateurs	=	Opérateur d'affectation: donner une valeur
	==	Opérateur d'égalité: égal à
	!=	Opérateur d'égalité: différent de
	OR	Opérateur logique: {expression} OR {expression}
	AND	Opérateur logique: {expression} AND {expression}
variable booléenne	FALSE	Antivalent2 (variable antivalente) a la valeur 0: 01 _b
	TRUE	Antivalent2 (variable antivalente) a la valeur 1: 10 _b

Les triggers et les guards utilisés dans ces types de diagrammes font appel aux conventions suivantes:

- Un trigger est un événement extérieur, par exemple une commande utilisateur, qui déclenche une transition d'état.
- Dans le cas particulier où un trigger est la réception d'un paquet ETBCTRL (périodique), les balises de contrôle présentes dans le paquet ETBCTRL déterminent la transition d'état. Pour ces cas uniquement, les guards sont indiqués dans les diagrammes d'états UML.

3.3.5 Annotation des structures de données

3.3.5.1 Généralités

Les structures de données sont définies en suivant la présentation et le codage des données transmises et stockées comme défini dans l'IEC 61375-2-1, qui se base sur la syntaxe ISO ASN.1.

Toutes les données au sein d'une structure de données sont organisées au format big-endian (octet de poids fort d'un élément de donnée transmis en premier).

Toutes les données au sein d'une structure de données sont harmonisées naturellement (une adresse de décalage, multiple de leur dimension, est stockée).

Les valeurs de données des paramètres au sein de structures de données ou de télégrammes marqués 'défini par l'application' sont définies dans l'IEC 61375-2-4 ou doivent être définies par l'utilisateur.

3.3.5.2 Data types supplémentaires

Outre les data types définis dans l'IEC 61375-2-1, les data types suivants sont utilisés:

(1) LABEL ::= ARRAY [16] OF CHAR

(2) AntivalentType ::= ANTIVALENT8

avec l'encodage suivant:

7	6	5	4	3	2	1	0	Interprétation
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
0	0	0	0	0	0	0	1	FALSE
0	0	0	0	0	0	1	0	TRUE
0	0	0	0	0	0	1	1	UNDEFINED
Toute autre valeur								ERROR

3.3.5.3 Mots-clés des data types

Outre les mots-clés/notations définis dans l'IEC 61375-2-1 comme une extension de la notation ASN.1, des mots-clés/notations plus courts mais équivalents sont utilisés dans la présente partie de l'IEC 61375 dans le but d'améliorer la lisibilité des structures de données. Ces mots-clés sont les suivants (voir Tableau 1):

Tableau 1 – Mots-clés et notations des data types

Mot-clé/notation utilisé dans la présente partie de l'IEC 61375	Mot-clé/notation équivalent utilisé dans l'IEC 61375-2-1
UINT#	UNSIGNED#
INT#	INTEGER#
CHAR	CHARACTER8
Type [IndexList]	ARRAY [IndexList] OF Type
(# = entier non signé)	

4 Architecture

4.1 Généralités

Le présent article définit l'architecture de communication relative au train qui sert de base au profil de communication. L'architecture de communication relative au train est définie de deux points de vue, à savoir l'architecture physique du train et l'architecture logique du train. L'architecture physique du train donne une décomposition du système en considérant les composants et les interfaces en jeu tandis que l'architecture logique du train se base sur une décomposition fonctionnelle ayant trait aux fonctions et à leurs interactions. Les deux points de vue sont nécessaires pour définir des concepts de profil de communication importants, comme la définition de la base de données topologiques (TTDB) du train ou la définition d'un concept d'adressage logique pour l'ensemble du train basé sur les URI.

4.2 Architecture physique du train (décomposition du système)

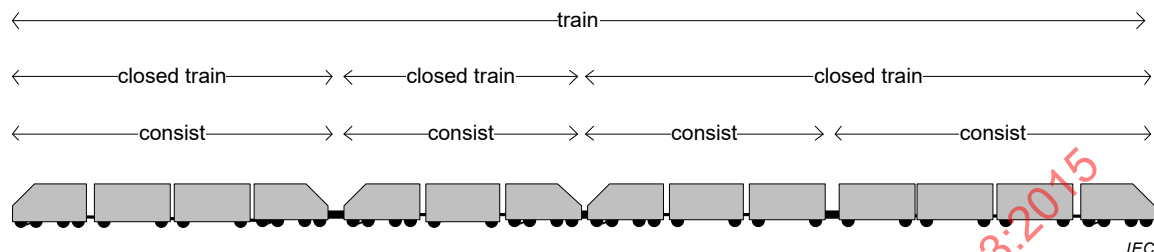
4.2.1 Généralités

L'architecture physique du train définit un train comme étant composé d'ensembles de rames insécables, de rames, de véhicules, d'appareils et des interfaces associées.

4.2.2 Architectures de réseau de train

4.2.2.1 Structure de train

L'IEC 61375-1 définit un train en tant que composition d'ensembles de rames insécables et de rames, chaque rame comportant un ou plusieurs véhicules et chaque ensemble de rames insécable comportant une ou plusieurs rames comme le montre la Figure 2.

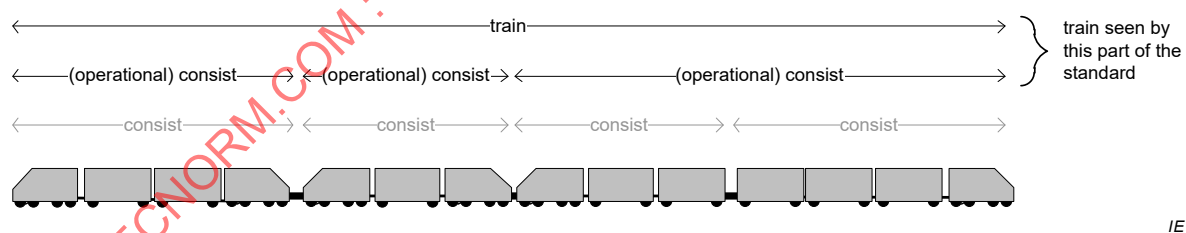


Anglais	Français
train	train
closed train	ensemble de rames insécables
consist	rame

Figure 2 – Structure de train selon l'IEC 61375-1 (exemple)

La restriction suivante s'applique à la présente partie de la norme:

Bien qu'un ensemble de rames insécables soit une composition d'une ou de plusieurs rames (rames insécables), il doit être considéré comme une rame opérationnelle dans le domaine d'application de ce profil de communication. Dans ce sens, un train est seulement composé de rames (opérationnelles), comme le montre la Figure 3, un certain nombre des rames opérationnelles pouvant être un ensemble de rames insécable. Par conséquent, tout ensemble de rames insécable doit aussi se comporter comme une rame du point de vue de ce profil de communication.



Anglais	Français
train	train
(operational) consist	rame (opérationnelle)
consist	rame
train seen by this part of the standard	train considéré du point de vue de la présente partie de la norme

Figure 3 – Structure de train considérée du point de vue du profil de communication (exemple)

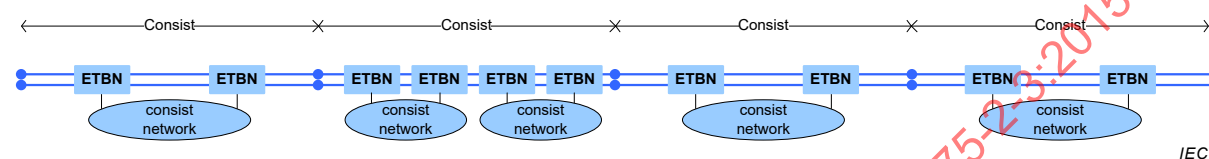
NOTE 1 La communication au sein d'un ensemble de rames insécables, par exemple entre les rames insécables qui constituent l'ensemble de rames insécables, n'entre pas dans le domaine d'application de la présente norme.

Ce profil de communication est applicable aux trains ayant un nombre maximal de 63 véhicules.

NOTE 2 La façon dont les véhicules sont affectés aux rames dépend de la conception de rame choisie. Par exemple, chaque véhicule d'un train peut être une rame (auquel cas un train peut alors comporter 63 rames) ou deux trains formés de 32 et 31 véhicules forment une rame (auquel cas le train ne comporte que 2 rames).

4.2.2.2 Affectation d'un réseau de train

L'IEC 61375-1 définit un réseau couvrant le train comme une composition de réseaux de rames interconnectés par un réseau central de train commun, comme le montre l'exemple de la Figure 4, dans lequel un ou plusieurs réseaux de rames par rame sont connectés à l'ETB par l'intermédiaire d'une paire redondante d'ETBN.



Anglais	Français
Consist	Rame
ETBN	ETBN
consist network	réseau de rame

Figure 4 – Réseau de train (exemple)

De plus, l'IEC 61375-1 permet l'application de 4 réseaux centraux de train maximum en parallèle.

Ce profil de communication définit l'utilisation des réseaux centraux de train de la manière suivante:

- Un ETB (avec réseaux de rames subordonnés) doit être utilisé en tant que réseau opérationnel, prenant en charge les services TCMS.
- Zéro, un ou plusieurs ETB (avec réseaux de rames subordonnés) doivent être utilisés en tant que réseau multimédia, prenant en charge les services OMTS.

L'ETB opérationnel peut également être utilisé pour les services OMTS.

NOTE 1 Le fait d'avoir des services TCMS et des services OMTS qui se partagent un ETB (opérationnel) peut servir à des applications simples.

NOTE 2 Si un réseau physique est utilisé à la fois pour des services TCMS et des services OMTS, une séparation peut être faite à une couche supérieure, par exemple sur la couche liaison en utilisant un VLAN. Il faut s'attendre cependant à ce que les exigences relatives à la sécurité et à la sûreté soient beaucoup plus difficiles à satisfaire pour ce type de réseau.

4.2.2.3 Redondance d'ETBN

Au moins un ETBN doit être fourni par réseau de rame. Pour des raisons de redondance, il convient de fournir au moins deux ETBN sur les unités de traction et sur le raccordement au réseau central de train des équipements de cabine de conduite des voitures-pilotes (cela s'applique également aux unités multiples).

4.2.2.4 Aspects relatifs à la sécurité et à la sûreté

Le réseau opérationnel doit être un système de transmission de catégorie 1 (préférentiel) ou 2 tel que défini dans l'IEC 62280.

4.2.2.5 Espace d'adressage IP

Le réseau opérationnel et le réseau multimédia doivent partager un espace d'adressage IP commun comme défini dans l'IEC 61375-2-5 (c'est-à-dire 10.128/9). L'espace d'adressage du réseau central de train doit être identifié pour chaque réseau.

NOTE Si les deux réseaux sont séparés physiquement, l'identification du réseau central de train, le 10^e et le 11^e bits de l'adresse réseau du train, servent à l'identification.

4.2.2.6 Interconnexion du réseau opérationnel et du réseau multimédia

4.2.2.6.1 Dépendance de l'ETB

Afin de garantir l'interopérabilité entre les trains comportant plusieurs ETB, toutes les fonctions connectées à ces ETB doivent partager une vision commune du train et une vision opérationnelle commune du train telles que définies en 4.2.4.1.

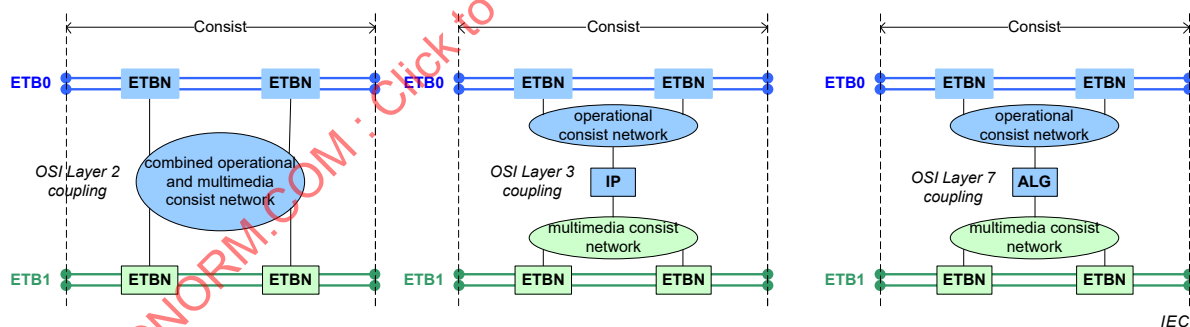
NOTE L'impact de cette exigence sur la TTDB est défini en 5.3.4.2.

4.2.2.6.2 Accouplements de réseaux

Il existe plusieurs façons d'accoupler le réseau opérationnel et le réseau multimédia (voir Figure 5).

- Les deux réseaux peuvent partager un réseau de rame (CN) commun, auquel cas les deux réseaux sont accouplés sur la couche OSI 2.
- Les deux réseaux peuvent être interconnectés par un routeur IP (éventuellement doté de fonctions pare-feu), ce qui se traduit par un accouplement sur la couche OSI 3.
- Les deux réseaux peuvent être séparés par une passerelle de couche application (ALG), ce qui se traduit par un accouplement sur la couche OSI 7.

NOTE Les routeurs IP modernes dotés de fonctions pare-feu permettent également un filtrage sur le contenu (donnés d'application). Dans ce cas, les réseaux sont accouplés sur les couches 3 et 7.



Anglais	Français
Consist	Rame
ETB0	ETB0
ETBN	ETBN
OSI Layer 2 coupling	accouplement couche OSI 2
OSI Layer 3 coupling	accouplement couche OSI 3
OSI Layer 7 coupling	accouplement couche OSI 7
operational consist network	réseau de rame opérationnel
multimedia consist network	réseau de rame multimédia
ETB1	ETB1
ETBN	ETBN

Figure 5 – Accouplements possibles du réseau opérationnel et du réseau multimédia

S'il existe un couplage sur la couche OSI 3 au moins, il est dit que les ED connectés au réseau de rame opérationnel sont associés à l'ETB opérationnel (ETB0) et que les ED connectés au réseau de rame multimédia sont associés à l'ETB multimédia (ETB1).

Dans le cas d'un accouplement sur la couche OSI 2, cette relation est indéterminée.

NOTE La relation entre les appareils terminaux est une information pertinente pour les affectations de groupe de multidiffusion IP (voir 5.4.5.2).

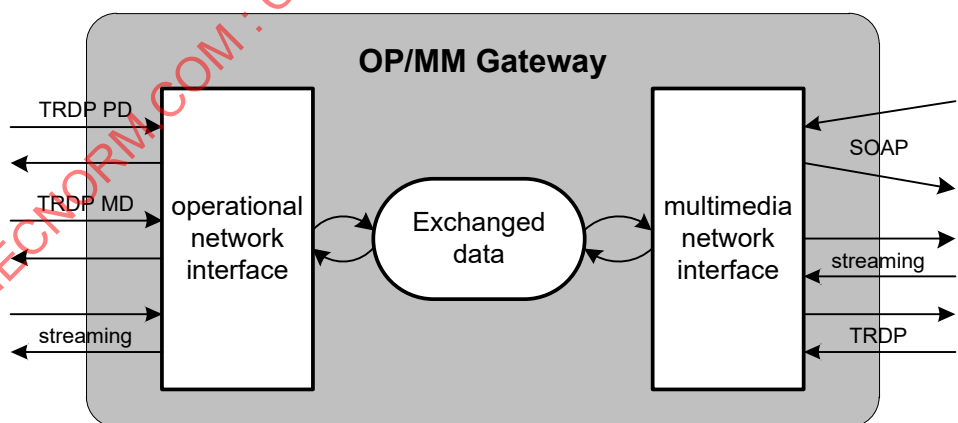
4.2.2.6.3 Echange de données entre le réseau opérationnel et le réseau multimédia

Le mode d'échange des données entre le réseau opérationnel et le réseau multimédia dépend de l'accouplement des deux réseaux. Alors que les accouplements sur les couches OSI 2 et 3 permettent un échange de données sur la base d'un télégramme IP qui est majoritairement traité au sein des couches de communication des systèmes d'exploitation modernes, un accouplement sur la couche OSI 7 permet un meilleur contrôle applicatif des données échangées.

L'exemple suivant décrit une conception de passerelle de couche application (ALG) possible pour une passerelle opérationnelle/multimédia:

EXEMPLE

La passerelle d'interconnexion du réseau opérationnel et du réseau multimédia (Figure 6) appartient formellement au réseau opérationnel. Cette passerelle n'est pas seulement nécessaire pour des raisons de sûreté et de système fermé. Elle sert également de point d'entrée aux applications du réseau multimédia vers les applications du réseau opérationnel et vice versa, offrant des services au réseau respectif et permettant d'échanger mutuellement des informations. Cela signifie également que la passerelle doit offrir deux interfaces différentes: l'interface avec la section opérationnelle basée sur les protocoles de communication utilisés dans la section opérationnelle et l'interface avec la section multimédia mettant en œuvre les protocoles de la section multimédia. À titre d'exemple, la passerelle peut recevoir l'information de répertoire de train en provenance de l'ECSP de la section opérationnelle ECSP dans un paquet de données de messagerie TRDP, puis offrir cette information aux participants de la section multimédia par l'intermédiaire d'un service Web. D'autres exemples de services d'interface consistent à fournir un flux de données (vidéo et audio) ou un échange direct de paquets de données TRDP. Il pourrait ainsi être nécessaire de disposer de deux types de services d'interface client/serveur, ainsi que de services d'interface de données temps réel. Du fait que les données temps réel exigent un haut débit, elles sont normalement routées sur la couche OSI 3 au lieu de la couche OSI 7. Afin de tenir compte des contraintes de sûreté, des mesures de sûreté supplémentaires doivent être présentes (telles que, par exemple, une fonction pare-feu).



IEC

Anglais	Français
OP/MM Gateway	Passerelle OP/MM
TRDP PD	PD TRDP
TRDP MD	MD TRDP
streaming	flux
operational network interface	interface de réseau opérationnel

Anglais	Français
Exchanged data	Données échangées
multimedia network interface	interface de réseau multimédia
SOAP	SOAP
streaming	flux
TRDP	TRDP

Figure 6 – Passerelle entre le réseau opérationnel et le réseau multimédia (exemple)

4.2.2.7 Aspects relatifs à la sécurité et à la sûreté

Une architecture comportant un réseau opérationnel et un réseau multimédia accouplés ne doit pas compromettre la catégorisation du système de transmission du réseau opérationnel (voir 4.2.2.4).

Au cas où le réseau multimédia est un système de transmission de catégorie 3 conforme à l'IEC 62280, au minimum un accouplement sur la couche OSI 3 avec des fonctions pare-feu doit être envisagé, mieux un accouplement sur la couche OSI 7.

La définition des capacités nécessaires du pare-feu ne relève pas du domaine d'application de la présente norme, celles-ci étant fortement relatives à la conception du système ainsi que de ses exigences de sûreté.

L'aptitude de l'accouplement à préserver la catégorisation du système de transmission du réseau opérationnel doit être démontrée dans le cadre du dossier de sécurité du système.

4.2.2.8 Rapprochement des réseaux

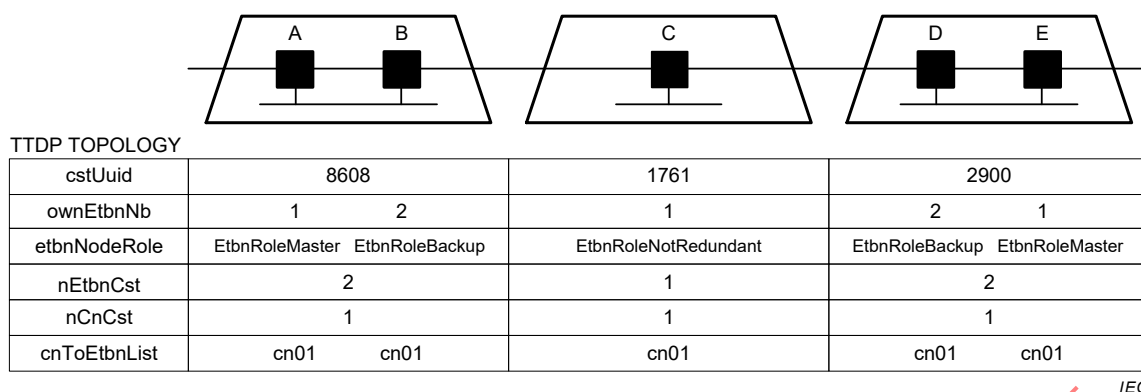
Pour les applications du réseau multimédia, il sera nécessaire d'avoir la même vision opérationnelle du train (voir 4.2.4.1) que pour une application du réseau opérationnel telle que la séquence des rames et la position du véhicule de tête, pour réaliser un schéma d'adressage sans ambiguïté au sein du train.

En règle générale, la vision opérationnelle du train déterminée par le réseau opérationnel et concrétisée au sein du répertoire opérationnel de train de la TTDB tel que défini en 5.3 doit faire autorité pour le réseau multimédia. Par conséquent, la séquence de rames et la position du véhicule de tête doivent être adoptées à partir du réseau opérationnel.

4.2.3 Ensembles de rames insécables

4.2.3.1 Généralités

Comme défini dans l'IEC 61375-1, un ensemble de rames insécables se compose de rames. La Figure 7 présente un exemple de trois rames accouplées, identifiées par les UUID 8608, 1761 et 2900. Chaque rame participe à l'inauguration ETB comme défini dans l'IEC 61375-2-5.



IEC

NOTE La table TTDP TOPOLOGY de la Figure 7 ne contient que les éléments pertinents.

Figure 7 – Exemple: trois rames accouplées

La rame identifiée par l'UUID 8608 comporte deux ETBN "A" et "B" aux positions relatives statiques 1 et 2 de la rame. Les ETBN "A" et "B" sont redondants. Les deux ETBN sont connectés au premier et unique réseau de rame.

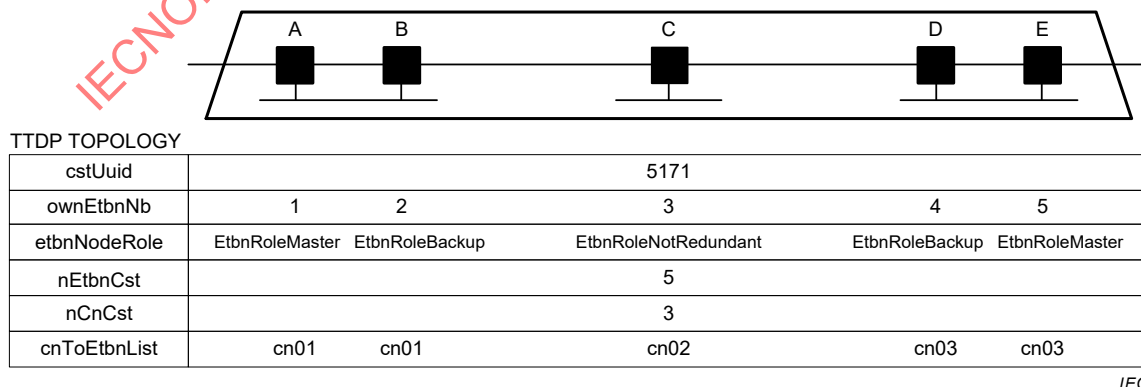
La rame identifiée par l'UUID 1761 comporte l'ETBN "C" à la position relative statique 1 de la rame. L'ETBN "C" n'est pas redondant. L'ETBN est connecté au premier et unique réseau de rame.

La rame identifiée par l'UUID 2900 comporte deux ETBN "D" et "E" aux positions relatives statiques 1 et 2 de la rame. Les ETBN "D" et "E" sont redondants. Les deux ETBN sont connectés au premier et unique réseau de rame.

4.2.3.2 Représentation d'un ensemble de rames insécables lors de l'inauguration ETB

Pour l'inauguration ETB selon l'IEC 61375-2-5, l'ensemble de rames insécables doit être représenté comme une rame unique.

Pour la composition d'un ensemble de rames insécables, les paramètres du contenu de la trame TTDP TOPOLOGY tel que défini dans l'IEC 61375-2-5 doivent être adaptés. La Figure 8 montre la vue externe de l'exemple d'un ensemble de rames insécables, ainsi que les éléments de la trame TTDP TOPOLOGY pertinents.



IEC

Figure 8 – Exemple: Ensemble de rames insécables

Dans l'exemple de la Figure 8, la rame identifiée par l'UUID 5171 comporte les cinq ETBN "A", "B", "C", "D" et "E" de la Figure 7 aux positions relatives statiques 1, 2, 3, 4 et 5 de la Rame, respectivement. Les ETBN "A"/"B" et "D"/"E" sont redondants. Le nombre total d'ETBN dans la Rame s'élève à cinq. Le nombre total de Réseaux de Rames dans la Rame s'élève à trois. Les ETBN "A" et "B" sont connectés au premier réseau de rame. L'ETBN "C" est connecté au deuxième Réseau de Rame. Les ETBN "D" et "E" sont connectés au troisième Réseau de Rame.

Lors de l'inauguration ETB, chaque ETBN de l'ensemble de rames insécables (à savoir la rame identifiée par l'UUID 5171) envoie sa trame TTDP TOPOLOGY. Par conséquent, une autre rame accouplée de l'extérieur connaît l'ensemble de la topologie réseau de l'ensemble de rames insécables.

4.2.3.3 Déficit d'information concernant la composition du train

Du fait que l'ensemble de rames insécables de la Figure 8 est représenté comme une rame unique, les informations concernant le mapping des ETBN et des réseaux de rames connectés sont perdues. Cette perte d'informations peut être rétablie lors du calcul de la TTDB.

4.2.4 Directions

4.2.4.1 Vues du train

Un train peut être vu de différents points de vue, qui dépendent du rôle de l'observateur.

Du point de vue du réseau central de train, un train se compose de réseaux de rames connectés à un ou plusieurs ETB par l'intermédiaire d'ETBN. Cette vue représente la topologie réseau du train. Une topologie de réseau central de train sera modifiée lorsque des ETBN sont connectés ou retirés. Cette vue est définie dans l'IEC 61375-2-5.

Du point de vue du train, un train se compose de rames, lesquelles se composent de véhicules. Une composition de train TCN sera modifiée lorsque des rames sont retirées ou ajoutées. Les directions au sein du train sont déterminées par ses extrémités définies.

Du point de vue opérationnel du train (vision du conducteur et du personnel du train), un train se compose d'une série de rames ainsi que d'une série de véhicules. Les directions au sein du train sont définies par la direction de vision principale de la cabine de conduite du véhicule de tête. Une vision opérationnelle de train sera modifiée si la cabine de tête est modifiée ou si des rames sont ajoutées ou retirées en l'absence de cabine de tête.

4.2.4.2 Directions de référence du train

Les directions TCN et l'orientation au sein d'un train, d'une rame et d'un véhicule sont spécifiées dans l'IEC 61375-1.

L'IEC 61375-1 spécifie également la numérotation des véhicules dans une rame et la numérotation des rames au sein d'un train. La numérotation est relative à l'Extrémité 1 d'un véhicule, d'une rame ou d'un train.

L'IEC 61375-1 laisse ouverte la manière dont les extrémités seront affectées. Les règles suivantes d'affectation des extrémités doivent s'appliquer:

- L'extrémité 1 d'un véhicule doit être prédéfinie statiquement par la conception de ce véhicule.
- L'extrémité 1 d'une rame doit être prédéfinie statiquement par la conception de cette rame.

- L'extrémité 1 d'un train est l'extrémité du train vers laquelle pointe la direction de référence 1 de l'ETB telle que définie dans l'IEC 61375-2-5.

4.2.4.3 Directions opérationnelles

Du point de vue de l'opérateur/du conducteur, une extrémité de train est la tête du train et l'extrémité opposée est la queue du train, déterminées par la direction opérationnelle du train. La direction opérationnelle du train pointe de la queue du train vers la tête du train. Les directions opérationnelles de train sont définies selon les règles a) à e).

- En cas de présence d'un véhicule de tête dans le train, la direction opérationnelle du train est déterminée par la direction de vision de la cabine de conduite du véhicule de tête.

En l'absence de véhicule de tête, la direction opérationnelle du train doit alors être déterminée par les règles suivantes:

- En cas de présence préalable d'un véhicule de tête, ce véhicule avec sa direction opérationnelle de train préalable détermine la direction opérationnelle du train.
- En cas de présence préalable de plusieurs véhicules de tête (par exemple après l'allongement d'un train), chacun d'entre eux ayant la même direction de vision, cette dernière détermine la direction opérationnelle du train.
- En cas de présence de rames multiples dans le train et d'une rame de traction à une extrémité, cette dernière définit la tête du train et la rame de traction comporte le véhicule 01 et détermine la direction opérationnelle du train.
- Dans tous les autres cas, la direction opérationnelle du train doit correspondre à la direction de référence ETB selon l'IEC 61375-2-5.

4.2.5 Propriétés de base des rames et des véhicules

4.2.5.1 Généralités

Les rames et les véhicules doivent être décrits par un ensemble de base de propriétés de rames et de véhicules, qui seront stockées dans la Base de données topologiques de train (TTDB). Pour cela, il convient de faire une distinction entre les propriétés statiques, topologiques et dynamiques:

- Les propriétés statiques sont des fonctions intégrées au véhicule ou à la rame et ne dépendent par conséquent que de sa classe ou de sa conception (ex: "identificateur de véhicule", "véhicule de première classe" ou "unité de traction à entraînement électrique").
- Les propriétés topologiques sont des propriétés qui présentent la configuration physique de la structure d'un train (ex: séquence des véhicules ou orientation des véhicules).
- Les propriétés dynamiques sont des propriétés qui définissent la mission d'un train (ex: "véhicule de tête", ou "direction de référence du train").

4.2.5.2 Informations relatives au type de véhicule et de rame

Les véhicules et les rames peuvent recevoir un classificateur de type permettant aux applications de contrôler plus facilement si les véhicules ou les rames ont des propriétés statiques communes. Ces classificateurs de type sont définis comme suit:

VehType	LABEL	-- Type of the vehicle Zero terminated string
CstType	LABEL	-- Type of the consist Zero terminated string

La définition des classificateurs de type est spécifique à l'application. Dans le cas de ce profil de communication, le classificateur de type est seulement informel.

4.2.5.3 Propriétés statiques

L'IEC 61375-2-4 donne une liste des propriétés statiques des rames et des véhicules.

4.2.5.4 Identification

Une rame doit être identifiée de manière unique par un UUID de rame tel que défini dans l'IEC 61375-2-5.

Un identificateur de véhicule défini par une application doit identifier un véhicule de manière unique.

EXEMPLE L'UIC a défini un numéro de wagon à 12 chiffres pour l'identification unique des voitures, par exemple le numéro "50 80 31 – 34 025 – 2".

4.3 Architecture Logique de Train (décomposition fonctionnelle)

4.3.1 Généralités

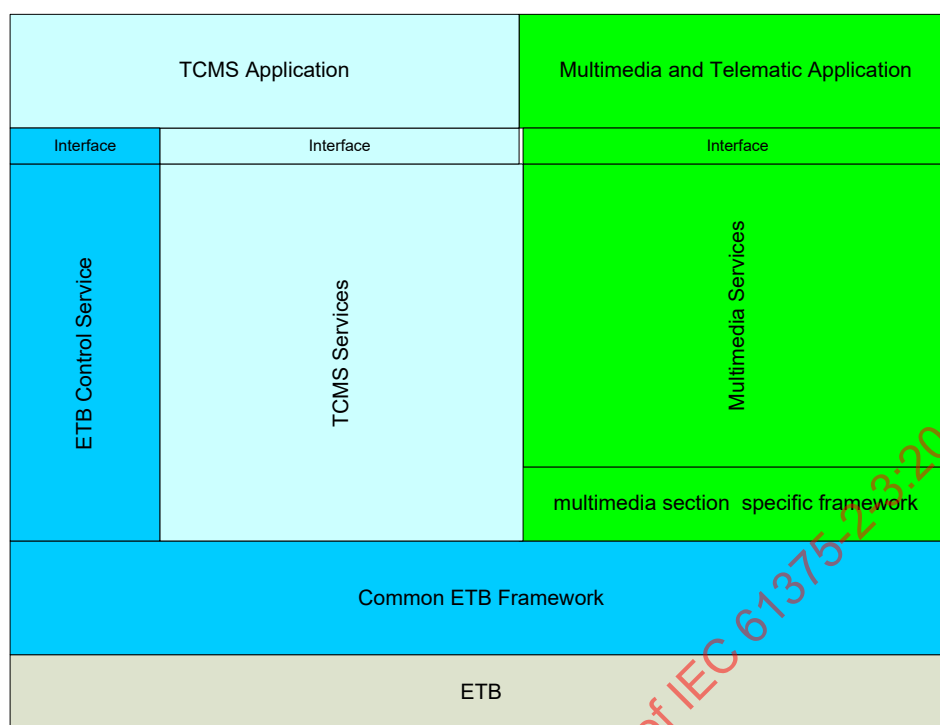
L'architecture logique de train définit un train comme étant composé de services et des informations échangées entre les services et entre utilisateur et services.

4.3.2 Classification des services

Les services peuvent être classés en (voir Figure 9).

- Services opérationnels, tels que définis dans la présente partie de l'IEC 61375 et dans l'IEC 61375-2-4;
- Services multimédias, tels que définis dans l'IEC 62580.

La présente partie de la norme concerne les services de contrôle ETB et les services de système ETB partagés. Les services opérationnels sont définis dans l'IEC 61375-2-4.



-  IEC 61375-2-5
-  IEC 61375-2-3
-  IEC 61375-2-4
-  IEC 62580

IEC

Anglais	Français
TCMS Application	Application TCMS
Multimedia and Telematic Application	Application multimédia et télématique
Interface	Interface
ETB Control Service	Service de contrôle ETB
TCMS Services	Services TCMS
Multimedia Services	Services multimédias
multimedia section specific framework	cadre spécifique à la section multimédia
Common ETB Framework	Cadre ETB commun
ETB	ETB

Figure 9 – Classification des services

Tous les services font appel à des fonctions, protocoles et définitions provenant d'un cadre ETB commun, défini à l'Article 5.

Pour les services multimédias, un cadre multimédia spécifique peut être ajouté; il fournit des fonctions, protocoles et définitions multimédias spécifiques. La définition du cadre multimédia spécifique ne relève pas du domaine d'application de la présente norme.

4.3.3 Présentation des services opérationnels

Au-delà des services TCMS tels que spécifiés dans l'IEC 61375-2-4, la présente partie de la norme définit à l'Article 6 un service de contrôle ETB qui fournit les fonctions de service énumérées dans le Tableau 2.

NOTE La décomposition fonctionnelle des services TCMS est relative à l'EN 15380-4.

Tableau 2 – Service de contrôle ETB

Fonction	Forme abrégée	Classification	Emplacement	Paragraphe concerné	Description
Gère l'information relative au véhicule de tête	Fonction véhicule de tête	contrôle	Rame	6.5	Définit ou réinitialise le véhicule de tête sur demande et gère les conflits entre les véhicules de tête
Confirme la configuration du train	Fonction confirmation/correction	contrôle	Rame	6.6	Permet à un utilisateur de confirmer ou corriger la composition de train indiquée
Gère le mode économie d'énergie Gère le mode de protection de la batterie	Fonction mode sommeil	contrôle	Rame	6.8	Met le train en mode faible puissance, en mode réveil sur commande ou sur activité de l'ETB

4.3.4 Fournisseur de service

Le service de contrôle ETB doit être fourni par l'ECSP (voir aussi 6.2).

Dans les rames comportant plusieurs réseaux de rames, les services de contrôle ETB doivent être fournis uniquement pour un réseau de rame. Ce sera le réseau de rame qui héberge également l'appareil terminal responsable du contrôle ETB (ECSC). La raison de cette restriction est d'éviter les conflits de contrôle au sein d'une rame.

En cas d'ECSP redondant, seul un ECSP fournit activement le service.

5 Cadre ETB commun

5.1 Généralités

5.1.1 Vue d'ensemble

Le cadre ETB commun définit les fonctions et les concepts de base qui forment ensemble l'infrastructure à laquelle les services sont intégrés. Il définit notamment:

- le télégramme CSTINFO (voir 5.2);
- la base de données topologiques de train TTDB (voir 5.3);
- la découverte de service (voir 5.4);
- le serveur DNS TCN (voir 5.5);
- l'échange de données (voir 5.6);
- la découverte de service (voir 5.6.4);
- le service d'infos train (voir 5.8).

5.1.2 Interopérabilité

Les télégrammes échangés sur l'ETB entre les rames et leurs structures de données associées sont versionnés au moyen d'une structure de données relatives aux informations de version, qui est définie comme suit:

```
VERSION ::= RECORD
{
    mainVersion      UINT8      -- main version, incremented for
                                incompatible changes

                                value range: 1..255 (0 = reserved)
    subVersion       UINT8      -- sub version, incremented for
                                compatible changes
                                value range: 0..255
}
```

Afin de garantir l'interopérabilité entre les rames, tous les changements affectant les articles normatifs de la présente norme doivent être traités comme des changements incompatibles, ce qui conduit à une incrémentation du paramètre 'mainVersion', sauf spécification contraire.

La présente norme ne définit aucune "négociation" des versions prises en charge lorsque deux rames sont accouplées en mettant en œuvre différentes versions des articles normatifs de la présente norme. Une telle négociation serait nécessaire si des changements compatibles étaient permis.

Des changements compatibles pourraient s'appliquer à des parties informelles, comme l'échange de données entre l'ECSC et l'ECSP défini à l'Annexe E.

5.2 Télégramme CSTINFO

5.2.1 Généralités

Tel que défini dans l'IEC 61375-2-5, suite à une inauguration de train ETB, au moins un appareil autorisé dans chaque rame doit émettre un télégramme CSTINFO sur l'ETB vers tous les autres appareils autorisés au sein du train pour les informer des propriétés de la rame.

Par souci de simplicité, l'ECSP doit être autorisé à émettre et recevoir les télégrammes CSTINFO.

5.2.2 Prise en charge des ensembles de rames insécables (option)

Pour l'inauguration ETB, un ensemble de rames insécables est représenté comme une rame unique (voir 4.2.3.2). Par conséquent, au moins un ECSP au sein de l'ensemble de rames insécables doit envoyer un télégramme CSTINFO qui décrit l'ensemble de rames insécables comme une rame (classe d'informations de rame 2, voir 5.2.4).

Afin de connaître la structure de rames d'un ensemble de rames insécables, au moins un ECSP au sein d'une rame insécable doit envoyer un télégramme CSTINFO (classe d'informations de rame 3, voir 5.2.4).

5.2.3 Protocole

Le protocole utilisé pour l'envoi des télégrammes CSTINFO est défini en 5.3.2.

5.2.4 Classes CSTINFO

Les classes CSTINFO permettent de distinguer les télégrammes CSTINFO envoyés par les rames, les ensembles de rames insécables, ainsi que les rames insécables. Trois classes sont définies:

- Classe CSTINFO 1: télégrammes CSTINFO envoyés par des rames qui ne sont pas des ensembles de rames insécables
- Classe CSTINFO 2: télégrammes CSTINFO envoyés par des ensembles de rames insécables
- Classe CSTINFO 3: télégrammes CSTINFO envoyés par des rames insécables

Les rames qui ne sont pas des ensembles de rames insécables doivent pouvoir recevoir et traiter les classes CSTINFO 1 et 2, mais peuvent ne pas pouvoir recevoir les télégrammes de la classe d'informations de rame 3.

Seuls les télégrammes de classe CSTINFO 2 doivent fournir une liste des rames insécables (groupement "cltrCstList" du télégramme CSTINFO). Pour les classes CSTINFO 1 et 3, ce groupement doit être omis (paramètre "cltrCstListCnt" réglé sur 0, voir 5.2.5).

5.2.5 Message de notification CSTINFO

Les messages CSTINFO doivent être envoyés dans la charge utile d'un message TRDP tel que défini en l'Article A.7.

La taille CSTINFO doit être limitée à la valeur de paramètre DatasetLength maximale d'une MD-PDU telle que définie en l'Article A.7.

Paramètres du télégramme de notification MD TRDP

MsgType:	'Mn'
Protocole:	UDP
ComId:	2
SourceURI (section utilisateur):	vide
DestinationURI (section utilisateur):	vide
DestinationURI (section hôte):	grpECSP.anyVeh.aCst.aCITrn.ITrn
SourceURI (section hôte):	-
Dataset:	CSTINFO

Les données MD TRDP UserDataset sont définies comme suit (voir Figure 10):

```

CSTINFO ::= RECORD
{
  cstInfo      CONSIST_INFO  -- consist information as defined in
                             subclause 5.3.3.2.6

```

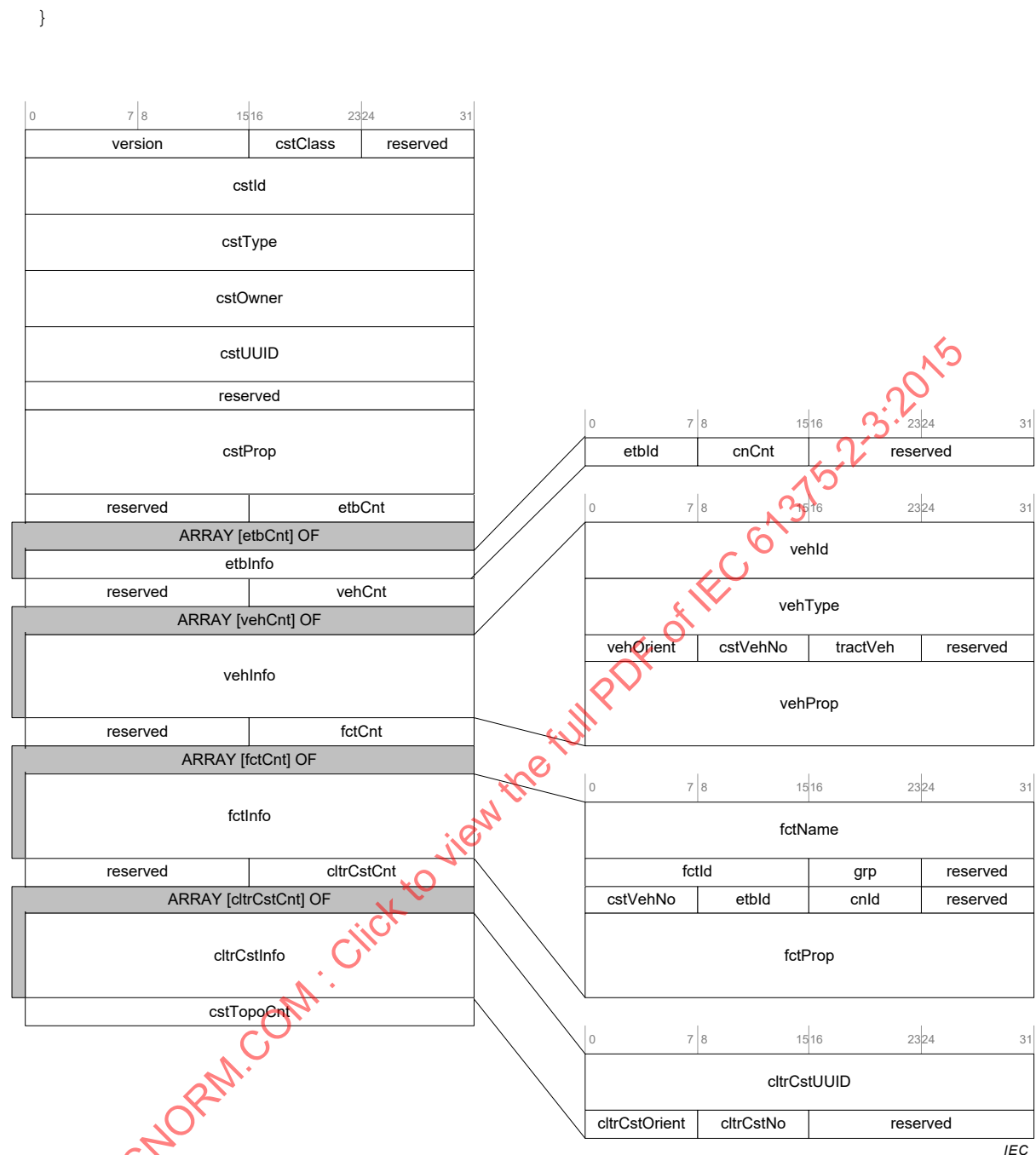


Figure 10 – Données de notification CSTINFO

5.2.6 Demande CSTINFO

L'envoi de télégrammes CSTINFO peut être commandé et demandé par un message CSTINFOCTRL tel qu'illustré à la Figure 11.

Le protocole utilisé pour l'envoi des messages de notification CSTINFOCTRL est défini en 5.3.2.

Les messages de notification CSTINFOCTRL doivent être envoyés dans la charge utile d'un message TRDP tel que défini en l'Article A.7.

Les messages de notification CSTINFOCTRL prennent en charge deux modes de commande (différentiés par le paramètre 'trnCstNo'):

trnCstNo > 0	Utilisé pour demander l'envoi de télégrammes CSTINFO. L'ECSP destinataire doit mémoriser la valeur de paramètre 'trnTopoCnt'. Ce mode doit être utilisé par l'ECSP responsable de l'échange CSTINFO au sein d'une rame (voir 5.3.2.3).
trnCstNo = 0	Utilisé pour demander l'envoi de télégrammes CSTINFO, mais sans suivre la valeur de paramètre 'trnTopoCnt'. Ce mode doit être utilisé par tous les appareils concernés par CSTINFO, à l'exception de l'ECSP responsable de l'échange CSTINFO.

Paramètres du télégramme de notification MD TRDP

MsgType:	'Mn'
Protocole:	UDP
ComId:	3
SMI:	3 (nécessaire uniquement si SDTv2 est utilisé)
UserDataVersion:	'0100'H (nécessaire uniquement si SDTv2 est utilisé)
SourceURI (section utilisateur):	vide
DestinationURI (section utilisateur):	vide
DestinationURI (section hôte):	grpECSP.anyVeh.aCst.aCITrn.ITrn
SourceURI (section hôte):	-
Dataset:	CSTINFOCTRL

Les données MD TRDP UserDataset sont définies comme suit (voir Figure 11):

```

CSTINFOCTRL ::= RECORD
{
    version      VERSION      -- version of the telegram
                                mainVersion = 1
                                subversion = 0

    trnCstNo     UINT8        -- train consist number
                                0 = without trnTopoCnt tracking
                                > 0 = with trnTopoCnt tracking

    cstCnt       UINT8        -- number of consists in list

    cstList      ARRAY[] OF CONSIST
                                -- consist list.
                                If trnCstNo > 0 this shall be an
                                ordered list starting with trnCstNo = 1
                                (exactly the same as in structure
                                TRAIN_DIRECTORY).
                                If trnCstNo = 0 it is not mandatory to
                                list all consists (only consists which

```

```

    should send CSTINFO telegram).
    parameters 'trnCstNo' and 'cstOrient'
    are optional and can be set to 0.

trnTopoCnt  UINT32      -- trnTopoCnt value
                        trnCstNo > 0: actual value
                        trnCstNo = 0: set to 0

safetyTrail ETBCTRL_VDP -- ETB-VDP trailer
                        parameter 'safeTopoCount' = 0
                        completely set to 0 = SDTv2 not used
}

```

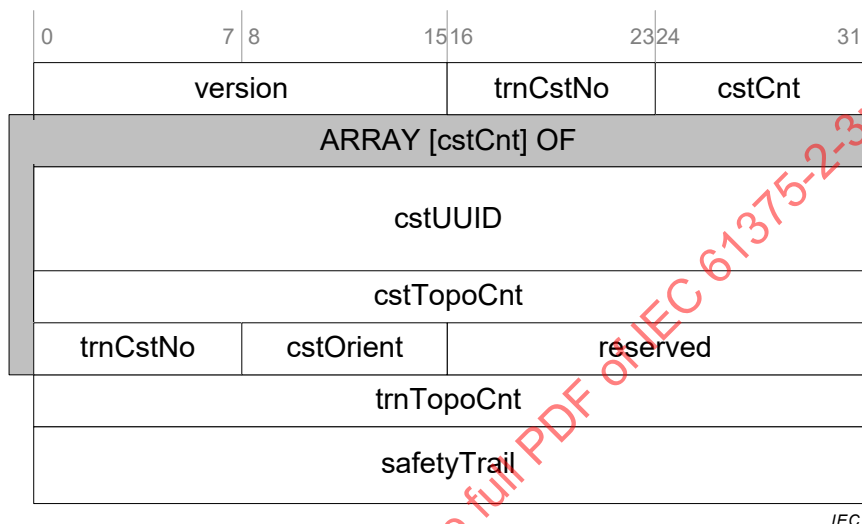


Figure 11 – Télégramme CSTINFOCTRL

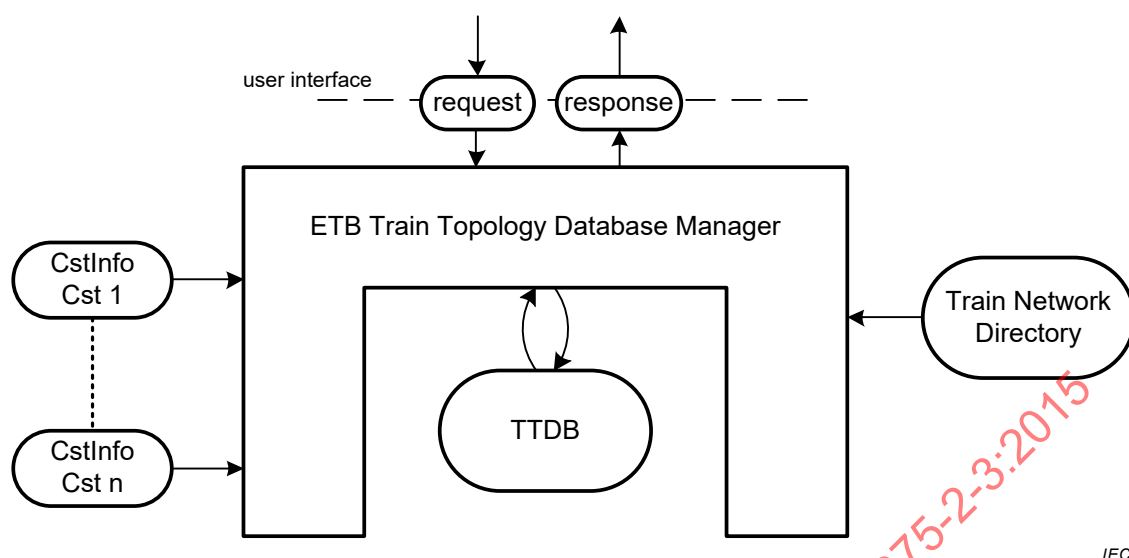
5.3 Base de données topologiques de train

5.3.1 Généralités

La base de données topologiques de train (TTDB) est une bibliothèque de données pour toutes les informations relatives à la composition réelle du train et à l'état réel de l'ETB.

Une instance de la TTDB doit être disponible au moins une fois dans chaque rame.

NOTE Dans les rames comportant plusieurs réseaux de rames, il pourrait être avantageux d'avoir plusieurs instances TTDB, une pour chaque réseau de rame.

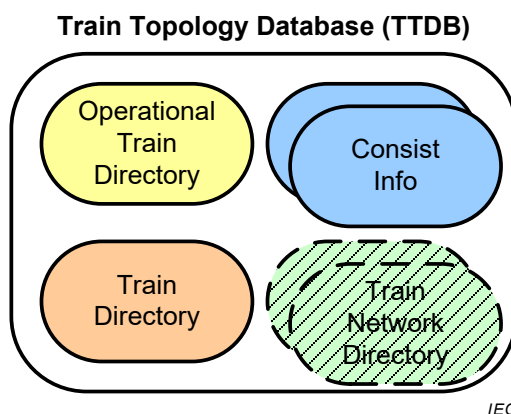


Anglais	Français
user interface	interface utilisateur
request	demande
response	réponse
CstInfo Cst 1	CstInfo Cst 1
CstInfo Cst n	CstInfo Cst n
ETB Train Topology Database Manager	Gestionnaire de base de données topologiques de train ETB
TTDB	TTDB
Train Network Directory	Répertoire du réseau de train

Figure 12 – Diagramme de blocs de gestion de la TTDB

La TTDB est maintenue par une fonction de gestionnaire de la TTDB (voir Figure 12) qui lit dans les informations de rame provenant des autres rames (fournies par les télégrammes CSTINFO) et du résultat de l'inauguration de train ETB train obtenu à partir de l'ETBN (répertoire de réseau de train tel que défini dans l'IEC 61375-2-5) et stocke ces informations dans la TTDB. Vis-à-vis des services d'application, elle offre une interface utilisateur permettant de récupérer tout ou partie des informations de la TTDB (voir Annexe E et Annexe G). Le gestionnaire TTDB doit systématiquement tenir à jour la TTDB.

La TTDB consiste en quatre blocs de données associés comme le montre la Figure 13.



Anglais	Français
Train Topology Database (TTDB)	Base de données topologiques de train (TTDB)
Operational Train Directory	Répertoire opérationnel de train
Consist Info	Informations de rame
Train Directory	Répertoire de train

Figure 13 – Contenu de la TTDB

Informations de Rame

Contient les descriptions de chaque rame telles qu'elles sont distribuées après l'inauguration de train ETB dans les télégrammes CSTINFO. Cela comporte également une liste des fonctions assurées par la rame et toutes les informations relatives à la rame dont un serveur DNS TCN a besoin pour la résolution des adresses URI TCN. Cette partie est préconfigurée statiquement pour chaque rame. Dans les trains comportant plusieurs ETB, les informations de rame sont disponibles pour chaque ETB et le contenu des différents blocs de données d'informations de rame peut différer d'un ETB à l'autre (voir 5.3.4).

Les ensembles de rames insécables sont représentés comme des rames dans le bloc de données d'informations de rame, et des informations de rames insécables peuvent être ajoutées. Ces entrées sont identifiables par le paramètre de classe d'informations de rame (voir 5.2.4).

Répertoire du Réseau de Train

Contient la description de la topologie réseau d'un ETB. Il s'agit du résultat d'une inauguration de train ETB telle que définie dans l'IEC 61375-2-5. Le répertoire de réseau de train est marqué par une signature unique appelée etbTopoCnt. La TTDB peut comporter jusqu'à quatre répertoires réseau représentant les quatre ETB maximaux du réseau de train.

Répertoire de Train

(Re)calculé chaque fois qu'il y a une modification dans la composition du train, indiquée par un changement de l'etbTopoCnt tel que défini dans l'IEC 61375-2-5. Le répertoire de train fournit une liste ordonnée de toutes les rames qui ont été découvertes sur l'ETB auquel le répertoire de train est associé.

Répertoire Opérationnel de Train

(Re)calculé chaque fois qu'il y a une modification dans le répertoire de train ou sur demande d'un utilisateur pour la fonction véhicule de tête (voir 6.5) ou pour une correction (voir 6.6). Ce bloc de données ajoute les informations de tête et de correction dans la TTDB et représente la vision opérationnelle du train telle que définie en 4.2.4.1. Dans les trains comportant plusieurs ETB, seul un répertoire opérationnel de train pour l'ensemble des ETB sera généré.

Chacun de ces blocs de données est protégé par une signature, qui permet de contrôler l'intégrité des données stockées (voir 5.3.3 pour une définition détaillée de ces blocs de données).

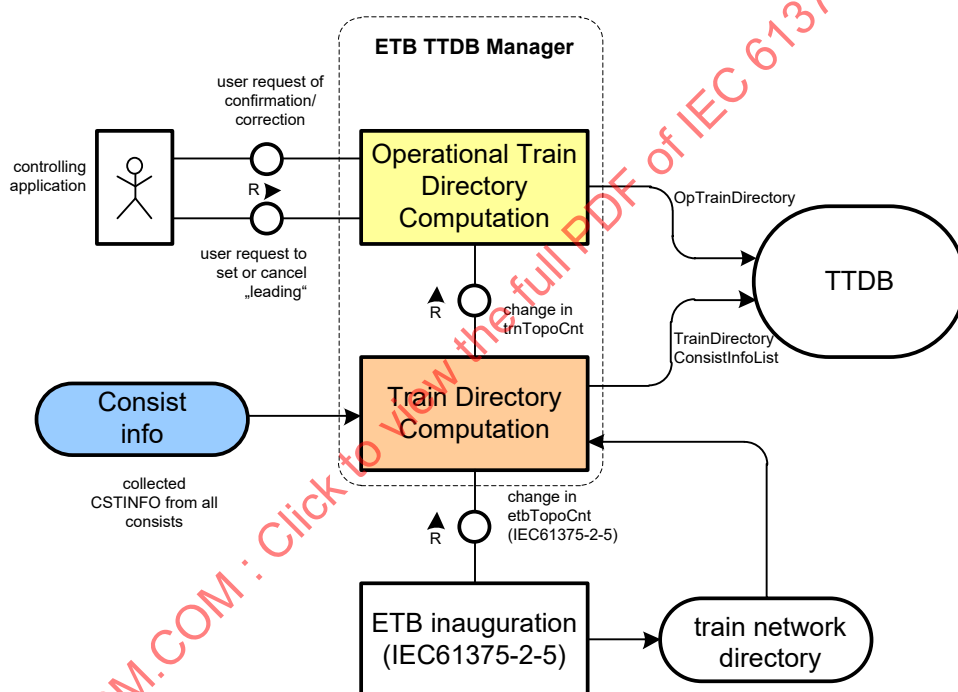
5.3.2 Calcul de la TTDB

5.3.2.1 Généralités

La TTDB est calculée en plusieurs étapes et par différentes fonctions (voir Figure 14):

- Inauguration ETB telle que définie dans l'IEC 61375-2-5
- Calcul du répertoire de train sur la base du répertoire de réseau de train et des informations de rame
- Calcul du répertoire opérationnel de train sur la base des informations de tête et de correction

Les différentes fonctions sont définies plus en détail dans les alinéas qui suivent.



IEC

Anglais	Français
ETB TTDB Manager	Gestionnaire TTDB ETB
user request of confirmation/correction	demande utilisateur de confirmation/correction
controlling application	application de contrôle
user request to set or cancel "leading"	demande utilisateur de définition ou d'annulation du "véhicule de tête"
Operational Train Directory Computation	Calcul du Répertoire Opérationnel de Train
change in TrnTopoCnt	modification de TrnTopoCnt
Consist info	Informations de rame
Train Directory Computation	Calcul du Répertoire de Train
collected CSTINFO from all consists	CSTINFO collectées depuis toutes les rames
change in etbTopoCnt (IEC61375-2-5)	modification de etbTopoCnt (IEC 61375-2-5)
ETB inauguration (IEC61375-2-5)	Inauguration ETB (IEC 61375-2-5)
train network directory	répertoire du réseau de train

Figure 14 – Diagramme de blocs du calcul de la TTDB

5.3.2.2 Inauguration de train ETB

La TTDB se base sur le répertoire de réseau de train calculé conformément à l'algorithme défini à l'IEC 61375-2-5 et représentant la topologie de l'ETB en tant que séquence de l'ETBN appartenant aux rames. Une topologie réseau ETB découverte sera identifiée de manière unique par son compteur de topographie (etbTopoCnt).

5.3.2.3 Calcul du répertoire de train

Un (nouveau) calcul du répertoire de train doit être effectué chaque fois qu'il y a modification du compteur etbTopoCnt.

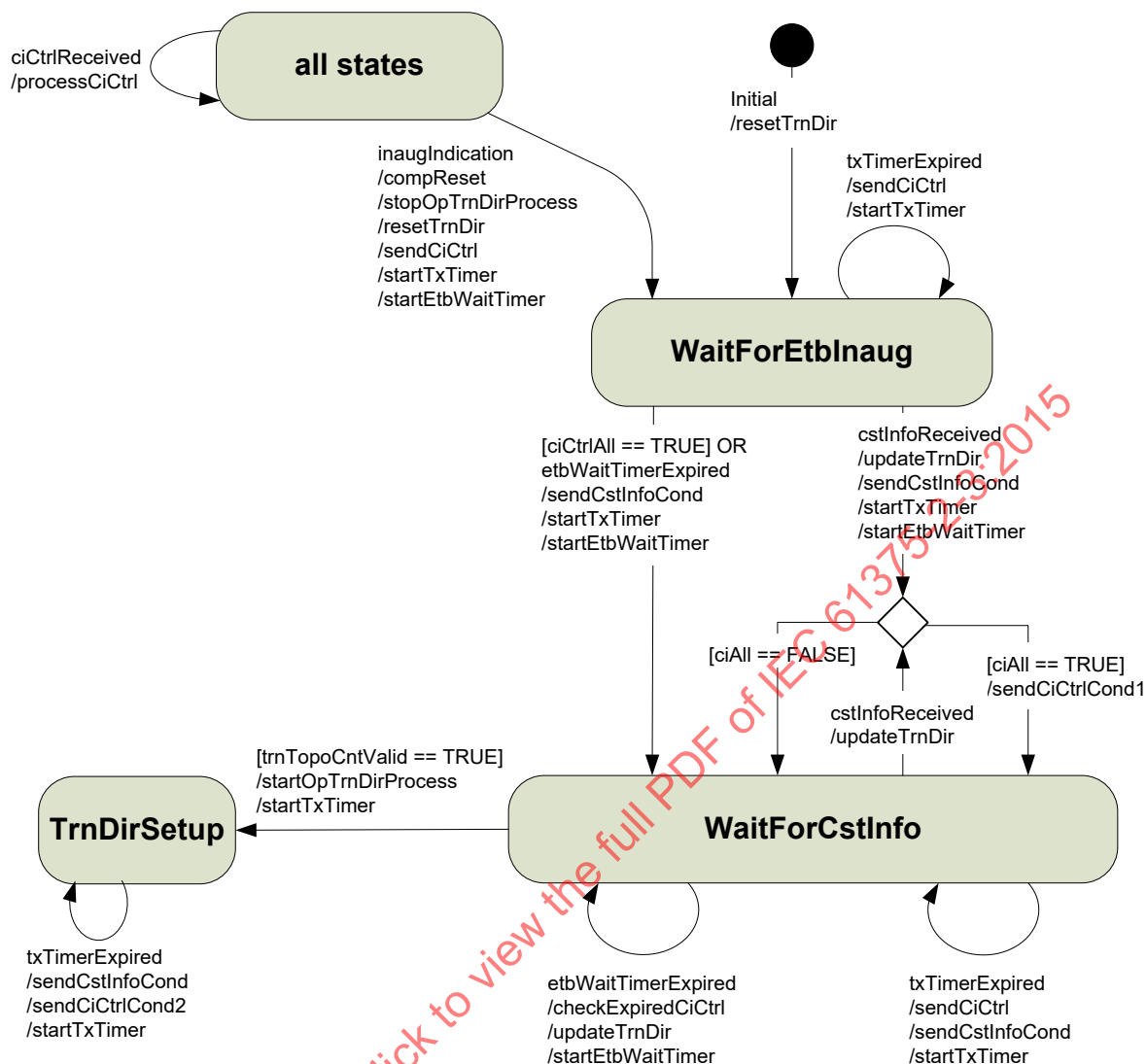
Pendant le calcul du répertoire de train, la TTDB doit être enrichie par les informations de rame recueillies en provenance des rames du train (télégrammes CSTINFO).

Le processus de calcul du répertoire de train est défini par le diagramme d'états présenté à la Figure 15 et décrit dans le Tableau 3 jusqu'au Tableau 5.

NOTE 1 Un nouveau calcul du répertoire de train sera évité si les inaugurations de train sont inhibées comme défini dans l'IEC 61375-2-5.

Pour l'optimisation de l'échange de messages CSTINFO et CSTINFOCTRL (réduction de la charge réseau), un ensemble de balises contrôlant l'envoi des messages CSTINFO et CSTINFOCTRL doit être pris en charge, défini comme suit:

ciTxFlag	BOOLEAN	-- CSTINFO send control flag FALSE: CSTINFO not to send TRUE: CSTINFO to send
cicTxCnt1	UINT8	-- CSTINFOCTRL send control flag (counter) value range: 0..1
cicTxCnt2	UINT8	-- CSTINFOCTRL send control flag (counter) value range: 0..2



IEC

Figure 15 – Diagramme d'états de calcul du répertoire de train

NOTE 2 Pour certaines transitions du diagramme d'états UML représenté à la Figure 15, un trigger est la réception d'un paquet CSTINFOCTRL. Pour ces cas uniquement, les guards sont indiqués dans les transitions du diagramme d'états UML.

Tableau 3 – Calcul du répertoire de train – triggers

Trigger	Description
Initial	Initialise ou relance l'ECSP
inaugIndication	Indique une nouvelle inauguration de train fourni par l'ETBN local conformément à l'IEC 61375-2-5.
ciCtrlReceived	Un télégramme CSTINFOCTRL a été reçu.
cstInfoReceived	Un télégramme CSTINFO a été reçu.
etbWaitTimerExpired	Le minuteur d'attente CSTINFOCTRL a expiré – indiquant qu'une rame répertoriée dans le répertoire de train n'envoie pas de paquets CSTINFOCTRL, autrement dit l'ECSP de cette rame ne fonctionne pas correctement.
txTimerExpired	Le minuteur pour la transmission périodique des télégrammes CSTINFO et CSTINFOCTRL a expiré. L'expiration de ce minuteur indique que le télégramme CSTINFO ou CSTINFOCTRL transmis précédemment n'a pas été reçu par tous les ECSP.

Tableau 4 – Calcul du répertoire de train – guards

Guard	Description
[trnTopoCntValid == TRUE]	Un message CSTINFOCTRL a été reçu en provenance de toutes les rames répertoriées dans le répertoire de train avec un paramètre 'cstTopoCnt' != 0 et toutes les rames à partir desquelles le CSTINFOCTRL a été reçu ont une valeur trnTopoCnt identique. NOTE 1 La valeur trnTopoCnt est connue à partir des télégrammes CSTINFOCTRL reçus avec un paramètre trnCstNo > 0. NOTE 2 Cela signifie que tous les ECSP actifs partagent les mêmes données d'entrée pour le calcul du Répertoire Opérationnel de Train.
[CiAll == TRUE]	Un message CSTINFO a été reçu en provenance de toutes les rames répertoriées dans le répertoire de train.
[CiAll == FALSE]	Un message CSTINFO n'a pas (encore) été reçu en provenance de toutes les rames répertoriées dans le répertoire de train.
[CiCtrlAll == TRUE]	Un message CSTINFOCTRL a été reçu en provenance de toutes les rames répertoriées dans le répertoire de train.
[CiCtrlAll == FALSE]	Un message CSTINFOCTRL n'a pas (encore) été reçu en provenance de toutes les rames répertoriées dans le répertoire de train.

Tableau 5 – Calcul du répertoire de train – actions

Action	Description
startEtbWaitTimer	(Re)lancer le minuteur en une prise avec une valeur temporelle de $(5,0 \pm 0,2)$ s. Pendant ce temps, il convient que les télégrammes CSTINFOCTRL provenant de toutes les rames répertoriées dans le Répertoire de Train soient reçus.
startTxTimer	(Re)lancer le minuteur en une prise avec une valeur temporelle de $(1,0 \pm 0,1)$ s. Ce minuteur gère la retransmission périodique des télégrammes CSTINFOCTRL et CSTINFO en cas de défaillance réseau. NOTE Le message CSTINFO est retransmis uniquement si cela est exigé par un télégramme CSTINFOCTRL.

Action	Description
resetTrnDir	Réinitialiser le répertoire de train: - le répertoire de train comporte l'ensemble des rames identifiées par l'inauguration ETB (répertoriées dans le Répertoire de Réseau de Train). - Les valeurs cstTopoCnt de toutes les structures CONSIST sont définies sur 0, sauf pour la rame considérée. - calculer la valeur trnTopoCnt Effacer la liste d'informations de rame, à l'exception de l'entrée correspondant à la rame considérée Redéfinir l'envoi des balises de contrôle: ciTxFlag = FALSE cicTxCnt1 = 1 cicTxCnt2 = 0
updateTrnDir	Stocker CSTINFO dans ConsistInfoList. Mettre à jour le répertoire de train: - Mettre à jour les structures CONSIST du répertoire de train avec la valeur cstTopoCnt des télégrammes CSTINFO reçus - Calculer la nouvelle valeur de trnTopoCnt.
startOpTrnDirProcess	Lancer le traitement du répertoire opérationnel de train en déclenchant l'événement 'startOpTrnDirProcess' dans le diagramme d'états de traitement ETBCTRL (voir 6.4.5)
stopOpTrnDirProcess	Arrêter le traitement du répertoire opérationnel de train en déclenchant l'événement 'stopOpTrnDirProcess' dans le diagramme d'états de traitement ETBCTRL (voir 6.4.5)
sendCstInfoCond	Envoi conditionnel du télégramme CSTINFO: IF (ciTxFlag == TRUE) THEN send CSTINFO telegram; ciTxFlag = FALSE; END; NOTE L'envoi conditionnel du télégramme CSTINFO a pour objectif de réduire la charge réseau.
sendCiCtrl	Envoyer le télégramme CSTINFOCTRL
sendCiCtrlCond1	Envoi conditionnel du télégramme CSTINFOCTRL: IF (cicTxCnt1 > 0) THEN send CSTINFOCTRL telegram; cicTxCnt1 = cicTxCnt1 – 1; END; NOTE L'envoi conditionnel du télégramme CSTINFOCTRL a pour objectif de réduire la charge réseau.
sendCiCtrlCond2	Envoi conditionnel du télégramme CSTINFOCTRL: IF (cicTxCnt2 > 0) THEN send CSTINFOCTRL telegram; cicTxCnt2 = cicTxCnt2 – 1; END; NOTE L'envoi conditionnel du télégramme CSTINFOCTRL a pour objectif de réduire la charge réseau.
processCiCtrl	Traiter le télégramme CSTINFOCTRL reçu. IF (CSTINFOCTRL parameter 'trnCstNo' > 0) THEN memorize the CSTINFOCTRL parameter 'trnTopoCnt'; memorize the time when CSTINFOCTRL has been

Action	Description
	<pre> received; END; IF ((CSTINFOCTRL parameter 'trnCstNo' > 0) AND (state == TrainDirSetup) AND (CSTINFOCTRL parameter 'trnTopoCnt != trnTopoCnt value in local train directory)) THEN cicTxCnt2 = 2; END; IF (CSTINFOCTRL message does contain an entry of parameter 'cstTopoCnt' with a value == 0 for the local consist, which is the consist the receiving ECSP belongs to) THEN ciTxFlag = TRUE; END; NOTE Cela signifie que l'ECSP émetteur n'a pas reçu le message CSTINFO de la rame locale et qu'il demande désormais qu'il soit (ré)envoyé.</pre>
checkExpiredCiCtr	Vérifier la présence éventuelle de messages CSTINFOCTRL expirés. Pour chaque rame à partir de laquelle le dernier message CSTINFOCTRL reçu est antérieur à la valeur etbWaitTimer: • Supprimer le message CSTINFOCTRL, ainsi que la valeur trnTopoCnt mémorisée. • Supprimer les informations de rame de la liste d'informations de rame. • Définir le paramètre de rame 'cstTopoCnt' du répertoire de train sur 0.
compReset	Réinitialiser la composition en déclenchant l'événement 'compReset' dans le diagramme d'états de la fonction véhicule de tête (voir 6.5.3.2) et le diagramme d'états de la fonction confirmation/correction (voir 6.6.4).

NOTE Le processus de calcul du répertoire de train est déclenché par l'ETBN après l'inauguration du train. Ensuite, les ECSP échangent des télégrammes CSTINFOCTRL et CSTINFO pour le calcul du répertoire de train. L'ETBN et l'ECSP ont besoin d'être synchronisés de manière à ce que l'ECSP puisse lancer le processus de calcul du répertoire de train immédiatement après l'indication de l'inauguration du train. Si la différence de temps est trop importante (ex: supérieure à etbWaitTime), cette rame n'apparaîtra pas dans le répertoire de train avec une valeur cstTopoCnt != 0 et n'apparaîtra donc pas non plus dans le répertoire opérationnel de train.

5.3.2.4 Calcul du répertoire opérationnel de train

Un calcul du répertoire opérationnel de train doit être effectué sur demande d'un utilisateur et chaque fois qu'il y a une modification du répertoire de train associé au réseau opérationnel (ETB0).

Si l'utilisateur demande d'activer la fonction véhicule de tête, le véhicule tête doit être déterminé et indiqué dans la base de données. Ce point est détaillé en 6.5.

Si l'utilisateur demande d'annuler la fonction véhicule de tête, l'indication du véhicule tête doit être supprimée de la base de données. Ce point est détaillé en 6.5.

Tant qu'il n'y a pas de demande de confirmation/correction de la part d'un utilisateur, la TTDB doit être à l'état "non confirmé".

Si l'utilisateur demande de corriger la TTDB, le répertoire opérationnel de train doit être mis à jour et la TTDB doit passer à l'état "confirmé" en l'absence de conflit ou d'erreur; sinon elle doit rester à l'état "non confirmé". Ce point est détaillé en 6.6.

Si la TTDB est à l'état "confirmé" et que le répertoire de train change, il doit être vérifié que le nouveau répertoire de train est en désaccord avec la composition de train confirmée et/ou corrigée. En cas de désaccord, la TTDB doit passer à l'état "non confirmé". Ce point est détaillé en 6.6.

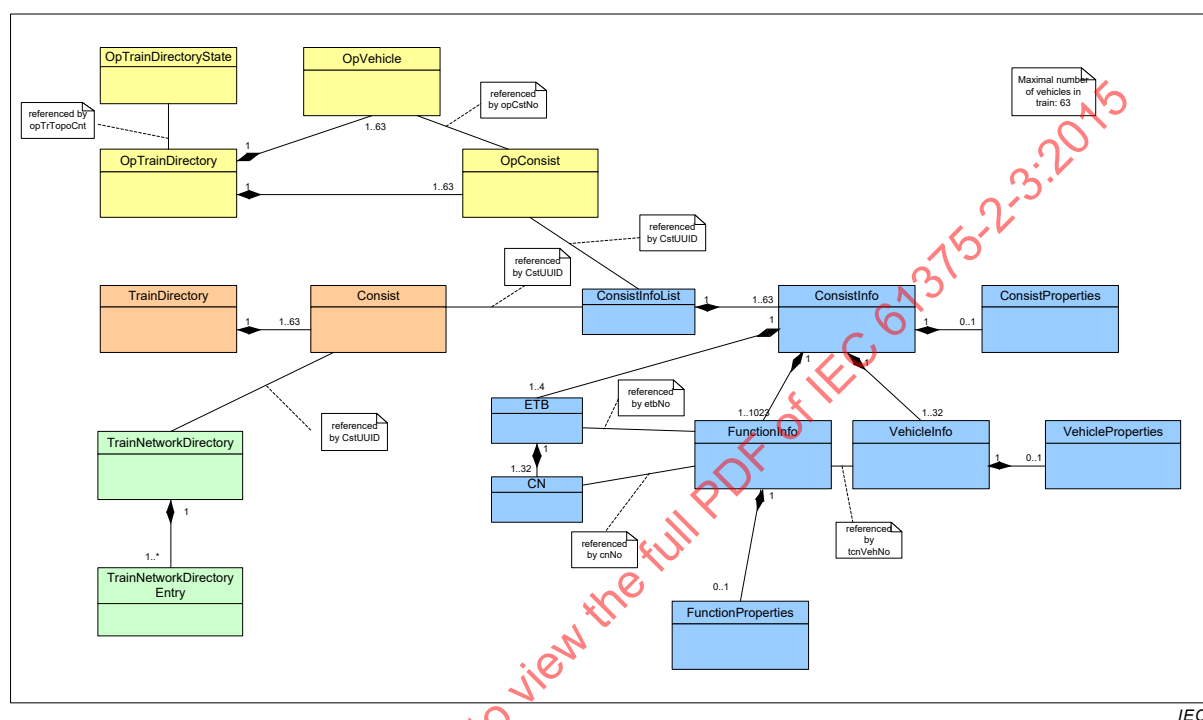
NOTE Cela se produira par exemple après l'accouplement de deux trains.

Le calcul du répertoire opérationnel de train est spécifié en détail en 6.7.

5.3.3 Structure des données

5.3.3.1 Modèle de données

La structure détaillée de la TTDB peut être modélisée comme illustrée à la Figure 16.



IEC

Anglais	Français
referenced by opTfTopoCnt	référéncé par opTfTopoCnt
referenced by CstUUID	référéncé par CstUUID
referenced by opCstNo	référéncé par opCstNo
referenced by etbNo	référéncé par etbNo
referenced by ecnNo	référéncé par ecnNo
referenced by tcnVehNo	référéncé par tcnVehNo
Maximal number of vehicles in train: 63	Nombre maximal de véhicules dans le train: 63

Figure 16 – Diagramme de classes TTDB (exemple)

Les quatre parties de la TTDB répertoriées en 5.3.1 sont séparées par des associations d'agréations.

NOTE Une interface applicative pour la récupération des données de la TTDB est définie à l'Annexe E.

5.3.3.2 Structures de données

5.3.3.2.1 Structure ETB_INFO – Informations ETB

La structure de données ETB_INFO est définie comme suit:

```
ETB_INFO ::= RECORD
{
```

```

etbId      UINT8      -- identification of the ETB
                        0 = ETB0 (operational network)
                        1 = ETB1 (multimedia network)
                        2 = ETB2 (other network)
                        3 = ETB3 (other network)

cnCnt      UINT8      -- number of CN within
                        consist connected to this ETB,
                        value range 1..16,
                        referring to cnId 0...15 acc. IEC 61375-2-5

reserved01  UINT16     -- reserved for future use (= 0)
}

```

5.3.3.2.2 Structure CLTR_CST_INFO – Informations sur la composition d'un ensemble de rames insécables

La structure de données CLTR_CST_INFO est définie comme suit:

```

CLTR_CST_INFO ::= RECORD
{
  cltrCstUUID      UINT8[16] -- UUID of the consist

  cltrCstOrient    UINT8      -- consist orientation
                        '01'B = same as closed train direction
                        '10'B = inverse to closed train direction

  cltrCstNo        UINT8      -- sequence number of the consist within the
                        closed train, value range 1..32

  reserved01       UINT16     -- for future use (= 0)
}

```

5.3.3.2.3 Structure PROPERTIES – Propriétés définies par l'application

La structure de données PROPERTIES est définie comme suit:

```

PROPERTIES ::= RECORD
{
  version          VERSION     -- properties version information,
                                application defined

  len              UINT16      -- properties length in number of octets,
                                application defined, shall be a multiple
                                of 4 octets for alignment reasons
                                value range: 0..32768

  prop             UINT8[]     -- properties, application defined
}

```

5.3.3.2.4 Structure FUNCTION_INFO – Attributs de fonctions

Ce dataset décrit les fonctions qui sont définies pour une rame.

La structure de données FUNCTION_INFO est définie comme suit:

```

FUNCTION_INFO ::= RECORD
{
  fctName          LABEL       -- function device or group label as defined
                                in 5.4.4.6.2

  fctId            UINT16      -- host identification of the function
                                device or group as defined in
                                IEC 61375-2-5, application defined.
                                Value range:

```

```

Devices = 1..16382
consist-limited/ETB-related group
= 1..254
ETB-related group = 256..16382
All-train group = 256..32767
grp          BOOLEAN8  -- is a function group (will be resolved
                        to an IP multicast address)
                        0 = no
                        1 = yes
reserved01   UINT8     -- reserved for future use (= 0)
cstVehNo     UINT8     -- sequence number of the vehicle within the
                        consist the function belongs to
                        value range: 1..32
                        0 = not defined
etbId        UINT8     -- identification of the related ETB
                        0 = ETB0 (operational network)
                        1 = ETB1 (multimedia network)
                        2 = ETB2 (other network)
                        3 = ETB3 (other network)
                        255 = related to all ETB (all-train group)
cnId         UINT8     -- identification of connected consist network
                        in the consist, related to the etbId
                        value range: 0..32
                        0 = not relevant (e.g. for groups)
reserved02   UINT8     -- reserved for future use (= 0)
}

```

La combinaison des paramètres 'fctName', 'fctId', 'cnId' et 'etbId' doit être unique.

La présente partie de l'IEC 61375 ne définit que la portion des informations de fonction qui est nécessaire à l'identification et l'adressage de la fonction. Les informations détaillées sur la fonction proprement dite et la manière de récupérer ces informations via le réseau doivent être définies au sein du profil d'application associé.

NOTE Le profil d'application peut définir un message TRDP ainsi que les primitives de service associées pour récupérer les informations de fonction. Une bonne pratique consiste à définir une méthode normalisée, commune à l'ensemble des services.

5.3.3.2.5 Structure VEHICLE_INFO – Attributs statiques des véhicules

La structure de données VEHICLE_INFO est définie comme suit:

```

VEHICLE_INFO ::= RECORD
{
    vehId      LABEL      -- vehicle identifier label, application defined
                        (e.g. UIC vehicle identification number)
                        vehId of vehNo = 1 can be used also as
                        consist identifier
    vehType    LABEL      -- vehicle type, application defined
    vehOrient  UINT8      -- vehicle orientation
                        '01'B = same as consist direction
                        '10'B = inverse to consist direction
    cstVehNo   UINT8      -- sequence number of vehicle within the
                        consist (1..32)
    tractVeh   ANTIVALENT8 -- vehicle is a traction vehicle
                        '01'B = vehicle is not a traction vehicle
}

```

```

'10'B = vehicle is a traction vehicle
reserved01  UINT8      -- for future use (= 0)
vehProp     PROPERTIES -- static vehicle properties
}

```

5.3.3.2.6 Structure CONSIST_INFO – Attributs de rames statiques

Ce dataset est complété à l'aide des informations, qui ont été reçues avec le télégramme CSTINFO correspondant.

La structure de données CONSIST_INFO est définie comme suit:

```

CONSIST_INFO ::= RECORD
{
    version      VERSION      -- CONSIST_INFO data structure version
                                mainVersion = 1
                                subVersion = 0
    cstClass     UINT8         -- consist info classification
                                1 = (single) consist
                                2 = closed train
                                3 = closed train consist
    reserved01   UINT8         -- reserved for future use (= 0)
    cstId        LABEL         -- consist identifier, application defined
                                (e.g. UIC identifier)
    cstType      LABEL         -- consist type, application defined
    cstOwner     LABEL         -- consist owner, application defined
                                (e.g. "trenitalia.it")
    cstUUID      UINT8[16]     -- UUID of the consist
    reserved02   UINT32        -- reserved for future use (= 0)
    cstProp      PROPERTIES    -- static consist properties
    reserved03   UINT16        -- reserved for future use (= 0)
    etbCnt       UINT16        -- number of ETB's, range: 1..4
    etbInfoList  ARRAY [] OF ETB_INFO
                                -- ETB information for ETBs in the consist
                                Ordered list starting with lowest etbId
    reserved04   UINT16        -- reserved for future use (= 0)
    vehCnt       UINT16        -- number of vehicles in consist
                                value range: 1..32
    vehInfoList  ARRAY [] OF VEHICLE_INFO
                                -- vehicle info for the vehicles in the consist
                                Ordered list starting with cstVehNo = 1
    reserved05   UINT16        -- reserved for future use (= 0)
    fctCnt       UINT16        -- number of consist functions
                                value range: 0..1024
    fctInfoList  ARRAY [] OF FUNCTION_INFO
                                -- function info for the functions in consist
                                lexicographical ordered by fctName
    reserved06   UINT16        -- reserved for future use (= 0)
    cltrCstCnt   UINT16        -- number of closed train consists
                                value range: 0..32
                                0 = consist is no closed train
    cltrCstInfoList ARRAY [] OF CLTR_CST_INFO
                                -- info on closed train composition
                                Ordered list starting with cltrCstNo = 1
    cstTopoCnt   UINT32        -- consist topography counter
}

```



```

trnCstNo      UINT8      -- Sequence number of consist
                        in train (1..63)
cstOrient      UINT8      -- consist orientation
                        '01'B = same as train reference direction
                        '10'B = inverse to train reference
                        Direction
reserved01     UINT16     -- for future use (= 0)
}

```

5.3.3.2.9 Structure TRAIN_DIRECTORY

Ce dataset fournit les informations relatives à la composition de train découverte durant l'inauguration ETB.

La structure de données TRAIN_DIRECTORY est définie comme suit:

```

TRAIN_DIRECTORY ::= RECORD
{
    version      VERSION    -- train directory data structure version
                        mainVersion = 1
                        subVersion = 0
    etbId        UINT8      -- identification of the ETB the train
                        directory is computed for
                        0 = ETB0 (operational network)
                        1 = ETB1 (multimedia network)
                        2 = ETB2 (other network)
                        3 = ETB3 (other network)
    cstCnt       UINT8      -- number of consists in train
                        range: 1..63
    cstList      ARRAY [] OF CONSIST
                        -- dynamic consist list
                        ordered list starting with trnCstNo = 1
    trnTopoCnt   UINT32     -- Train topography counter
                        computed as defined in 5.3.3.2.16
                        from the beginning of the record until the
                        last element before the topography counter
                        (seed value: etbTopoCnt)
}

```

5.3.3.2.10 Structure OP_VEHICLE – Attributs de véhicules opérationnels

La structure de données OP_VEHICLE est définie comme suit:

```

OP_VEHICLE ::= RECORD
{
    vehId        LABEL      -- unique vehicle identifier, application
                        defined (e.g. UIC Identifier)
    opVehNo      UINT8      -- operational vehicle sequence number in train
                        value range: 1..63
    isLead       ANTIVALENT8 -- vehicle is leading
    leadDir      UINT8      -- vehicle leading direction
                        0 = not relevant
                        1 = leading direction 1
                        2 = leading direction 2
    trnVehNo     UINT8      -- vehicle sequence number within the train
}

```

with vehicle 01 being the first vehicle
in ETB reference direction 1 as defined in
IEC 61375-2-5
value range: 1..63
a value of 0 indicates that this vehicle
has been inserted by correction

```

vehOrient    UINT8    -- vehicle orientation
                    '0'B = not known (corrected vehicle)
                    '01'B = same as operational train direction
                    '10'B = inverse to operational train
                    direction
ownOpCstNo    UINT8    -- operational consist number the vehicle
                    belongs to
reserved01    UINT8    -- reserved for future use (= 0)
reserved02    UINT8    -- reserved for future use (= 0)
}

```

5.3.3.2.11 Structure OP_CONSIST – Attributs de rames opérationnels

La structure de données OP_CONSIST est définie comme suit:

```

OP_CONSIST ::= RECORD
{
    cstUUID      UINT8[16]    -- Reference to static consist attributes
                                0 = not available (e.g. correction)
    opCstNo      UINT8        -- operational consist number in train (1..63)
    opCstOrient  UINT8        -- consist orientation
                                '0'B = not known (corrected vehicle)
                                '01'B = same as operational train direction
                                '10'B = inverse to operational train
                                direction
    trnCstNo     UINT8        -- Sequence number of consist
                                in train with consist 01 being the first
                                consist in ETB reference direction 1
                                as defined in IEC 61375-2-5
                                value range 0..63
                                a value of 0 indicates that this consist
                                has been inserted by correction
    reserved01   UINT8        -- reserved for future use (= 0)
}

```

5.3.3.2.12 Structure OP_TRAIN_DIRECTORY_STATE

La structure de données OP_TRAIN_DIRECTORY_STATE est définie comme suit:

```

OP_TRAIN_DIRECTORY_STATE ::= RECORD
{
    version      VERSION      -- TRAIN_DIRECTORY_STATE data structure version
                                mainVersion = 1
                                subVersion = 0
    reserved01   UINT8        -- reserved for future use (= 0)
    reserved02   UINT8        -- reserved for future use (= 0)
    etbId        UINT8        -- identification of the ETB the TTDB is
                                computed for
}

```

```

                                0 = ETB0 (operational network)
                                1 = ETB1 (multimedia network)
                                2 = ETB2 (other network)
                                3 = ETB3 (other network)
trnDirState      UINT8      -- TTDB status
                                1 = UNCONFIRMED
                                2 = CONFIRMED
opTrnDirState    UINT8      -- operational train directory status
                                1 = INVALID
                                2 = VALID
                                4 = SHARED
reserved03       UINT8      -- reserved for future use (= 0)
trnId            LABEL      -- train identifier, application defined
                                (e.g. "ICE75", "IC346"), informal
trnOperator      LABEL      -- train operator, e.g. "trenitalia.it",
                                informal
opTrnTopoCnt     UINT32     -- operational train topography counter
                                set to 0 if opTrnDirState = INVALID
crc              UINT32     -- SC-32 computed over record
                                (seed value: 'FFFFFFFF'H)
}

```

5.3.3.2.13 Structure OP_TRAIN_DIRECTORY

Le répertoire opérationnel de train représente la vision opérationnelle du train telle que définie en 4.2.4.1.

La structure de données OP_TRAIN_DIRECTORY est définie comme suit:

```

OP_TRAIN_DIRECTORY ::= RECORD
{
    version          VERSION      -- OpTrain data structure version
                                mainVersion = 1
                                subVersion = 0
    etbId            UINT8        -- identification of the ETB the train
                                directory computed for
                                0 = ETB0 (operational network)
                                1 = ETB1 (multimedia network)
                                2 = ETB2 (other network)
                                3 = ETB3 (other network)
    opTrnOrient      UINT8        -- operational train orientation
                                '00'B = unknown
                                '01'B = same as train direction
                                '10'B = inverse to train direction
    reserved01       UINT8        -- reserved for future use (= 0)
    reserved02       UINT8        -- reserved for future use (= 0)
    reserved03       UINT8        -- reserved for future use (= 0)
    opCstCnt         UINT8        -- number of consists in train (1..63)
    opCstList        ARRAY [] OF OP_CONSIST
                                -- dynamic operational consist list
                                ordered list starting with opCstNo = 1
    reserved04       UINT8        -- reserved for future use (= 0)
    reserved05       UINT8        -- reserved for future use (= 0)
    reserved06       UINT8        -- reserved for future use (= 0)
}

```



```

opVehCnt      UINT8          -- number of vehicles in train (1..63)
opVehList     ARRAY [] OF OP_VEHICLE
-- dynamic operational vehicle list
-- ordered list starting with opVehNo = 1
opTrnTopoCnt  UINT32         -- operational train topography counter
-- computed as defined in 5.3.3.2.16
-- from the beginning of the record until the
-- last element before the topography counter
-- (seed value: trnTopoCnt)
}

```

La structure de données OP_TRAIN_DIRECTORY (identique aux structures de données TRAIN_DIRECTORY et CONSIST_INFO) est définie au moyen de groupements dynamiques dans le but de réduire la longueur des télégrammes lors de la transmission, mais également d'avoir une définition facile des paramètres inclus dans le calcul du compteur de topographie. Pour la mise en œuvre, des groupements statiques pourraient être utilisés à la place, ce qui faciliterait l'accès aux paramètres membres du groupement. Le calcul du compteur de topographie a été ajusté de manière à obtenir le même résultat que défini ici.

5.3.3.2.14 Structure TRAIN_NETWORK_DIRECTORY_ENTRY (informelle)

L'entrée de répertoire de réseau de train représente une entrée dans la liste trnNetDir du répertoire de réseau de train. Pour obtenir une description plus détaillée, se reporter à l'IEC 61375-2-5.

La structure de données TRAIN_NETWORK_DIRECTORY_ENTRY est définie comme suit:

```

TRAIN_NETWORK_DIRECTORY_ENTRY ::= RECORD
{
  cstUUID      UINT8[16]      -- unique consist identifier
  cstNetProp    UINT32         -- consist network properties
-- bit 0..1: consist orientation
-- bit 2..7 = 0
-- bit 8..13 = ETBN Id
-- bit 14..15 = 0
-- bit 16..21 = Subnet Id
-- bit 24..29 = CN Id
-- bit 30..31 = 0
}

```

5.3.3.2.15 Structure TRAIN_NETWORK_DIRECTORY (informelle)

Le répertoire de réseau de train représente la topologie réseau d'un ETB telle que définie dans l'IEC 61375-2-5.

La structure de données TRAIN_NETWORK_DIRECTORY est définie comme suit:

```

TRAIN_NETWORK_DIRECTORY ::= RECORD
{
  reserved01    UINT16         -- reserved for future use (=0)
  entryCnt      UINT16         -- number of entries in train net directory
  trnNetDir     ARRAY [entryCnt] OF TRAIN_NETWORK_DIRECTORY_ENTRY
  etbTopoCnt    UINT32         -- train network directory CRC
}

```

5.3.3.2.16 Calcul des valeurs du compteur de topographie

La TTDB a défini trois compteurs de topographie, qui sont utilisés à différentes fins:

cstTopoCnt:	signature unique des informations de rame
trnTopoCnt:	signature unique du répertoire de train
opTrnTopoCnt:	signature unique du répertoire opérationnel de train

Pour le calcul de ces compteurs de topographie, les règles générales suivantes doivent s'appliquer:

- Tout d'abord, une somme de contrôle SC-32 (voir l'Article B.7) sur la plage de données et avec la valeur de base spécifiées doit être calculée.
- Si le calcul donne pour résultat une valeur différente de 0, on doit affecter au compteur de topographie la valeur de la somme de contrôle SC-32.
- Si le calcul donne pour résultat une valeur égale à 0, on doit affecter au compteur de topographie la valeur de $(2^{32} - 1)$.
- L'affectation de la valeur 0 au compteur de topographie le déclare invalide.
- La valeur du compteur de topographie doit être stockée au format big-endian.

NOTE 1 L'utilisation de la valeur 0 pour déclarer un compteur de topographie invalide évite de définir une balise de validité supplémentaire.

NOTE 2 La probabilité P_{crc} de ne pas détecter une modification d'un paquet de données protégé par un contrôle de redondance cyclique SC-32 est d'environ $P_{crc} \approx 2^{-32}$. La probabilité P_{tc} de ne pas détecter une modification d'un paquet de données signé par un compteur de topographie tel que spécifié dans la présente partie de l'IEC 61375 est d'environ $P_{tc} \approx (2^{-32} + 2^{-63})$, une différence qui est négligeable.

5.3.4 Base de Données Topologiques de Train pour plusieurs ETB (option)

5.3.4.1 Généralités

La TTDB est conçue sur la base du répertoire de réseau de train, qui représente la vision du réseau central de train par un ETB particulier. Cette vision du réseau central de train peut être différente d'un ETB à l'autre dans le train. Si une TTDB était calculée pour chaque ETB, la conséquence serait qu'un répertoire de train et un répertoire opérationnel de train différents seraient calculés pour chaque ETB, ce qui pourrait donner lieu à différentes vues du train selon l'ETB utilisé pour référence.

Afin d'éviter toute ambiguïté dans les vues de train comportant plusieurs ETB (voir 4.2.2.6.1), le répertoire de train et le répertoire opérationnel de train doivent être calculés par l'ETB0 (ETB opérationnel) seulement et doivent être adoptés par les autres ETB.

5.3.4.2 Adoption de la TTDB au sein d'ETB dépendants

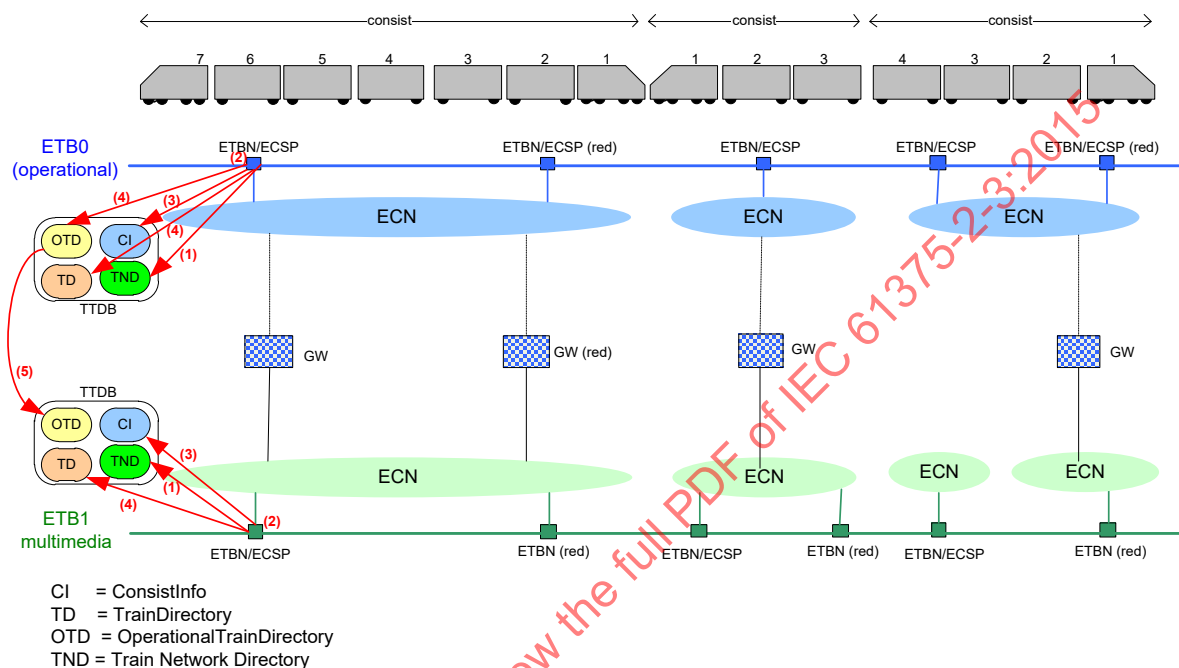
L'adoption du répertoire de train et du répertoire opérationnel de train à partir de l'ETB0 peut se définir par un processus en 5 étapes (illustré à la Figure 17):

- Chaque ETB exécute l'inauguration du train et génère le répertoire de réseau de train tel que défini dans l'IEC 61375-2-5.
- Selon le résultat de l'inauguration, les adresses IP des ETBN ainsi que les adresses de sous-réseau des réseaux de rames connectés sont affectées de la manière définie dans l'IEC 61375-2-5.
- Les télégrammes CSTINFO sont échangés sur chaque ETB comme indiqué en 5.2.3 et stockés dans le bloc de données CONSIST_INFO_LIST. Le répertoire de train est établi tandis que les télégrammes CSTINFO sont enregistrés.

- d) L'ETB0 calcule le répertoire opérationnel de train.
 e) Les autres ETB copient ou référencent le répertoire opérationnel de train calculé par l'ETB0.

NOTE 1 Au lieu de copier le répertoire opérationnel de train, un accès peut également être fourni depuis un autre ETB au répertoire opérationnel de train de l'ETB0 ('référencement').

Tous les ETB doivent continuellement surveiller la valeur opTrnTopoCnt de l'ETB0, et en cas de changement, doivent mettre à jour leur TTDB.



IEC

Anglais	Français
consist	rame
ETB0 (operational)	ETB0 (opérationnel)
ETB1 multimedia	ETB1 multimédia
GW	Passerelle
Consist Info	Informations de rame
Train Directory	Répertoire de train
Operational Train Directory	Répertoire opérationnel de train
Train Network Directory	Répertoire du réseau de train

Figure 17 – Adoption de la TTDB (illustrée par rapport à la première rame dans cet exemple)

NOTE 2 Une conséquence sera qu'une rame injoignable sur l'ETB0, mais disponible sur le réseau multimédia, ne sera pas répertoriée dans la TTDB du réseau multimédia.

5.3.4.3 Restrictions relatives à l'architecture réseau

Afin de s'assurer que tous les télégrammes CSTINFO sont envoyés par toutes les rames sur l'ensemble des ETB, il convient que chaque rame du train soit connectée à l'ensemble des ETB.

5.3.4.4 Restrictions relatives aux télégrammes CSTINFO

Les télégrammes CSTINFO envoyés par une rame sur les différents ETB doivent indiquer le même numéro, la même séquence, la même orientation et la même identité des véhicules contenus.

Les fonctions devant être accessibles à partir des différents ETB doivent être répertoriées dans les télégrammes CSTINFO associés à tous les ETB.

NOTE Un cas type d'utilisation est celui d'une fonction du réseau opérationnel qui envoie des données à une (autre) fonction (de rame) distante sur le réseau multimédia. Elle cherche ensuite toutes les informations de la fonction distante sur sa TTDB locale.

Toutes les autres données figurant dans le télégramme CSTINFO peuvent être différentes vu qu'elles ne présentent pas d'intérêt en ce qui concerne l'interopérabilité.

5.4 Adressage des services

5.4.1 Généralités

L'adressage des services est une partie essentielle du système de communication, car il identifie les points d'extrémité d'une relation de communication ainsi que les partenaires de la communication. La communication, ainsi que l'adressage associé, peuvent être définis entre des entités physiques telles que des appareils physiques ou des êtres humains, mais peuvent également être définis entre des entités plus abstraites telles que des fonctions d'application ou d'un système.

Un schéma d'adressage d'entités physiques sur l'ensemble du train est défini dans l'IEC 61375-2-5.

La présente partie de la norme définit un schéma d'adressage qui doit être utilisé pour l'adressage des fonctions embarquées sur l'ensemble du train.

Au niveau fonctionnel, un schéma d'adressage plus abstrait basé sur des noms de domaine TCN (voir 5.4.3) est défini et est équivalent au système de noms de domaine défini pour Internet. Les principales raisons d'introduction d'un schéma d'adressage supplémentaire, en plus des adresses IP, sont les suivantes:

- Sert à gérer des fonctions abstraites au lieu d'adresses IP associées aux appareils. Par exemple, la même fonction pourrait être accessible par des adresses IP différentes dans des rames différentes, mais elle porte le même nom dans toutes les rames.
- Les adresses IP peuvent être modifiées sans qu'il y ait une influence sur le nom de domaine.
- Les noms peuvent être plus facilement compris et interprétés par des êtres humains. Cela pourrait devenir plus important dans le futur lorsque les adresses IPv6 seront utilisées.

L'adressage fonctionnel est relatif aux fonctions logiques.

Les sources d'information peuvent être:

- une fonction au sein d'un véhicule ou d'une rame.

Les destinations d'information peuvent être:

- une fonction au sein d'un véhicule ou d'une rame.
- une fonction au sein de tout ou partie des rames.

NOTE Une fonction peut être mise en œuvre par un ou plusieurs appareils terminaux dans une rame.

5.4.2 Système de Noms de Domaine TCN (DNS TCN)

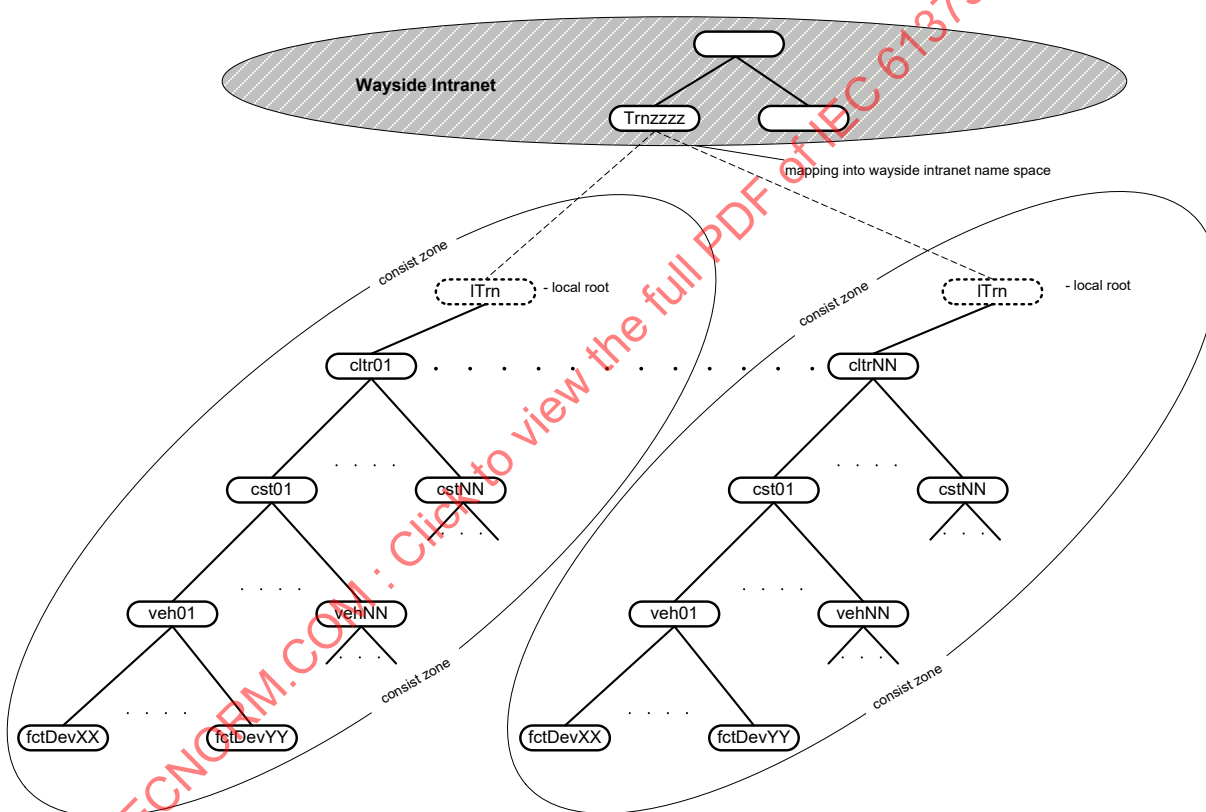
5.4.2.1 Généralités

Le Système de Noms de Domaine TCN définit un schéma de nommage hiérarchique, basé sur un domaine, ainsi qu'une base de données répartie pour mettre en œuvre ce schéma de nommage. Son intérêt premier est de mapper les noms d'hôtes en adresses IP.

NOTE Le concept de DNS TCN est fortement lié au DNS Internet tel que défini dans les RFC 1034 et RFC 1035.

5.4.2.2 Espace et Zones de Noms de Domaine TCN

L'Espace de noms de domaine TCN est défini par une arborescence, le train local ('lTrn') étant la racine (locale). Le premier niveau au-dessous de la racine est défini par les sous-domaines de l'ensemble de rames insécables (cltr01 ... cltrNN). Les sous-domaines de l'ensemble de rames insécables sont encore structurés en sous-domaines de rame (cst01 ... cstNN), les sous-domaines de véhicule (veh01 ... vehNN) aboutissant aux hôtes qui sont les appareils hébergeant les fonctions logiques (fctDevXX ... fctDevYY).



IEC

Anglais	Français
Wayside Intranet	Intranet au sol
mapping into wayside Intranet name space	mapping vers l'espace de noms de l'Intranet au sol
consist zone	zone rame
local root	racine locale

Figure 18 – Espace de noms DNS TCN avec division en zones

Pour les communications sol-trains, l'ensemble de l'espace de noms DNS TCN peut être mappé avec des intranets supérieurs appartenant par exemple à l'opérateur ou au prestataire de maintenance du train (cela ne relevant néanmoins pas du domaine d'application de la présente partie de la norme). Dans ce cas, l'espace de noms DNS TCN mappé peut être identifié par le nom racine spécifique d'un opérateur (ex: 'Trnzxxx').

L'espace de noms DNS TCN est divisé en zones, dont chacune est administrée par un serveur de noms DNS faisant autorité.

Des sous-domaines de rame peuvent être définis en tant que zone unique comme le montre la Figure 18, un serveur de noms DNS ayant autorité pour desservir cette zone. Dans les rames comportant plusieurs réseaux de rames, il pourrait être avantageux de subdiviser le sous-domaine de rame en plusieurs zones et de laisser le serveur de noms DNS de rame déléguer ces zones à des serveurs de noms DNS (subordonnés).

5.4.3 Noms de domaine TCN

Les entités de communication (sources et destinations) doivent être identifiées par un nom de domaine TCN entièrement qualifié, dérivé de l'espace de noms de domaine TCN.

5.4.4 Schéma de l'URI TCN

5.4.4.1 Généralités

La syntaxe de l'espace des noms de domaine TCN est définie par un schéma URI suivant les recommandations données dans la RFC 3986.

5.4.4.2 Syntaxe schématique de l'URI TCN

La syntaxe de l'URI TCN est définie conformément à la RFC 3986 en utilisant la notation BNF (Backus Naur Form). La définition spécifique de l'URI TCN est soumise aux paragraphes suivants.

L'URI TCN utilise des éléments de base pour la définition de la syntaxe, qui sont définis au Tableau 6.

Tableau 6 – Syntaxe de base de l'URI TCN

```
label = alpha 1{uchar}13.alphanum
uchar = alphanum | mark
alphanum = alpha | digit
alpha = "a" | "b" | "c" | "d" | "e" | "f" | "g" | "h" | "i" |
        "j" | "k" | "l" | "m" | "n" | "o" | "p" | "q" | "r" |
        "s" | "t" | "u" | "v" | "w" | "x" | "y" | "z" |
        "A" | "B" | "C" | "D" | "E" | "F" | "G" | "H" | "I" |
        "J" | "K" | "L" | "M" | "N" | "O" | "P" | "Q" | "R" |
        "S" | "T" | "U" | "V" | "W" | "X" | "Y" | "Z"
digit = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
mark  = "-"
```

NOTE Cette définition d'étiquette (label) s'écarte de la RFC 1035 en ce qui concerne les points suivants:

- la longueur maximale est limitée à 15 caractères (RFC 1035: 63 caractères permis);
- le premier caractère ne peut pas être un chiffre (RFC 1035: un chiffre permis).

5.4.4.3 Schéma général de l'URI TCN

L'identificateur URI TCN doit être utilisé pour identifier la fonction/l'appareil source et la fonction/l'appareil destinataire de l'information. Le schéma général de l'URI TCN est défini au Tableau 7.

Tableau 7 – Schéma général de la syntaxe

```

TCN-URI = [scheme ":" "//"] user "@" host
scheme   = "trn"
user      = usr
host      = fctdev "." vehicle "." consist "." [cltrain "."]train
["."]

```

D'après cette définition, l'identificateur URI TCN peut s'écrire comme suit (voir Figure 19):

trn://usr@fctdev.vehicle.consist.cltrain.train

scheme
user part
host part

Anglais	Français
scheme	schéma
user part	section utilisateur
host part	section hôte

Figure 19 – Schéma d'URI TCN

NOTE 1 La section hôte de l'URI TCN peut se résoudre en une adresse IP avec le service DNS TCN.

NOTE 2 L'étiquette 'fctdev' définit l'appareil logique qui met en œuvre la fonction de service adressée. Un appareil "logique" signifie que la manière dont est mis en œuvre cet appareil n'est pas connue. Par exemple, il pourrait s'agir un appareil physique simple mettant en œuvre la fonction ou la fonction pourrait être distribuée sur plusieurs appareils physiques.

Pour l'adressage d'un appareil physique, le schéma d'adressage physique défini dans l'IEC 61375-2-5 doit être utilisé.

5.4.4.4 Domaine d'application de l'URI TCN

Dans le cadre d'une relation de communication, la source et la destination de l'information sont identifiées par des URI TCN. Le fait qu'un URI TCN spécifique soit relatif à une source ou une destination est appelé son domaine d'application. Les règles de syntaxe des URI TCN peuvent être différentes selon que l'URI TCN est utilisé pour une source ou une destination.

5.4.4.5 Section utilisateur de l'URI TCN (facultatif)

La section utilisateur peut être utilisée pour les profils d'application définis dans l'IEC 61375-2-4 afin de distinguer différentes fonctions et instances de fonction, situées sur le même appareil physique.

La présente partie de l'IEC 61375 définit une chaîne de caractères de 32 octets, qui est réservée pour la section utilisateur.

5.4.4.6 Section hôte de l'URI TCN

5.4.4.6.1 Directions, orientations et numéros dans un train (informatif)

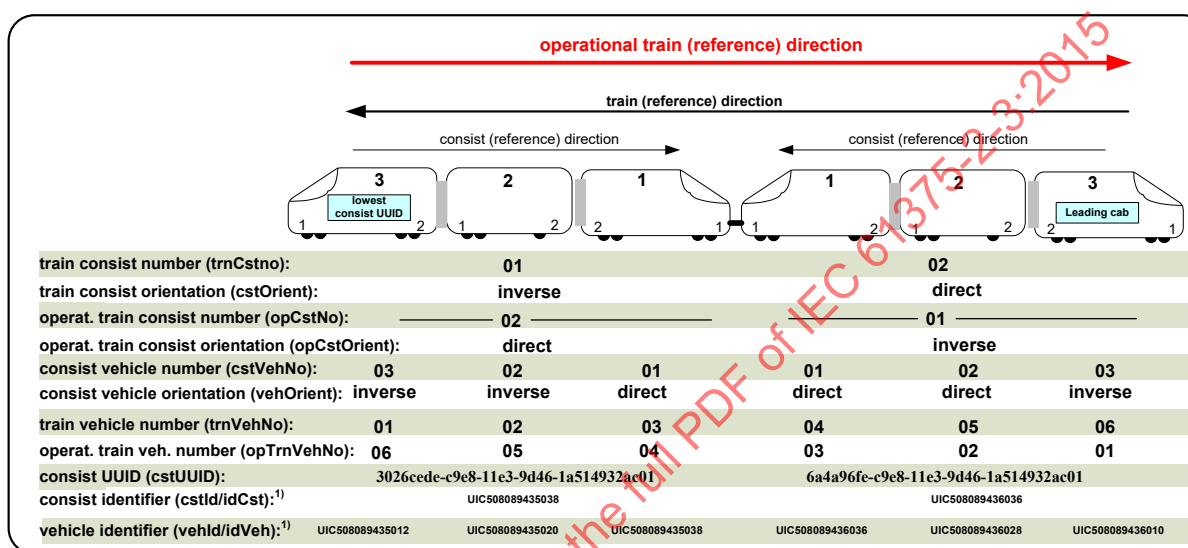
La navigation dans un réseau de train exige une définition précise des directions et des schémas de numérotation dans un train. Les directions et numérotations sont définies dans

les différentes parties de l'IEC 61375. La Figure 20 a pour but de fournir un récapitulatif de ces directions et numérotations.

NOTE 1 La direction de référence de la rame est basée sur la construction de la rame. Le véhicule 1/cabine 1 est toujours l'extrémité 1 (avant) de la rame. La direction de référence de rame sert à la numérotation des véhicules au sein de la rame.

NOTE 2 La direction de référence TCN est basée sur l'UUID des rames tel que défini dans l'IEC 61375-2-5. La rame de plus faible UUID est toujours la rame 1 du TCN.

NOTE 3 La direction opérationnelle est basée sur le véhicule de tête/la rame de tête. L'extrémité 1 (extrémité avant) du train est définie en fonction de la direction de vision de la cabine de conduite du véhicule de tête (voir 4.2.4.3). La direction opérationnelle peut servir à la gestion de la direction opérationnelle en fonction des signaux.



¹⁾ Identifiers as defined by UIC are used in these examples, e.g. 'UIC508089435038' corresponds to UIC identifier '50 80 89 - 43 503 - 8'

IEC

Anglais	Français
operational train (reference) direction	direction (de référence) opérationnelle du train
train (reference) direction	direction (de référence) du train
consist (reference) direction	direction (de référence) de la rame
lowest consist UUID	UUID de rame le plus faible
leading cab	cabine de tête
train consist number (trnCstno):	numéro de la rame dans le train (trnCstno):
train consist orientation (cstOrient):	orientation de la rame dans le train (cstOrient):
operat. train consist number (opCstNo):	numéro de la rame opérationnelle dans le train (opCstNo):
operat. train consist orientation (opCstOrient):	orientation de la rame opérationnelle dans le train (opCstOrient):
train vehicle number (trnVehNo):	numéro du véhicule dans le train (trnVehNo):
operat. train veh. number (opTrnVehNo):	numéro du véhicule opérationnel dans le train (opTrnVehNo):
consist UUID (cstUUID):	UUID de rame (cstUUID):
consist identifier (cstId):	identificateur de rame (cstId):
veh. identifier (vehId):	Identificateur du véhicule (vehId):

Figure 20 – Directions, orientations et numéros dans le train

5.4.4.6.2 Etiquette de fonction/appareil

Les appareils de fonction doivent être identifiés par des étiquettes en suivant la syntaxe donnée au Tableau 8:

Tableau 8 – Syntaxe des étiquettes d'appareil

```
fctDev = "lDev" | deviceName | group
devicename = label
group = "grpAll" | grpDev
grpDev = "grp" 1{uchar}12
```

Le sens des différentes étiquettes d'appareil est défini au Tableau 9.

Tableau 9 – Définition des étiquettes d'appareil

Etiquette <Appareil>	Domaine d'application S=source D=destinataire	Obligatoire (O) Facultatif (F)	Définition
"lDev"	D	F	La fonction destinataire est localisée sur l'appareil terminal local. A utiliser pour l'adressage des fonctions résidant sur l'appareil considéré.
deviceName	S,D	O	Définit l'appareil de fonction (appareil logique). EXEMPLE 'fctDoorCtrl' (fonction de contrôle des portes), 'fctBrake' (fonction de freinage), 'devHMI' (interface homme-machine). Cette étiquette sera résolue en adresse de monodiffusion IP.
grpDev	D	O	Définit un groupe d'appareils de fonction (appareil logique). EXEMPLE 'grpDoorCtrl' (groupe contrôle des portes), 'grpBrake' (groupe freinage), 'grpETBN' (groupe ETBN), 'grpECSC' (groupe ECSC). Cette étiquette sera résolue en adresse de multidiffusion IP.
"grpAll"	D	O	Tous les appareils de fonction (appareil logique) ou l'appareil physique. Cette étiquette sera résolue en adresse de multidiffusion IP.
anyDev	S	F	L'emplacement de la fonction source est inconnu. Cette étiquette peut être utilisée pour l'abonnement TRDP en fonction du ComId, ainsi que de la rame source ou de l'ensemble de rames insécables source. Cette étiquette doit être utilisée conjointement avec 'anyVeh' et sera résolue en plage d'adresses IP couvrant tous les réseaux de rames d'une rame ou d'un élément de rames insécables. EXEMPLE 1 L'URI TCN 'anyDev.anyVeh.cst05.anyCITrn.ITrn' peut être utilisé pour s'abonner à un ComId spécifique de la rame 05. EXEMPLE 2 L'URI TCN 'anyDev.anyVeh.leadCcst.anyCITrn.ITrn' peut être utilisé pour s'abonner à un ComId spécifique de la rame de tête.

NOTE Des noms d'alias peuvent également être définis pour les appareils, ce qui signifie que le même appareil physique apparaît avec des étiquettes d'appareil différentes.

Il convient d'utiliser des préfixes définis pour les noms d'appareil, par exemple 'fct' pour un appareil de fonction (appareil logique) et 'dev' pour un appareil physique.

Pour prendre en charge l'adressage fonctionnel abstrait, les étiquettes d'appareils ou les étiquettes de groupes d'appareils peuvent être normalisées. Par exemple, la fonction "brake" peut être définie pour être adressée via une étiquette d'appareil "fctBrake" qui est normalisée pour l'ensemble des applications TCN. La présente norme n'a cependant pas pour but de donner ces définitions.

5.4.4.6.3 Etiquette de véhicule

Les véhicules doivent être identifiés par des étiquettes en suivant la syntaxe donnée au Tableau 10:

Tableau 10 – Syntaxe des étiquettes de véhicule

```
veh = "lVeh" | cstVehNo | opVehNo | "leadVeh" | "aVeh" |
      idVeh | "anyVeh"
cstVehNo = "veh" digit digit
opVehNo = "opVeh" digit digit
idVeh = label
```

Le sens des différentes étiquettes de véhicule est défini au Tableau 11.

Tableau 11 – Définition des étiquettes de véhicule (veh)

Etiquette <veh>	Domaine d'application S=source D=destinataire	Obligatoire (O) Facultatif (F)	Définition
lVeh	D	F	La fonction est dans le véhicule considéré.
aVeh	D	O	Fonctions localisées dans tout ou partie des véhicules. L'étiquette de véhicule "aCar" ne peut être utilisée qu'en combinaison avec une étiquette d'appareil de groupe ("grpName"). Les appareils terminaux qui sont membres d'un groupe d'appareils peuvent être répartis sur la rame ou, pour les groupes couvrant l'ensemble du train, sur la totalité du train.
opVehNo	S,D	O	Fonction dans le véhicule avec le numéro opérationnel x e {01,...,63}. Cette étiquette peut seulement être utilisée conjointement avec l'étiquette "anyCst". L'étiquette 'opVehno' est unique au sein du train et se réfère au véhicule de tête (direction opérationnelle). EXEMPLE 'opVeh24'
cstVehNo	S,D	O	Fonction dans le véhicule de la rame avec le numéro cstVehNo e {01,...,32} EXEMPLE 'veh05'
leadVeh	S,D	O	La fonction est dans le véhicule de tête. Cette étiquette peut seulement être utilisée conjointement avec l'étiquette "leadCst".

Etiquette <veh>	Domaine d'application S=source D=destinataire	Obligatoire (O) Facultatif (F)	Définition
idVeh	S,D	O	Fonction dans un véhicule avec un identificateur unique (ex: numéro de véhicule UIC à 7/12 chiffres) EXEMPLE "UIC948002343044" // véhicule portant l'identificateur UIC 94800234304-4
anyVeh	D	O	Fonction localisée dans l'un des véhicules. Cette étiquette peut être utilisée lorsque la localisation de l'appareil dans une rame n'a aucune importance. En pareil cas, l'étiquette d'appareil doit être unique au sein de la rame.

5.4.4.6.4 Etiquette de rame

Les rames doivent être identifiées par des étiquettes en suivant la syntaxe donnée au Tableau 12:

Tableau 12 – Syntaxe des étiquettes de rame

```

consist = "lCst" | "aCst" | "anyCst" | "leadCst" | cstNo
          | opCstNo | idCst
cstNo = "cst" digit digit
opCstNo = "opCst" digit digit
idCst = label

```

Le sens des différentes étiquettes de rame est défini au Tableau 13.

Tableau 13 – Définition des étiquettes de rame

Etiquette <rame>	Domaine d'application S=source D=destinataire	Obligatoire (O)/ Facultatif (F)	Définition
lCst	D	F	Fonction localisée dans la rame considérée (locale)
cstNo	S,D	O	Fonctions localisées dans la rame avec le numéro trnCstNo e {01,...,63}. EXEMPLE 'cst05' Il s'agit du numéro de séquence de rame obtenu à partir du répertoire de train, ce qui ne respecte pas les corrections. Pour l'adressage des fonctions dans un train en service, l'étiquette 'opCstNo' peut être utilisée à la place, ce qui respecte également la correction.
opCstNo	S,D	F	Fonctions localisées dans la rame avec le numéro opérationnel opCstNo x e {01,...,63}. Cette étiquette peut seulement être utilisée conjointement avec l'étiquette "anyVeh". L'étiquette est unique au sein du train et se réfère à la rame de tête (direction opérationnelle). EXEMPLE 'opCst29'
aCst	D	O	Fonctions localisées dans tout ou partie des rames. Cette étiquette peut seulement être utilisée conjointement avec une étiquette de groupe d'appareils.

Etiquette <rame>	Domaine d'application S=source D=destinataire	Obligatoire (O)/ Facultatif (F)	Définition
anyCst	D	O	Fonction localisée dans l'une des rames. Cette étiquette peut être utilisée lorsque la localisation de l'appareil dans le train n'a aucune importance, auquel cas l'étiquette d'appareil doit être unique au sein du train, ou cette étiquette doit être utilisée si un véhicule opérationnel est adressé. Cette étiquette peut seulement être utilisée conjointement avec l'étiquette "anyVeh" ou avec l'étiquette "opVehNo".
idCst	S,D	O	Identificateur unique d'une rame EXEMPLE 'UIC948002343044' // Rame identifiée par son véhicule de rame avec identificateur UIC 94 80 0234 304-4
leadCst	S,D	O	La fonction destinataire est dans la rame qui contient le véhicule de tête.

5.4.4.6.5 Etiquette d'ensemble de rames insécables (facultatif)

Les ensembles de rames insécables doivent être identifiés par des étiquettes en suivant la syntaxe donnée au Tableau 16:

Tableau 14 – Syntaxe d'étiquette d'ensemble de rames insécables

```
cltrain = "lClTrn" | "anyClTrn" | "aClTrn"
```

Le sens des différentes étiquettes d'ensemble de rames insécables est défini au Tableau 17.

Tableau 15 – Définition des étiquettes d'ensemble de rames insécables

Etiquette <cltrain>	Domaine d'application S=source D=destinataire	Obligatoire (O)/Facultatif (F)	Définition
lClTrn	S,D	F	Fonction destinataire/source dans l'ensemble de rames insécables considéré
anyClTrn	D	O	Fonction localisée dans l'un des ensembles de rames insécables. Cette étiquette peut également être utilisée en l'absence d'ensemble de rames insécables.
aClTrn	D	O	Fonction localisée dans tous les ensembles de rames insécables. Cette étiquette peut également être utilisée en l'absence d'ensemble de rames insécables.

5.4.4.6.6 Etiquette de train

Les trains doivent être identifiés par des étiquettes en suivant la syntaxe donnée au Tableau 16:

Tableau 16 – Syntaxe d'étiquette de train

```
train = "lTrn"
```

Le sens des différentes étiquettes de train est défini au Tableau 17.

Tableau 17 – Définition des étiquettes de train

Etiquette <train>	Domaine d'application S=source D=destinataire	Obligatoire (O)/ Facultatif (F)	Définition
lTrn	S,D	O	Fonction destinataire/source dans le train considéré

NOTE Le schéma d'URI décrit ici est seulement défini pour l'adressage à bord du train, mais peut facilement être étendu à (et est préparé pour) l'adressage sol-train et train-train. Une extension peut, par exemple, avoir la forme suivante:

```
trdp://instance.function@fctdev.vehicle.consist.cltrain.train.fleet.operator
```

5.4.5 Mapping d'URI TCN en adresses IP

5.4.5.1 Généralités

Les URI TCN doivent être mappés en adresses IP conformément au schéma d'adressage IP défini dans l'IEC 61375-2-5.

La résolution d'URI en adresses IP est généralement gérée par le service DNS TCN comme spécifié en 5.4.2.

5.4.5.2 Mapping d'adresses de groupes de multidiffusion IP

L'IEC 61375-2-5 affecte la plage d'adresses IP 239.192.0.0/14 pour l'adressage des groupes de multidiffusion IP de train. Pour utiliser cette plage d'adresses conjointement avec le schéma d'URI TCN défini, en particulier pour prendre en charge les groupes, qui sont limités à une rame, une décomposition de la plage assignée d'adresses de groupes de multidiffusion IP doit être définie.

La décomposition générale est la suivante:

```
11101111.110000rr.zzzzzzzz.zzzzzzzz
```

avec les champs décrits au Tableau 18.

Tableau 18 – Décomposition générale des adresses de groupes de multidiffusion IP

Section numéro de sous-réseau:	
[r]	Définit la plage. '00'B = groupes relatifs à tous les trains '01'B = groupes associés à un ETB '10'B = groupes limités à une rame '11'B = ouverte à d'autres usages
[z]	A des fins de décomposition supplémentaire

Les groupes relatifs à tous les trains sont des groupes dont les membres sont associés à différents ETB¹ du train.

La décomposition des groupes relatifs à tous les trains est la suivante:

11101111.11000000.gggggggg.gggggggg

avec les champs décrits au Tableau 19.

Tableau 19 – Décomposition des groupes relatifs à tous les trains

Section numéro d'hôte:	
[g]	Numéro de groupe pour tous les trains (plage de valeurs: 0 .. 65535) 0 .. 255 = comme spécifié dans l'IEC 61375-2-5 256 .. 65534 = utilisable pour les groupes relatifs à tous les trains définis par l'utilisateur 65535 = réservé à une utilisation future

EXEMPLE Un groupe de multidiffusion IP contenant des appareils terminaux issus du réseau opérationnel et du réseau multimédia est un groupe relatif à tous les trains.

Une adresse de groupe relatif à tous les trains doit être affectée à une fonction si le paramètre FUNCTION_INFO 'etbld' est défini sur 255 (voir 5.3.3.2.4).

NOTE 1 L'utilisation du TRDP pour la communication avec les groupes relatifs à tous les trains est problématique, car les groupes relatifs à tous les trains peuvent comporter des appareils terminaux associés à différents ETB et les valeurs etbTopoCnt sont différentes pour les ETB. La valeur etbTopoCnt est contenue dans l'en-tête TRDP et utilisée par les appareils terminaux destinataires afin de vérifier l'exactitude du message reçu, qui échouera si la valeur etbTopoCnt contenue est associée à un autre ETB. L'utilisation d'une passerelle entre les ETB altérant la valeur etbTopoCnt pourrait être une solution, mais reste complexe.

Les groupes associés à un ETB sont des groupes dont les membres sont associés à un ETB.

La décomposition des groupes associés à un ETB est la suivante:

11101111.11000001.bbpgggggg.gggggggg

avec les champs décrits au Tableau 20.

Tableau 20 – Décomposition des groupes associés à un ETB

Section numéro de sous-réseau:	
[b]	Valeur de l'identificateur ETB (etbld) = 0..3
Section numéro d'hôte:	
[g]	Numéro de groupe (plage de valeurs: 0 .. 16383) 0 = tous les appareils terminaux 1 = tous les ECSP (ETB0 seulement) 2..16382 = utilisable pour les groupes associés à un ETB définis par l'utilisateur 16383 = réservé à une utilisation future

EXEMPLE Un groupe de multidiffusion IP contenant des appareils terminaux issus du réseau opérationnel seulement est un groupe associé à un ETB. Un groupe de multidiffusion IP contenant des appareils terminaux issus du réseau multimédia est un autre groupe associé à un ETB.

¹ Pour la définition de l'expression '... relative(s) à un ETB', voir 4.2.2.6.2.

Les groupes limités à une rame sont des groupes associés à un ETB dont les membres appartiennent à une rame.

La décomposition des groupes associés à un ETB est la suivante:

11101111.11000010.bbgggggg.gggggggg

avec les champs décrits au Tableau 21.

Tableau 21 – Décomposition des groupes limités à une rame

Section numéro de sous-réseau:	
[b]	Valeur de l'identificateur ETB (etbld) = 0..3
[c]	trnCstNo (plage de valeurs: 1 .. 63) 0 = rame locale
Section numéro d'hôte:	
[g]	Numéro de groupe (plage de valeurs: 0 .. 254) 0 = tous les appareils terminaux de tous les réseaux de rames de la rame connectée à l'ETB 1 = ECSP de la rame (ETB0 seulement) 2 .. 254 = utilisable pour les groupes limités à une rame définis par l'utilisateur 255 = réservé à une utilisation future

Les identificateurs de groupe ont besoin d'être uniques au sein d'une rame. La même plage de valeurs d'identificateur de groupe à 8 bits est utilisée à la fois pour les groupes limités à un véhicule et les groupes limités à une rame, ce qui implique qu'elle doit être partagée. Par exemple, la valeur de grpHMI est différente pour ce groupe défini au niveau rame ("grpHMI.aVeh.IVeh") et pour ce groupe défini au niveau véhicule (ex: "grpHMI.veh02.ICst").

Noter que "tous les appareils terminaux" est codé en définissant tous les bits grp-id sur 0 et non sur 1. Cette convention est utilisée pour éviter toute confusion, car les bits grp-id définis sur 1 pourraient également signifier "tous les groupes".

5.4.5.3 Rattachement de groupes de multidiffusion IP

Pour rejoindre des groupes de multidiffusion IP, il convient de tenir compte de certains aspects.

- Les groupes relatifs à tous les trains et les groupes associés à un ETB possèdent des adresses de multidiffusion IP définies qui ne changeront pas en cas de modification de la composition du train, ce qui fait que les appareils terminaux ont juste à rejoindre le groupe au démarrage du système.
- Les groupes limités à une rame sont des groupes de multidiffusion IP dynamiques possédant des adresses de groupes de multidiffusion IP qui peuvent changer après chaque inauguration de train. Étant donné leur nature dynamique, tous les appareils terminaux rejoignant un tel groupe ont besoin de quitter le groupe dans le cadre d'une nouvelle inauguration de train et de le rejoindre avec la nouvelle adresse de multidiffusion IP. Il convient également de traduire les adresses de multidiffusion IP tout en routant les télégrammes IP de l'ETB à l'ECN (ETBN). Durant cette traduction d'adresse réseau (NAT, voir RFC 2663, RFC 2766, RFC 3022), les bits [c] représentant le numéro de rame du train dans l'adresse de destination de multidiffusion IP peuvent être substitués par 0 (rame locale). Dans ce cas, les appareils terminaux ont juste à rejoindre le groupe au démarrage du système.
- Il convient que les appareils terminaux pouvant recevoir des télégrammes TRDP adressés aux groupes limités à une rame ainsi qu'aux groupes relatifs à un ETB et aux groupes relatifs à tous les trains rejoignent les deux groupes de multidiffusion IP.

EXEMPLE La fonction 'fctDoor' peut être adressable à la fois sur l'ensemble du train (par rapport à l'ETB0) (ex: URI TCN 'fctDoor.anyVeh.aCst.aCItrn.ITrn'), mais également être limitée à une rame (ex: URI TCN 'fctDoor.anyVeh.cst01.anyCItrn.ITrn'). Si nous supposons que la traduction d'adresse réseau est utilisée et que la fonction fctDoor possède la valeur fctId=50, les deux adresses de multidiffusion IP suivantes ont besoin d'être rattachées: 239.193.0.50 et 239.194.1.50.

5.4.5.4 URI TCN prédéfinis

Certains mappings d'URI TCN en adresses IP sont prédéfinis et n'exigent pas une résolution explicite de l'URI par l'intermédiaire de DNS TCN. Ces URI TCN sont répertoriés au Tableau 22.

Tableau 22 – URI TCN prédéfinis

URI TCN (section hôte)	Domaine d'application	Adresse IP	Description
grpAll.aVeh.aCst.aCItrn.ITrn	D	239.193.0.0	Diffusion vers tous les appareils terminaux associés à l'ETB0
grpAll.aVeh.ICst.ICITrn.ITrn	D	239.194.0.0	Diffusion vers tous les appareils terminaux associés à l'ETB0 au sein de la rame locale
grpAll.aVeh.ICst.ICITrn.ITrn	D	239.255.0.0	Diffusion alternative vers tous les appareils terminaux au sein de la rame locale si tous les appareils terminaux sont connectés au même réseau de rame (seulement un réseau de rame connecté à l'ETB) NOTE Cette adresse est définie dans l'IEC 61375-3-4.
IDev.IVeh.ICst.ICITrn.ITrn	S,D	127.0.0.1	Appareil considéré (rebouclage local)
grpECSP.anyVeh.aCst.aCItrn.ITrn	D	239.193.0.1	Diffusion vers tous les ECSP

EXEMPLE

L'exemple suivant donne quelques conseils pour la mise en œuvre de l'URI grpAll.aVeh.ICst.ICITrn.ITrn dans une rame comportant plusieurs réseaux de rames.

Chaque appareil terminal dans une rame s'abonne à l'adresse 239.194.0.0.

Les ETBN utilisent le résultat de l'inauguration ETB pour ajouter une règle à leur table de routage de multidiffusion pour l'adresse 239.194.X.0 où X représente leur rame considérée ($X = \text{trnCstNo} + (\text{etbId} \times 64)$). Toutes les autres adresses à partir de la plage 239.194.X.0 ne sont pas acheminées jusqu'à l'ECN.

Les ETBN traduisent l'adresse de multidiffusion 239.194.X.0 en 239.194.0.0 lorsqu'ils l'acheminement de l'ETB à l'ECN.

En cas de réseaux de rames multiples dans une rame, le paquet possédant l'adresse de destination 239.194.0.0 doit être géré par les ETBN de cette rame également afin de la communiquer à l'ensemble des réseaux de rames. Une solution consiste à l'acheminer à partir du réseau de rame source à l'ETB et de traduire l'adresse de destination en 239.194.X.0. Une autre solution est que l'ETBN dans le réseau de rame source transmet en mode tunnel le paquet aux autres ETBN de la rame en utilisant l'adresse de monodiffusion IP et que les ETBN de destination retransmettent ensuite le paquet en mode multidiffusion dans leur réseau de rame. La première approche est plus facile à mettre en œuvre, mais le paquet est envoyé à l'ensemble des ETBN du train. L'approche utilisant le mode tunnel exige une plus grande prise en charge de la part des ETBN; par contre le paquet ne franchit jamais les limites de la rame.

5.4.5.5 Acheminement IP dans les réseaux comportant plusieurs ETB

Les règles de base pour la gestion de l'acheminement IP dynamique sont définies dans l'IEC 61375-2-5.

La règle supplémentaire suivante doit être observée:

Pour la transmission de paquets IP à un appareil terminal destinataire qui est associé à un autre ETB que celui auquel est associé l'appareil terminal source, les paquets IP doivent être transférés via l'ETB auquel est associé l'appareil terminal destinataire.

NOTE 1 Pour la définition de l'expression 'appareil terminal associé à un ETB', voir 4.2.2.6.2.

NOTE 2 Cette règle s'explique par le fait qu'un appareil terminal émetteur ne peut pas s'attendre à ce qu'un paquet IP puisse être acheminé entre les ETB de la rame destinataire. Il a donc besoin d'être acheminé au sein de la rame de l'émetteur jusqu'à l'ETB auquel est associé l'appareil terminal destinataire.

EXEMPLE Cet exemple utilise l'architecture de train présentée à la Figure 21. Si l'appareil terminal source de la rame 'B' souhaite envoyer un paquet à l'appareil terminal portant le fctDevNo = 10020 dans la rame 'E', il doit commencer par acheminer le paquet vers son ECN multimédia local et ensuite via l'ETB1 jusqu'à la rame 'E'.

5.4.6 Prise en charge d'autres schémas d'URI

Hormis le schéma d'URI TCN défini qui s'adresse principalement à l'adressage fonctionnel à l'intérieur des trains, d'autres schémas d'URI comme les localisateurs de ressources uniformes (URL) communément employés avec la syntaxe²

```
scheme://domain:port/path?....."
```

peuvent également être utilisés. Le 'domaine' correspond à la section hôte de l'URI TCN et peut être résolu par le service DNS TCN. De cette façon, des schémas d'URI autres que celui de l'URI TCN peuvent être pris en charge, ce qui peut être nécessaire pour des applications OMTS spécifiques.

EXEMPLES

Demande HTTP: `http://devECSP.anyVeh.ICst.ITrn:80`

Service de messagerie: `mailto:bob@devECSP.anyVeh.ICst.ITrn`

5.5 Serveur TCN-DNS

5.5.1 Généralités

Les URI TCN section hôte sont résolus en adresses IP avec l'aide du DNS TCN comme défini en 5.4.2. L'interaction entre clients DNS (le "résolveur") et le serveur DNS local est spécifiée dans la RFC 1034 et la RFC 1035.

Une interface plus efficace avec le serveur DNS local plus particulièrement adaptée à la résolution de plusieurs URI TCN est décrite à l'Annexe E.

5.5.2 Architecture

Chaque rame doit fournir au moins un serveur de noms DNS ayant autorité pour ce domaine de rame.

NOTE Le domaine de la rame peut être subdivisé en plusieurs zones où chaque zone possède son propre serveur de noms DNS. Dans le cas particulier d'une rame comportant un réseau opérationnel et multimédia distincts, deux serveurs de noms DNS dédiés (l'un pour le réseau opérationnel et l'autre pour le réseau multimédia) peuvent être fournis.

5.5.3 Résolution d'adresse fonctionnelle

Le serveur DNS de la rame doit pouvoir résoudre les sections hôte des URI TCN (telles que définies en 5.4.4) en adresses IP correspondantes (telles que définies dans l'IEC 61375-2-5).

² Une URL est, d'une manière générale, un type d'identificateur de ressource uniforme (URI) tel que défini dans la RFC 3986.

EXEMPLES

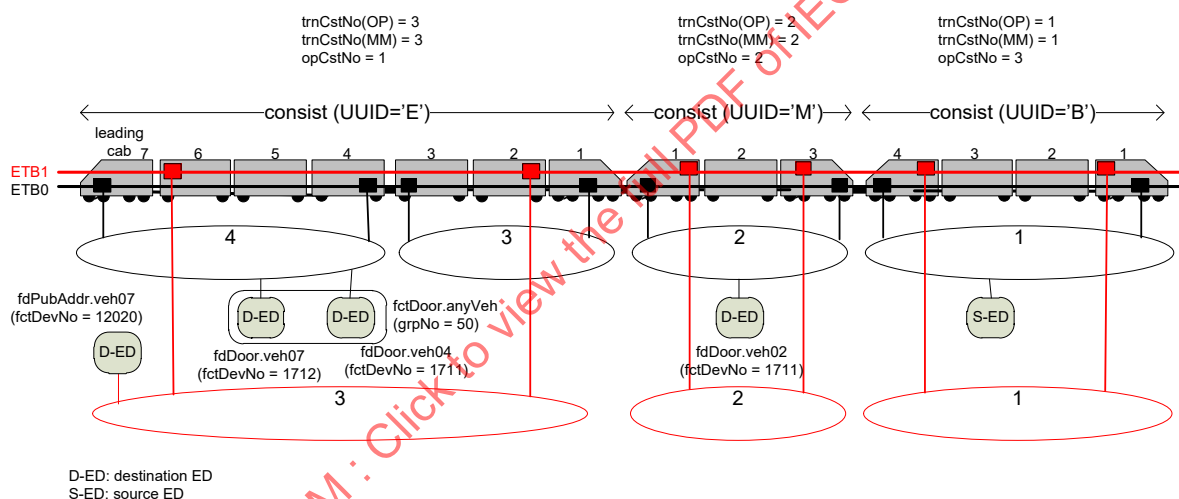
Supposons un train comportant deux réseaux centraux de train, l'ETB0 (réseau opérationnel 'OP') et l'ETB1 (réseau multimédia 'MM'), composé de trois rames, comme illustré à la Figure 21. Les directions de référence TCN sont identiques pour l'ETB0 et l'ETB1. En raison de la position de la cabine de tête, les numéros de rame opérationnels sont opposés aux numéros de rame TCN de l'ETB0. La rame identifiée par l'UUID='E' a deux ECN connectés à l'ETB0 et un ECN connecté à l'ETB1. Les autres rames ont seulement un ECN pour chaque ETB. En outre, il est supposé qu'il existe deux serveurs de noms DNS, un pour le réseau opérationnel et un pour le réseau multimédia, et que les appareils terminaux connectés à l'un des deux réseaux utilisent le serveur de noms DNS associé pour la résolution d'adresse. Il existe une TTDB pour chaque ETB avec des informations partiellement partagées suivant les règles définies en 5.3.4.

La tâche consistera pour un appareil terminal source situé dans la rame identifiée par l'UUID = 'B' et connectée au réseau opérationnel à résoudre les URI TCN des deux appareils terminaux destinataires indiqués:

- appareil terminal doorCTRL@fdDoor.veh02.cst02.anyCITrn.ITrn dans la rame identifiée par l'UUID = 'M'
- appareil terminal publicAnnounce@fdPubAddr.anyVeh.leadCst. anyCITrn.ITrn dans la rame identifiée par l'UUID = 'E' (véhicule de tête)

mais également à résoudre les adresses de groupes du groupe contrôleur de porte (illustré seulement pour la rame 'E' à la Figure 21), en tant que groupe couvrant l'ensemble du train et groupe limité à une rame:

- groupe doorCTRL@fctDoor.anyVeh.aCst.aCITrn.ITrn dans le train
- groupe doorCTRL@fctDoor.anyVeh.cst03.anyCITrn.ITrn dans la rame 'E'



IEC

Anglais	Français
consist	rame
leading cab	cabine de tête

Figure 21 – Résolution d'URI TCN dans un train

EXEMPLE 1

Pour résoudre l'URI TCN 'doorCTRL@fdDoor.veh02.cst02.anyCITrn.ITrn', le serveur DNS de l'ETB0 doit exécuter les étapes répertoriées dans le Tableau 23:

Tableau 23 – Résolution d'URI TCN – Exemple 1

Etape	Action	Résultat (exemple)
1	Isoler la section hôte de l'URI TCN et détacher les sections inutiles.	= "fdDoor.veh02.cst02"
2	Rechercher 'cstUUID' de l'étiquette 'cstNo' = 2 dans le répertoire de train de la TTDB locale.	= "M"
3	Rechercher 'cstUUID' dans la ConsistInfoList.	Enregistrement CONSIST_INFO localisé pour rame "M"

Etape	Action	Résultat (exemple)
4	Rechercher ['fctName' = "fdDoor"] conjointement avec [cstVehNo = 02] dans le groupement d'informations de fonction de l'enregistrement localisé CONSIST_INFO et récupérer la valeur des paramètres 'fctId', 'etbNo' et 'cnNo'.	fctId = 1711 etbNo = 0 cnNo = 1
5	La valeur de 'subnet id' peut être déduite à partir du répertoire de réseau de train associé à l'ETB0.	subnet id = 2
6	Calculer l'adresse IP à partir des valeurs des paramètres 'etbNo', 'subnet id' et 'fctId' conformément à l'IEC 61375-2-5.	IP = 10.128.134.175

EXEMPLE 2

Pour résoudre l'URI TCN 'publicAnnounce@fdPubAddr.anyVeh.leadCst.ITrn', le serveur DNS de l'ETB0 doit exécuter les étapes répertoriées dans le Tableau 24:

Tableau 24 – Résolution d'URI TCN – Exemple 2

Etape	Action	Résultat (exemple)
1	Isoler la section hôte de l'URI TCN et détacher les sections inutiles.	= "fdPubAddr.anyVeh.leadCst"
2	Rechercher le véhicule de tête dans le répertoire opérationnel de train et récupérer les valeurs des paramètres 'opVehNo' et 'ownCstNo'.	opVehNo = 1 ownOpCstNo = 1
3	Chercher 'cstUUID' de 'opCstNo' = 1 dans le répertoire opérationnel de train.	= "E"
4	Déterminer l'orientation de la rame par rapport à l'orientation opérationnelle du train en lisant le paramètre 'opCstOrient' dans le répertoire opérationnel de train.	= inverse
5	Rechercher 'cstUUID' dans la ConsistInfoList.	Enregistrement CONSIST_INFO localisé pour rame "E"
6	Déterminer le nombre de véhicules dans la rame considérée (paramètre 'vehCnt') et calculer le 'cstVehNo' du véhicule de tête.	Nombre de véhicules = 7. La rame étant opposée à la direction opérationnelle du train, la valeur 'cstVehNo' du véhicule de tête identifié par opVehNo = 1 est cstVehNo = 7.
7	Rechercher ['fctdevName' = "fdPubAddr" AND 'cstVehNo = 7] dans le groupement d'informations de fonction de l'enregistrement localisé CONSIST_INFO et récupérer la valeur des paramètres 'fctId', 'etbNo' et 'cnNo'. NOTE La fonction fdPubAddr est associée au réseau multimédia. Pour la rendre visible dans la TTDB de l'ETB0, cette fonction figure à la fois dans l'enregistrement CONSIST_INFO de la TTDB de l'ETB0 et dans l'enregistrement CONSIST_INFO de la TTDB de l'ETB1 (voir 5.3.4.4).	fctId = 12020 etbNo = 1 cnNo = 1
8	La valeur de 'subnet id' peut être déduite à partir du répertoire de réseau de train associé à l'ETB1.	subnet id = 3
9	Calculer l'adresse IP à partir des valeurs des paramètres 'etbNo', 'subnet id' et 'fctId' conformément à l'IEC 61375-2-5.	IP = 10.160.238.244

EXEMPLE 3

L'exemple suivant illustre la manière dont un URI TCN peut être résolu en adresse de groupe de multidiffusion couvrant l'ensemble du train. Ici, il est supposé que le groupe de multidiffusion IP 'fctDoor' a été défini comme un groupe couvrant l'ensemble du train identifié par l'ID groupe = 50. Pour résoudre l'URI TCN 'doorCTRL@fctDoor.anyVeh.aCst.aCITrn.ITrn', le serveur DNS de l'ETB0 doit exécuter les étapes répertoriées dans le Tableau 25.

Noter que les adresses de groupes de multidiffusion couvrant l'ensemble du train doivent être uniques à l'échelle du train.

Tableau 25 – Résolution d'URI TCN – Exemple 3

Etape	Action	Résultat (exemple)
1	Isoler la section hôte de l'URI TCN et détacher les sections inutiles.	= "fctDoor"
2	Rechercher la première occurrence de ['fctName' = "fctDoor"] dans la liste d'informations de rame en faisant des itérations dans les enregistrements CONSIST_INFO et récupérer la valeur des paramètres 'fctId'.	fctId = 50
3	Calculer l'adresse IP à partir de la valeur de paramètre 'fctId' conformément au schéma défini ci-dessus.	IP = 239.193.0.50

EXEMPLE 4

L'exemple suivant illustre la manière dont un URI TCN peut être résolu en adresse de groupe de multidiffusion limité à une rame. Ici, il est supposé que les deux contrôleurs de porte de la rame E forment un groupe de multidiffusion identifié par l'ID groupe = 50. Pour résoudre l'URI TCN 'doorCTRL@fctDoor.anyVeh.cst03.anyCITrn.ITrn', le serveur DNS de l'ETB0 doit exécuter les étapes répertoriées dans le Tableau 26:

Tableau 26 – Résolution d'URI TCN – Exemple 4

Etape	Action	Résultat (exemple)
1	Isoler la section hôte de l'URI TCN et détacher les sections inutiles.	= "fctDoor.anyVeh.cst03"
2	Rechercher 'cstUUID' de l'étiquette 'cstNo' = 3 dans le répertoire de train de la TTDB locale.	= "E"
3	Rechercher 'cstUUID' dans la ConsistInfoList.	Enregistrement CONSIST_INFO localisé pour rame "E"
4	Rechercher ['fctName' = "fdDoor"] dans le groupement d'informations de fonction de l'enregistrement localisé CONSIST_INFO et récupérer la valeur du paramètre 'fctId'.	fctId = 50
5	Calculer l'adresse IP à partir des valeurs des paramètres 'trnCstNo' et 'fctId' conformément au schéma défini ci-dessus.	IP = 239.194.1.50

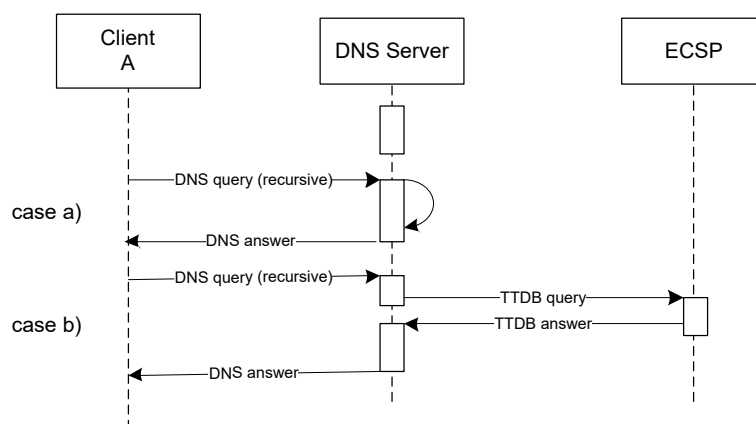
5.5.4 Protocole

Le serveur de noms DNS de la rame doit prendre en charge les protocoles DNS comme spécifié aux RFC 1034 et RFC 1035.

Le serveur de noms DNS de la rame doit accepter les requêtes récursives de clients comme illustré à l'exemple de la Figure 22.

Les URI TCN adressant les destinataires au sein des rames locales ou distantes doivent être résolus dans un délai de 1,0 s.

NOTE Le temps est mesuré de la réception de la requête jusqu'à la fin de l'envoi de la réponse au client.



Anglais	Français
DNS Server	Serveur DNS
DNS query (recursive)	requête DNS (récursive)
DNS answer	Réponse DNS
case a)	cas a)
case b)	cas b)
TTDB query	requête TTDB
TTDB answer	réponse TTDB

Figure 22 – Protocole DNS (cas a sans, cas b avec interrogation TTDB)

5.5.5 ETB multiples

Le mapping d'URI TCN en adresses IP est spécifique à l'ETB pour les communications inter-rammes, car cela dépend du répertoire de réseau de train de l'ETB spécifique. Par conséquent, un serveur DNS TCN doit être fourni pour chaque ETB.

Une fonction doit utiliser le serveur DNS TCN pour la résolution d'adresse URI TCN, qui est associé à l'ETB avec lequel les fonctions souhaitent communiquer.

NOTE L'adressage d'une fonction dans une rame différente peut donner lieu à des adresses IP différentes selon l'ETB qui est utilisé aux fins de communication.

S'assurer que le chemin de retour d'une communication inter-rammes utilisera le même ETB si, pour les télégrammes retournés, l'adresse IP de l'émetteur d'origine est utilisée comme adresse de destination.

5.6 Echange de données

5.6.1 Généralités

Le présent paragraphe définit l'échange de données entre les fonctions et les appareils sur l'ETB. Selon les fonctions impliquées, différents protocoles de communication pourraient être appliqués. La présente norme ne définit que les protocoles de communication utilisés par les services qui y sont spécifiés.

5.6.2 Réseau opérationnel de communication

Tous les services du réseau opérationnel doivent utiliser le protocole TRDP tel qu'il est spécifié à l'Annexe A (selon les informations supplémentaires données à l'Annexe C et à

l'Annexe D) pour le transfert de données de processus et de données de messagerie entre les fonctions du réseau opérationnel.

Pour le transfert de flux de données (audio/vidéo), des protocoles appropriés aux flux de données peuvent être utilisés, par exemple (liste non exhaustive):

RTSP (Real Time Streaming Protocol)	Protocole de flux en temps réel	RFC 2326
RTCP (Real Time Control Protocol)	Protocole de contrôle en temps réel	RFC 3605
RTP (Real Time Transport)	Protocole de transport en temps réel	RFC 3550
SRTP (Secure Real Time Transport)	Protocole de transport en temps réel sécurisé	RFC 3261
SIP (Session Initiation Protocol)	Protocole d'ouverture de session	RFC 3711
TLS (Transport Layer Security)	Protocole de sécurité de la couche transport	RFC 5246

Pour les services non opérationnels tels que le transfert de fichiers ou l'accès aux services, des protocoles de données appropriés peuvent être utilisés, par exemple (liste non exhaustive):

FTP (File Transfer Protocol)	Protocole de transfert de fichiers	RFC 956
SFTP (SSH [Secure] File Transfer Protocol)	Protocole de transfert de fichiers (sécurisé)	SSH divers RFC
SNMP (Simple Network Management Protocol)	Protocole simple de gestion de réseau	divers RFC
SSH	(Secure Shell)	divers RFC
SCP (Secure Copy Protocol)	Protocole de copie sécurisé	-

5.6.3 Réseau OMTS de communication

En principe, les mêmes protocoles que ceux qui sont utilisés dans le réseau opérationnel peuvent également être utilisés pour le réseau OMTS. En raison de la nature particulière du réseau OMTS en tant que plate-forme pour les services multimédias et télématiques, des protocoles supplémentaires peuvent être utilisés, par exemple (liste non exhaustive):

HTTP	Protocole de transfert hypertexte	RFC 2616
SOAP	Protocole simple d'accès aux objets	World Wide Web Consortium (W3C)

5.6.4 Qualité de service (QoS)

L'IEC 61375-1 définit un ensemble de classes de données devant être prises en charge par l'ETB tandis que l'IEC 61375-2-5 définit un minimum de quatre priorités pour le transfert de données via l'ETB. Sauf définition contraire, l'affectation des priorités aux classes de données doit être réalisée conformément au Tableau 27.

Tableau 27 – Priorités des classes de données

Classe de données	Priorité	Remarques
Données de processus	3	
Données de messagerie	0 .. 1	
Données de flux	1 .. 2	Le choix des priorités peut dépendre du type de flux, par exemple un flux audio pourrait exiger une priorité supérieure à celle d'un flux vidéo.
Données du mode "au mieux"	0	
Données de supervision	3	Cette classe de données est uniquement définie dans l'IEC 61375-2-5.
3 = priorité maximale, 0 = priorité minimale		

5.7 Découverte de service

Chacune des rames doit annoncer les fonctions qu'elle prend en charge à l'intérieur de son télégramme CSTINFO. Toutes les fonctions prises en charge peuvent ensuite être récupérées à partir de la TTDB.

5.8 Service d'informations de train

Le service d'informations de train définit un service interne à la rame permettant de récupérer des données de la TTDB. Il ne relève pas du domaine d'application de la présente norme de définir les services internes à la rame, mais un ensemble utile de primitives de fonction est donné à l'Annexe E pour information.

6 Services du profil de communication – Service de contrôle ETB

6.1 Généralités

Le présent paragraphe spécifie les fonctions de service nécessaires d'une manière générale pour permettre le transfert des données utilisateur sur l'ETB. Ces fonctions sont les suivantes:

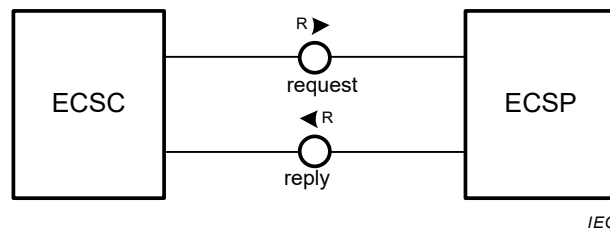
- Fonction véhicule de tête
- Fonction mode sommeil
- Fonction confirmation/correction

En cas d'ETB multiples, par exemple un réseau opérationnel et un réseau multimédia, ces fonctions doivent seulement être activées sur le réseau opérationnel.

NOTE Cette restriction évite les commandes contradictoires.

6.2 Modèle de communication

Les fonctions de contrôle ETB suivent le modèle client-serveur tel qu'illustré à la Figure 23. Le client du service de contrôle ETB (ECSC) demande une primitive de fonction au fournisseur du service de contrôle ETB (ECSP) tandis que l'ECSP répond en retournant le résultat d'exécution de la fonction.



Anglais	Français
request	demande
reply	réponse

Figure 23 – Modèle de service de contrôle ETB

Seul un ECSP à la fois doit être actif au sein d'une rame.

Il convient de mettre en œuvre plusieurs ECSP, combinés par exemple avec un ETBN, au sein d'une rame et de sélectionner l'un des ECSP de la rame pour devenir l'ECSP actif.

La présente partie de la norme ne fait aucune hypothèse en ce qui concerne le client de ce service. Pour la spécification du service, un client abstrait (ECSC) est modélisé. Il invoque les fonctions de service et récupère également les résultats des fonctions.

En général, il convient d'avoir un seul appareil au sein de chaque rame, dédié en tant que client de service de contrôle ETB afin d'éviter les conflits d'utilisation du serveur, par exemple un appareil demandant une certaine correction de la composition du train et un autre appareil demandant une autre correction.

6.3 Supervision ECSP

L'ECSP doit superviser la disponibilité de l'ECSC.

Une défaillance de l'ECSC doit être détectée au plus tard après une période de $T_{ECSC\ fail} = 5,0$ s. Si une défaillance est détectée, l'ECSP doit réagir tel que défini pour les fonctions de service de contrôle ETB individuelles.

NOTE L'Annexe E définit une interface entre l'ECSC et l'ECSP pour les services spécifiés dans le présent document.

6.4 Interconnexion d'ECSP

6.4.1 Généralités

Toutes les fonctions de contrôle de l'ETB exigent un échange de données entre tous les ECSP d'un train. Il s'agit de l'ensemble des ECSP qui sont associés à une rame figurant dans le répertoire de train avec un cstTopoCnt valide.

Pour l'échange des données entre les ECSP, chaque ECSP envoie cycliquement un télégramme ETBCTRL à tous les ECSP au sein du train, comme le montre la Figure 24. L'échange de données peut être "normal" pour une application non liée à la sécurité ou "de sécurité" pour les applications liées à la sécurité.

En cas d'ETB multiples, par exemple un réseau opérationnel et un réseau multimédia, les télégrammes ETBCTRL doivent seulement être échangés sur le réseau opérationnel (ETB0).

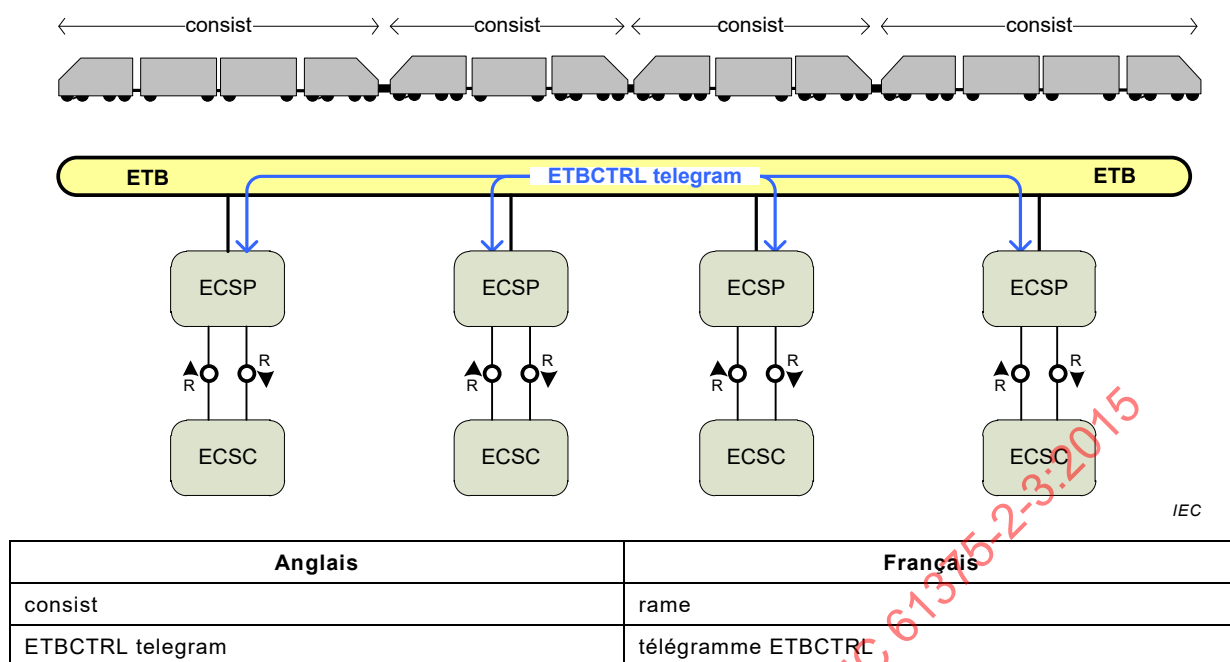


Figure 24 – Echange de télégrammes ETBCTRL

6.4.2 Sélection d'échange de télégrammes ETBCTRL

Un ECSP doit prendre en charge soit la transmission normale de télégrammes ETBCTRL soit la transmission sécurisée de télégrammes ETBCTRL en utilisant le protocole SDTv2 comme défini à l'Annexe B.

Un ECSP mettant en œuvre la transmission sécurisée de télégrammes ETBCTRL peut également traiter les télégrammes ETBCTRL normaux reçus.

6.4.3 Transmission de télégramme ETBCTRL

Paramètres du télégramme PD TRDP:

Type de message:	'Pd'
ComId:	1
SMI:	1 (nécessaire uniquement si SDTv2 est utilisé)
UserDataVersion:	'0100'H (nécessaire uniquement si SDTv2 est utilisé)
DestinationURI (section hôte):	grpECSP.anyVeh.aCst.aCITrn.ITrn
Dataset:	ETBCTRL_TELEGRAM
Durée de cycle:	(0,5 ± 0,1) s
Délai d'expiration:	3,0 s

6.4.4 Structure du télégramme ETBCTRL

La structure du paquet ETBCTRL est montrée à la Figure 25.

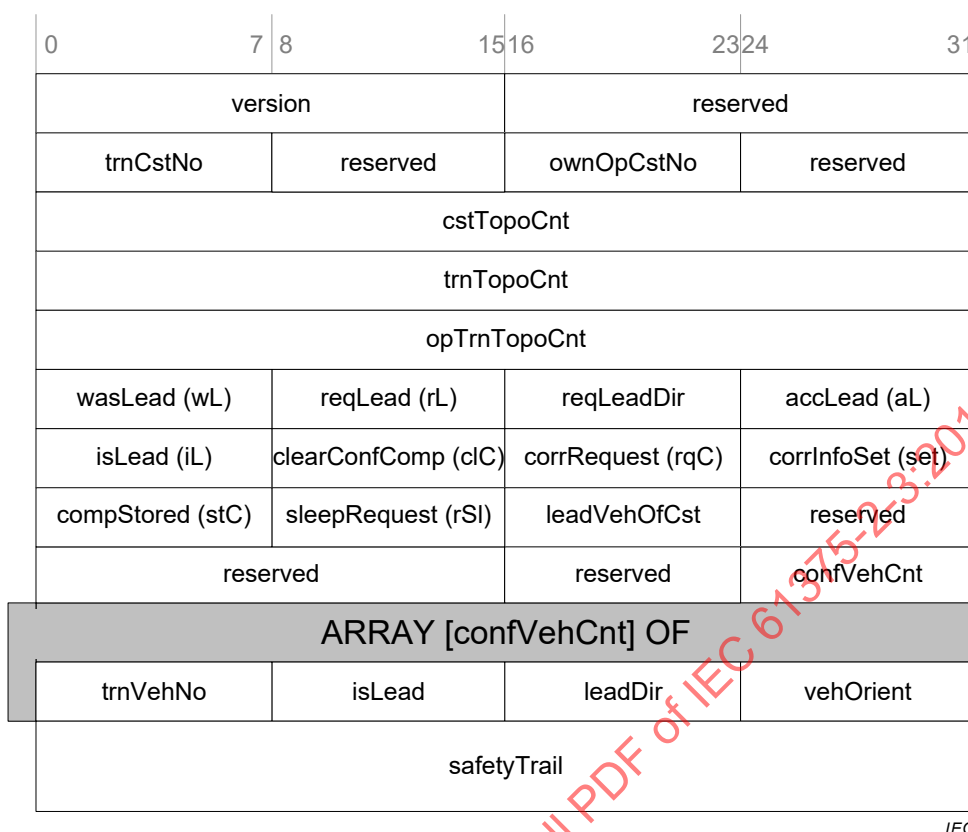


Figure 25 – Télégramme CSTINFOCTRL

```

CONFIRMED_VEHICLE_ ::= RECORD
{
    trnVehNo      UINT8      -- vehicle sequence number within the train
                           with vehicle 01 being the first vehicle
                           in ETB reference direction 1 as defined in
                           IEC 61375-2-5
                           value range: 0..63
                           a value of 0 indicates that this vehicle
                           has been inserted by correction
    isLead        ANTI-VALENT8 -- vehicle is leading, informal
                           may be used for plausibility checks
    leadDir       UINT8      -- vehicle leading direction, informal
                           0 = not relevant
                           1 = leading direction 1
                           2 = leading direction 2
                           may be used for plausibility checks
    vehOrient     UINT8      -- vehicle orientation, informal
                           0 = not known (corrected vehicle)
                           1 = same as operational train direction
                           2 = inverse to operational train
                           direction
                           may be used for plausibility checks
}
    
```

Les paramètres informels figurant dans le télégramme peuvent être utilisés à des fins de comparaison entre la liste de véhicules considérée et la liste de véhicules indiquée dans le cadre d'un contrôle de vraisemblance en plus du contrôle de valeur opTopoCnt.

ETBCTRL_VDP ::= RECORD

```
{
  reserved01      UINT32    -- reserved, shall be set to 0
  reserved02      UINT16    -- reserved, shall be set to 0
  userDataVersion VERSION   -- version of the vital ETBCTRL telegram
                                mainVersion = 1
                                subVersion = 0
  safeSequCount   UINT32    -- safe sequence counter, as defined in B.9
  safetyCode      UINT32    -- checksum, as defined in B.9
}
```

ETBCTRL_TELEGRAM ::= RECORD

```
{
  version          VERSION   -- data structure version
                                parameter 'mainVersion'
                                shall be set to 1.
  reserved01       UINT16    -- reserved (=0)
  trnCstNo         UINT8     -- own train consist number
                                value range: 1..63
  reserved02       UINT8     -- rsv: reserved (=0)
  ownOpCstNo       UINT8     -- own operational number
                                value range: 1..63
                                0 = unknown (e.g. after inauguration)
  reserved03       UINT8     -- rsv: reserved (=0)
  cstTopoCnt       UINT32    -- consist topography counter
  trnTopoCnt       UINT32    -- train directory topography counter
  opTrnTopoCnt     UINT32    -- operational Train directory topography
                                counter
  wasLead          ANTIVALENT8 -- wL: consist was leading control flag
                                '01'B = FALSE
                                '10'B = TRUE
  reqLead          ANTIVALENT8 -- rL: leading request control flag
                                '01'B = FALSE
                                '10'B = TRUE
  reqLeadDir       UINT8     -- rLD: (request) leading direction
                                0 = not relevant
                                1 = consist direction 1
                                2 = consist direction 2
  accLead          ANTIVALENT8 -- aL: accept remote leading request
                                control flag
                                '01'B = FALSE, not accepted
                                '10'B = TRUE, accepted
  isLead           ANTIVALENT8 -- iL: consist contains leading vehicle
                                control flag
                                '01'B = FALSE
                                '10'B = TRUE
  clearConfComp    ANTIVALENT8 -- clC: clear confirmed composition
                                control flag
                                '01'B = 0:FALSE
                                '10'B = 1:TRUE
  corrRequest      ANTIVALENT8 -- rqC: request confirmation control flag
                                '01'B = 0:FALSE
                                '10'B = 1:TRUE
  corrInfoSet      ANTIVALENT8 -- set: correction info set control flag
```

```

'01'B = 0:FALSE
'10'B = 1:TRUE
compStored      ANTIVALENT8  -- stC: corrected composition stored
                                control flag
                                '01'B = 0:FALSE
                                '10'B = 1:TRUE
sleepRequest     ANTIVALENT8  -- rSl: request sleep mode control flag
                                '01'B = FALSE
                                '10'B = TRUE
leadVehOfCst     UINT8        -- position of leading vehicle in consist
                                value range 0..32
                                0 = not defined
                                (1 = first vehicle in consist in
                                Direction 1, 2 = second vehicle, etc.)
reserved04       UINT8        -- rsv: reserved (=0)
reserved05       UINT16       -- rsv: reserved (=0)
reserved06       UINT8        -- rsv: reserved (=0)
confVehCnt       UINT8        -- number of confirmed vehicles in train
                                value range: 0..63,
                                0 = no correction requested
confVehList      ARRAY [confVehCnt] OF CONFIRMED_VEHICLE
                                -- dynamic ordered list
                                of confirmed vehicles in train,
                                starting with the vehicle at
                                train extremity 1 (see 4.2.4.2)
safetyTrail      ETBCTRL_VDP  -- ETB-VDP trailer
                                SafeTopoCount = 0
                                completely set to 0 = SDTv2 not used
}

```

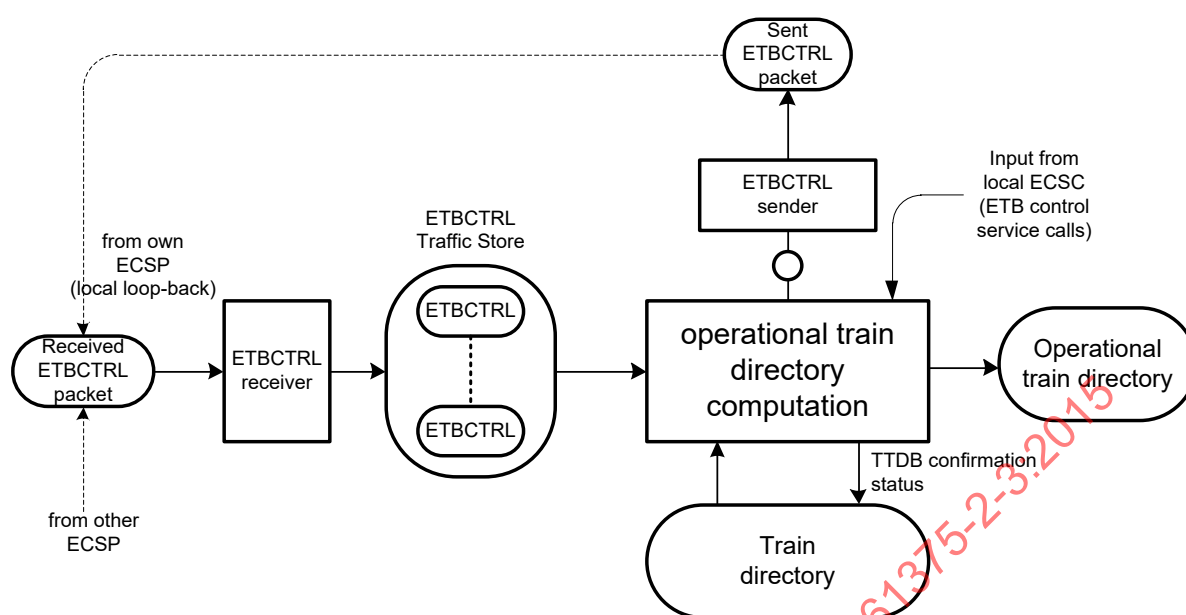
NOTE Le paramètre 'reqLeadDir' n'a de sens que si l'un des paramètres 'reqLead', 'wasLead' ou 'isLead' est défini sur TRUE. Ce paramètre est pertinent pour la détermination de la direction opérationnelle du train.

6.4.5 Processus de calcul du répertoire opérationnel de train

6.4.5.1 Présentation

Les fonctions de service de contrôle ETB "tête" et "confirmation/correction" déterminent le contenu du répertoire opérationnel de train. Le calcul du répertoire opérationnel de train est une fonction de chaque ECSP connecté à l'ETB0, qui est chargé de créer et de maintenir le répertoire opérationnel de train tel que défini en 5.3.3.

La principale opération du calcul du répertoire opérationnel de train est illustrée à la Figure 26.



IEC

Anglais	Français
Sent ETBCTRL packet	Paquet ETBCTRL envoyé
ETBCTRL sender	émetteur de l'ETBCTRL
from own ECSP (local loop-back)	provenant de l'ECSP considéré (rebouclage local)
Received ETBCTRL packet	Paquet ETBCTRL reçu
from other ECSP	provenant d'autres ECSP
ETBCTRL receiver	destinataire de l'ETBCTRL
ETBCTRL Traffic Store	Traffic Store de l'ETBCTRL
operational train directory computation	calcul du répertoire opérationnel de train
Train directory	Répertoire de train
TTDB confirmation status	Etat de confirmation de la TTBD
Input from local ECSC (ETB control service calls)	Saisie par l'ECSC local (appels du service de contrôle ETB)
Operational train directory	Répertoire opérationnel de train

Figure 26 – Diagramme d'états de calcul du répertoire de train

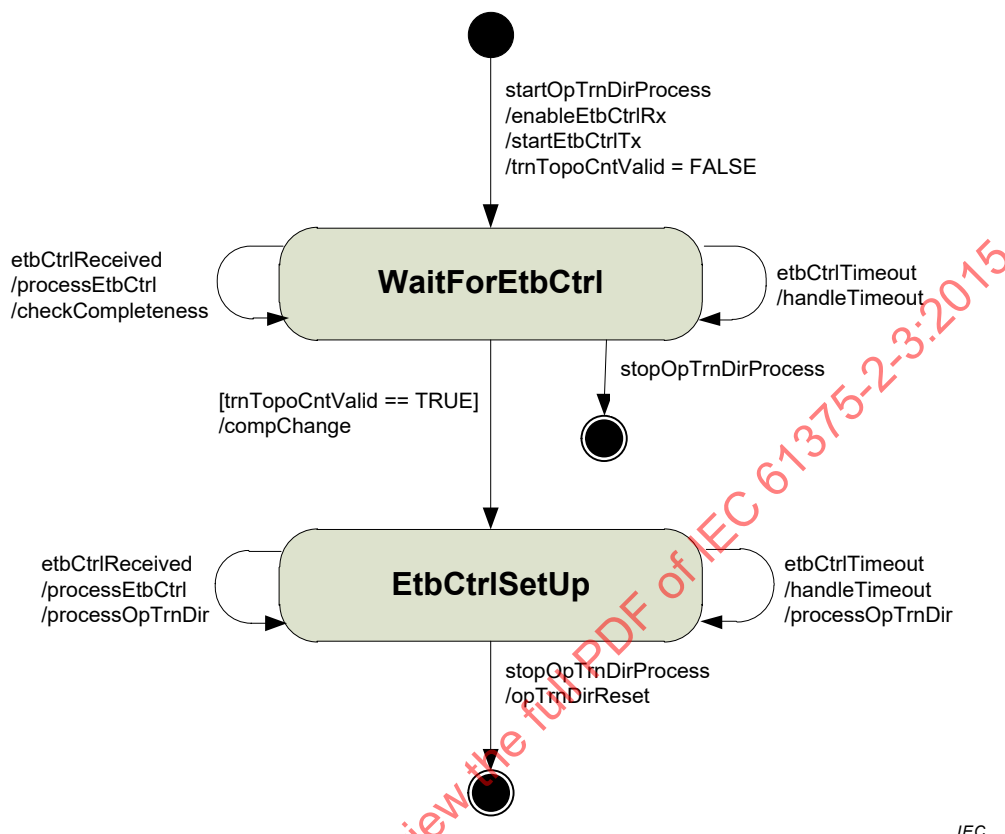
Le calcul du répertoire opérationnel de train se base sur la collecte des télégrammes ETBCTRL reçus par l'ensemble des ECSP. Cela inclut l'ECSP considéré (local), mais également les ECSP distants (à savoir tous les ECSP sauf l'ECSP considéré).

Les télégrammes ETBCTRL pourraient par exemple être stockés dans un Traffic Store (Mémoire Partagée) ETBCTRL à des fins de traitement ultérieur. Le calcul du répertoire opérationnel de train vérifie le contenu du Traffic Store ETBCTRL par rapport aux informations du véhicule de tête, aux informations de correction et à la valeur trnTopoCnt.

Si un changement est attendu, un nouveau répertoire opérationnel de train doit être calculé conformément aux règles énoncées ultérieurement dans le présent paragraphe et le statut de confirmation de la TTDB doit être actualisé (si nécessaire). De plus, une entrée actualisée au télégramme ETBCTRL envoyé par l'ECSP local doit être fournie.

6.4.5.2 Diagramme d'états

Le processus de calcul du répertoire opérationnel de train est défini par le diagramme d'états présenté à la Figure 27 et décrit dans le Tableau 28 jusqu'au Tableau 30.



IEC

Figure 27 – Diagramme d'états de traitement ETBCTRL

Tableau 28 – Traitement ETBCTRL: triggers

Trigger	Description
startOpTrnDirProcess	Lancer le traitement du répertoire opérationnel de train, déclenché par le diagramme d'états de calcul du répertoire de train (voir 5.3.2).
stopOpTrnDirProcess	Lancer le traitement du répertoire opérationnel de train, déclenché par le diagramme d'états de calcul du répertoire de train (voir 5.3.2.). Si la transmission du télégramme ETBCTRL se poursuit, le paramètre de télégramme 'trnTopoCnt' doit être défini sur 0 jusqu'au redémarrage du traitement du répertoire opérationnel de train (trigger 'startOpTrnDirProcess')
etbCtrlReceived	Un télégramme ETBCTRL a été reçu.
etbCtrlTimeOut	Un télégramme ETBCTRL a expiré.

Tableau 29 – Traitement ETBCTRL: guards

Guard	Description
[trnTopoCntValid == TRUE]	Des télégrammes ETBCTRL ont été reçus en provenance de tous les ECSP actifs répertoriés dans le répertoire de train local avec un paramètre cstTopoCnt != 0 et tous les télégrammes ETBCTRL valides reçus (y compris le télégramme ETBCTRL considéré) ont une valeur trnTopoCnt identique à la valeur trnTopoCnt stockée dans le répertoire de train local.

Tableau 30 – Traitement ETBCTRL: actions

Action	Description
opTrnDirReset	Le répertoire opérationnel de train doit être redéfini tel qu'indiqué en 6.7.3 (événement trigger 'opTrnDirReset').
enableEtbCtrlRx	Préparer la réception de télégrammes ETBCTRL en provenance de tous les ECSP de rame répertoriés dans le répertoire de train avec une valeur cstTopoCnt valide et établir une supervision des délais d'expiration individuels pour chacun de ces ECSP de rame. Définir le délai d'expiration de réception sur une valeur de $(3,0 \pm 0,1)$ s. Les minuteurs doivent être activés après la réception du premier télégramme ETBCTRL associé.
startEtbCtrlTx	Lancer l'envoi cyclique de télégrammes ETBCTRL Durée de cycle: $(0,5 \pm 0,1)$ s Une fois que la transmission de télégrammes ETBCTRL a commencé, elle peut se poursuivre indéfiniment, c'est-à-dire qu'elle peut ne pas s'arrêter en cas d'activation du trigger 'stopOpTrnDirProcess'.
processEtbCtrl	Les télégrammes ETBCTRL reçus doivent être traités comme suit: 1) Les télégrammes ETBCTRL en provenance d'un ECSP ne figurant pas dans le répertoire de train (valeur cstTopoCnt inconnue) ou figurant avec une valeur cstTopoCnt = 0 doivent être supprimés. 2) Les télégrammes ETBCTRL avec une valeur de paramètre 'trnTopoCnt' différente de la valeur du paramètre 'trnTopoCnt' stockée dans le répertoire de train local doivent être supprimés. 3) Les balises de contrôle et autres paramètres pertinents provenant des télégrammes ETBCTRL non supprimés doivent être mémorisés, autrement dit stockés dans le Traffic Store.
checkCompleteness	Le paramètre trnTopoCntValid doit être défini sur TRUE si des télégrammes ETBCTRL (non supprimés) ont été reçus en provenance de tous les ECSP répertoriés dans le répertoire de train avec un paramètre cstTopoCnt != 0, à l'exception des ECSP marqués comme ayant expiré.
processOpTrnDir	Lancer le processus de calcul du répertoire opérationnel: • Diagramme d'états de la fonction véhicule de tête tel que défini en 6.5 • Diagramme d'états de la fonction confirmation/correction tel que défini en 6.6 • Opérations associées telles que définies en 6.7 Ce processus de calcul peut être exécuté selon l'événement (tel qu'indiqué dans le diagramme d'états) ou de manière cyclique.
handleTimeout	Marquer la rame source associée comme ayant expiré. Définir les balises de contrôle (stockées en local) de l'ECSP source associée sur des valeurs différées (ex: dans le Traffic Store). Les valeurs différées sont définies: • en 6.5 pour la fonction véhicule de tête • en 6.6 pour la fonction confirmation/correction • en 6.8 pour la fonction mode sommeil
compChange	Déclencher un événement 'compChange': • en 6.5 pour la fonction véhicule de tête • en 6.6 pour la fonction confirmation/correction

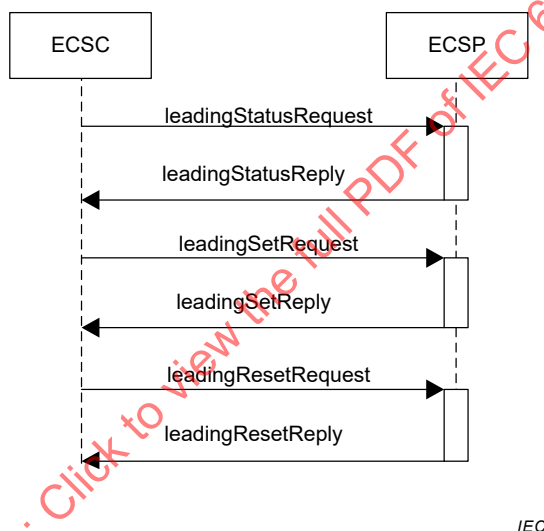
6.5 Fonction "véhicule de tête"

6.5.1 Généralités

Cette fonction permet de sélectionner l'un des véhicules du train pour devenir le véhicule de tête. Le véhicule de tête est, dans le cas général, le véhicule ayant le pupitre de conduite activé. Cette fonction doit assurer qu'à tout moment un seul véhicule au maximum est titulaire de la propriété "de tête". Dans certains cas spéciaux, par exemple après l'accouplement de deux trains, deux véhicules peuvent avoir cette propriété, ce qui conduit à un conflit entre véhicules de tête. Un tel conflit a besoin d'être détecté et signalé à l'utilisateur. Il appartient aux utilisateurs de résoudre le conflit entre véhicules de tête, par exemple en désactivant l'une des cabines de conduite actives.

6.5.2 Primitives de fonction

Les primitives de fonction définies dans le présent paragraphe (du Tableau 31 au Tableau 33) et présentées à la Figure 28 doivent être prises en charge.



IEC

Figure 28 – Diagramme de séquences de la fonction véhicule de tête

Tableau 31 – Primitives de la fonction véhicule de tête: F_leadingStatusRequest

Primitive de fonction:	F_leadingStatusRequest
Description:	Lecture du statut de tête de la rame considérée
Paramètre de demande:	-
Paramètre de réponse:	isLeading (est en tête) = yes/no (oui/non) wasLeading (était en tête) = yes/no (oui/non) leadingDir (direction en tête) = Dir1/Dir2 pendingLeadingRequest (demande de passage en tête en cours) = yes/no (oui/non) leadingConflict (conflit de tête) = yes/no (oui/non) leadingVehicle (véhicule de tête) = n ^{ième} véhicule de la rame de tête

Tableau 32 – Primitives de la fonction véhicule de tête: F_leadingSetRequest

Primitive de fonction:	F_leadingSetRequest
Description:	Demande pour devenir le véhicule de tête. Le paramètre leadingDir indique la principale direction de vision de la cabine de conduite. NOTE Dans les véhicules comportant deux cabines de conduite, il définit également l'extrémité du véhicule où se trouve la cabine de conduite.
Paramètre de demande:	leadingDir (direction en tête) = Dir1/Dir2
Paramètre de réponse:	-

Tableau 33 – Primitives de la fonction véhicule de tête: F_leadingResetRequest

Primitive de fonction:	F_leadingResetRequest
Description:	Demande d'élimination ou élimination de la propriété "véhicule de tête"
Paramètre de demande:	-
Paramètre de réponse:	-

6.5.3 Protocole d'ECSP à ECSP

6.5.3.1 Généralités

Le présent paragraphe définit le protocole du service lié à l'échange de données entre les ECSP homologues.

La fonction véhicule de tête utilise quatre paramètres (balises de contrôle) du télégramme ETBCTRL pour le contrôle (voir Tableau 34).

Tableau 34 – Balises de contrôle de la fonction véhicule de tête

control flag	Forme abrégée	description	Valeur différée
reqLead	rL	demande client ou abandon de la propriété de véhicule de tête	FALSE
accLead (accepter requête de tête à distance)	aL	acceptation d'une demande à distance de passage en tête	inutile
isLead	iL	la rame est en tête	FALSE
wasLead (était en tête)	wL	la rame était en tête précédemment	non modifiée NOTE Sous certaines conditions, cette valeur peut passer à 'FALSE' (voir la note ci-dessous).
opTrnTopoCnt	-	Valeur calculée du compteur de topographie opérationnel du train	inutile

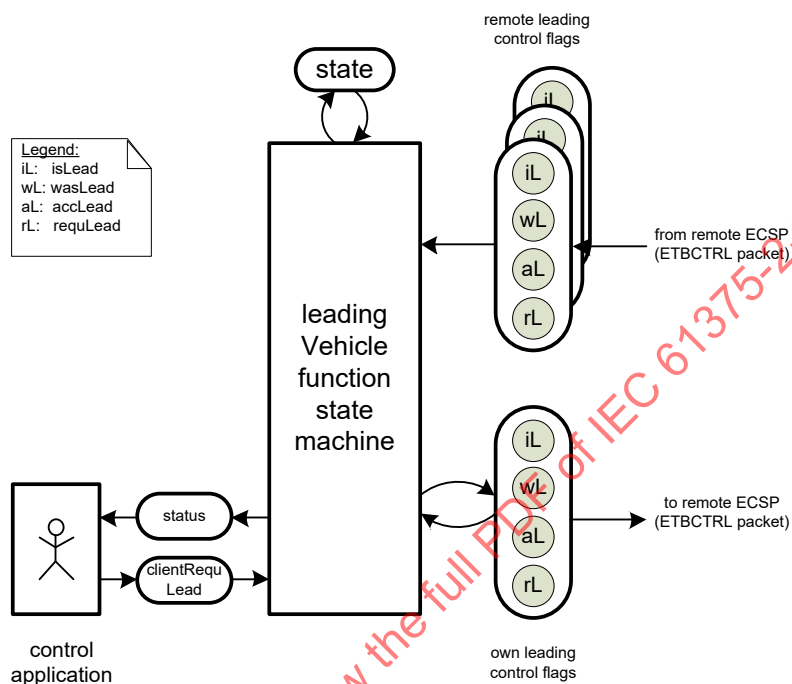
Légende

non modifiée = valeur précédemment stockée à conserver

inutile = cette valeur ne doit pas être utilisée pour le calcul (à ignorer)

NOTE Si les rames de tête arrêtent d'envoyer des télégrammes ECSP, la propriété de véhicule de tête est perdue et un nouveau répertoire opérationnel de train est calculé sans rame de tête. Néanmoins, la direction opérationnelle du train ne sera pas changée vu que la balise de contrôle 'wasLead' n'est pas modifiée. Si un nouveau véhicule de tête est choisi par la suite, cette balise de contrôle est définie sur 'FALSE' (voir 6.5.3.2).

L'état de la fonction véhicule de tête est déterminé par l'ensemble de balises reçues des autres rames (balises de contrôle de véhicule de tête à distance, envoyées par les rames répertoriées dans le répertoire de train avec une valeur $cstTopoCnt \neq 0$) et l'ensemble considéré de balises (balises de contrôle de véhicule de tête considérées) comme le montre la Figure 29.



IEC

Anglais	Français
state	état
Legend: iL: isLead wL: wasLead aL: accLead rL: requLead	Légende: iL: isLead wL: wasLead aL: accLead rL: requLead
remote leading control flags	balises de contrôle du véhicule de tête à distance
leading Vehicle function state machine	diagramme d'états de la fonction véhicule de tête
control application	application de contrôle
status	statut
from remote ECSP (ETBCTRL packet)	depuis l'ECSP distant (paquet ETBCTRL)
to remote ECSP (ETBCTRL packet)	vers l'ECSP distant (paquet ETBCTRL)
own leading control flags	balises de contrôle de véhicule de tête considérées

Figure 29 – Diagramme de blocs du diagramme d'états de la fonction véhicule de tête

Toutes ces balises sont échangées cycliquement à l'aide du télégramme ETBCTRL, mais il doit être considéré que, en raison de l'échange asynchrone des télégrammes ETBCTRL, l'ensemble complet de balises de contrôle pourrait ne pas être cohérent durant un certain temps, ce qui conduit à des effets transitoires.

Localement, le diagramme d'états de la fonction véhicule de tête est déclenché par l'application de contrôle ECSP qui envoie des demandes de passage en tête au diagramme d'états. Le diagramme d'états, en retour, envoie des informations sur l'état réel à l'application de contrôle. Il existe une information spéciale formée de l'indication d'un conflit de tête pouvant se produire si plus d'une rame est en tête en même temps ou si, par exemple après un accouplement, deux véhicules de tête coexistent.

6.5.3.2 Diagramme d'états

Le diagramme d'états montré à la Figure 30 définit le comportement d'un ECSP du point de vue de la fonction véhicule de tête. Les triggers, les guards et les actions sont définis du Tableau 35 au Tableau 37.

IECNORM.COM : Click to view the full PDF of IEC 61375-2-3:2015



Figure 30 – Diagramme d'états de la fonction véhicule de tête

NOTE 1 Pour les transitions déclenchées par la réception d'un télégramme ETBCTRL, seuls les guards sont illustrés par souci de lisibilité.

NOTE 2 Il convient que le guard "remIsLead == TRUE" à l'état IsLeading ne se produise jamais parce que, si un autre train avec propriété de véhicule de tête est accouplé, le trigger compChange se déclenchera à cause de l'inauguration. Néanmoins, il est plus sûr de traiter quand même ce guard.

NOTE 3 Un ECSP réinséré dans une composition avec un véhicule de tête transitera directement par l'état WaitForLead (guard remReqLead) pour passer à l'état IsLead (guard remIsLead).

NOTE 4 Si un ECSP inséré tardivement ne peut pas être intégré dans une composition, cela conduira à une modification de la composition (trigger compChange).

Tableau 35 – Fonction véhicule de tête: triggers

Trigger	Description
Initial	Alimentation ou relance d'ECSP
clientReqLead	L'ECSC demande de passer en tête (= TRUE) (primitive de service F_leadingSetrequest) ou demande d'abandonner la position de tête (= FALSE) (primitive de service F_leadingResetRequest).
compChange	La composition du train a changé. Cet événement est déclenché par le diagramme d'états ETBCTRL (voir 6.4.5).
compReset	Réinitialisation de la composition

Tableau 36 – Fonction véhicule de tête: guards

Guards	Description
reqLead	Balise de contrôle de l'ETBCTRL indiquant (si elle est définie) que la rame considérée demande la propriété de véhicule de tête NOTE Cette balise de contrôle est configurée par le trigger clientReqLead.
wasLead (était en tête)	Balise de contrôle de l'ETBCTRL indiquant (si elle est définie) que la rame considérée avait précédemment la propriété de véhicule de tête
isLead	Balise de contrôle de l'ETBCTRL indiquant (si elle est définie) que la rame considérée est en tête
remReqLead	Au moins une rame distante demande à être en tête, ce qui signifie que dans au moins un télégramme ETBCTRL reçu, à l'exception du télégramme ETBCTRL considéré, la balise de contrôle reqLead ('rL') est définie. NOTE Si plusieurs rames demandent d'être en tête, cela pourrait déboucher sur un conflit de tête.
remAccLead	Toutes les rames distantes acceptent la demande de passage en tête considérée. AND logique sur les balises de contrôle accLead de tous les ECSP, à l'exception de l'ECSP considéré et des ECSP marqués comme ayant expiré (voir 6.4.5.2)
remIsLead	Une rame distante est en tête. La balise de contrôle 'isLead' d'un ECSP, à l'exception de l'ECSP considéré, est définie sur TRUE
allOpTrnTopoCnt	Comparer le paramètre 'opTrnTopoCnt' dans tous les télégrammes ETBCTRL (non supprimés et n'ayant pas expiré). EQUAL = valeur identique dans tous les télégrammes

Tableau 37 – Fonction véhicule de tête – actions

Action	Description
computeOpTrnDir	Calculer le répertoire opérationnel de train en tenant compte des informations de correction obtenues par le biais du télégramme ETBCTRL et mettre à jour le paramètre de télégramme ETBCTRL 'opTrnTopoCnt' (voir 6.7.3).
setLeadDir	Définir le paramètre de télégramme ETBCTRL 'reqLeadDir' sur la direction de tête demandée par l'ECSC local.
leadConfl	Balise locale indiquant (si elle est définie) à l'application qu'un conflit de tête a été détecté.
handleDeferred	Au cas où il y a des ECSP marqués comme ayant expiré, redéfinir (= définir sur FALSE) la valeur différée de la balise de contrôle 'wasLead' de l'ECSP correspondant. NOTE Cela signifie que la valeur différée de la balise de contrôle 'wasLead' passe de 'non modifiée' à 'FALSE'.

Le paramètre 'wasLead' sert à empêcher une modification de la direction opérationnelle après un changement de composition lorsque cette rame était en tête avant le changement de composition.

6.5.3.3 Conditions préalables

Avant d'émettre une commande F_leadingSetRequest ou F_leadingResetRequest en direction de l'ECSP, il convient qu'un ECSC inhibe l'inauguration de train ETB.

L'inhibition de l'inauguration de train ETB peut être annulée une fois la transaction de demande de passage en tête terminée.

6.5.3.4 Conflit de tête

Un conflit de tête peut survenir lorsque deux trains sont accouplés alors que les deux ont une rame de tête.

Un conflit de tête doit être indiqué à l'ECSC au plus tard 1,0 s après sa détection.

NOTE 1 Un conflit de tête est indiqué par la primitive de fonction F_leadingStatusRequest.

NOTE 2 La présente partie de l'IEC 61375 ne spécifie pas la façon dont un conflit de tête peut être résolu, car cela dépend fortement du modèle applicatif. Ces détails peuvent être indiqués au conducteur de telle manière qu'il puisse désactiver l'une des rames de tête.

NOTE 3 En cas de conflit de tête, le fonctionnement peut être restreint jusqu'à ce que le conflit de tête ait été résolu.

6.5.3.5 Défaillance de l'ECSC

En cas de défaillance de l'ECSC (voir 6.3), l'ECSP doit définir le paramètre de télégramme 'reqLead' sur la valeur FALSE.

6.6 Fonction confirmation/correction

6.6.1 Généralités

Cette fonction permet de confirmer la composition du train lorsqu'elle a été découverte durant l'inauguration du train (se reporter à l'IEC 61375-2-5) ou de corriger la composition de train découverte en ajoutant/insérant des 'sauts'. Un saut est un véhicule ou une séquence de véhicules qui sont physiquement présents, mais qui n'ont pas été découverts durant l'inauguration du train (si les véhicules ne sont pas alimentés par exemple). Une confirmation ou correction est valable sur l'ensemble du train, ce qui signifie que cette information sera tout d'abord coordonnée avec tous les autres ECSP avant d'être validée. Cela implique

également que tous les autres ECSP ajouteront ces informations de confirmation et de correction à leur TTDB locale. Une composition de train confirmée ou corrigée recevra une signature unique (opTrnTopoCnt) pouvant être utilisée dans les fonctions liées à la sécurité pour s'assurer que toutes les communications sur l'ensemble du train se basent sur la même vision de composition du train.

NOTE Une correction de la composition de train découverte affectera seulement le répertoire opérationnel de train de la TTDB. Le répertoire de train TTDB et le répertoire de réseau de train tels que définis dans l'IEC 61375-2-5 ne seront pas affectés.

6.6.2 Primitives de fonction

Les primitives de fonction sont fournies par l'ECSP et peuvent être invoquées par l'ECSC local pour contrôler les confirmations et les corrections.

Les fonctions définies (du Tableau 38 au Tableau 40) et présentées à la Figure 31 doivent être prises en charge.

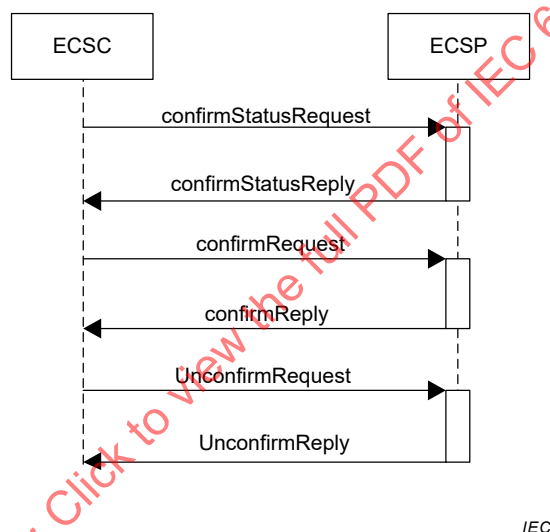


Figure 31 – Diagramme de séquences de la fonction confirmation

Tableau 38 – Primitives de la fonction confirmation: F_confirmStatusRequest

Primitive de fonction:	F_confirmStatusRequest
Description:	Lire l'état du répertoire de train
Paramètre de demande:	-
Paramètre de réponse:	Valeurs de paramètres TTDB: • trnDirState tel que défini en 5.3.3.2.12 • opTrnDirState tel que défini en 5.3.3.2.12 • opTrnTopoCnt tel que défini en 5.3.3.2.12

Tableau 39 – Primitives de la fonction confirmation: F_confirmRequest

Primitive de fonction:	F_confirmRequest
Description:	Corriger et confirmer une composition de train. L'information de correction doit être entrée dans le télégramme ETBCTRL (paramètres ConfVehCnt et ConfVehList). Si seule une confirmation est demandée, aucune liste de véhicules confirmés ne doit être remplie (paramètre ETBCTRL ConfVehCnt = 0)
Paramètre de demande:	ensemble de règles de confirmation et de correction (voir 6.7.2.2)
Paramètre de réponse:	-

Tableau 40 – Primitives de la fonction confirmation: F_unconfirmRequest

Primitive de fonction:	F_unconfirmRequest
Description:	Infirmier une composition de train. Cela comprend aussi le retrait de toute information de correction du télégramme ETBCTRL.
Paramètre de demande:	-
Paramètre de réponse:	-

6.6.3 Protocole d'ECSP à ECSP

6.6.3.1 Généralités

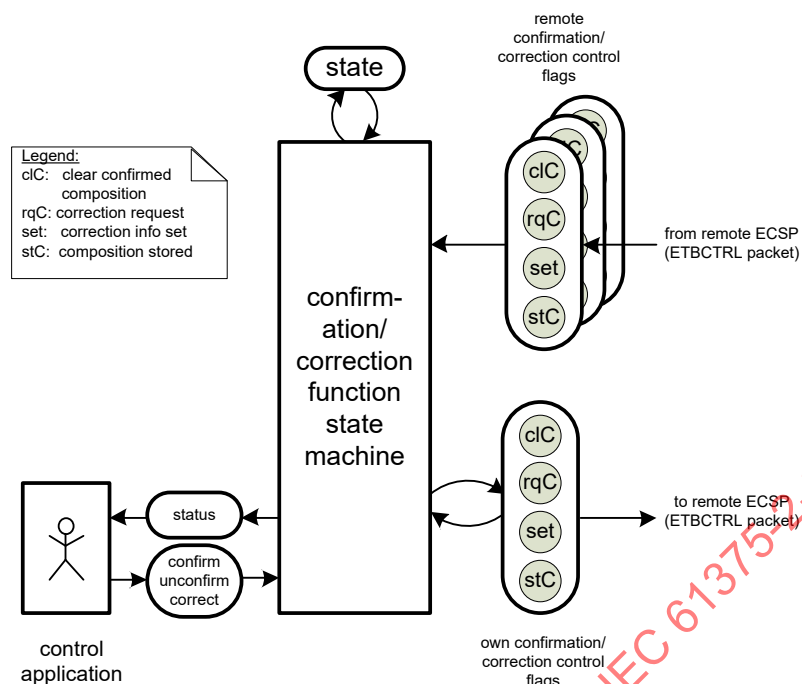
Le présent paragraphe définit le protocole du service lié à l'échange de données entre les ECSP homologues.

La fonction confirmation/correction utilise quatre paramètres ('balises de contrôle') du télégramme ETBCTRL pour le contrôle (voir Tableau 41).

Tableau 41 – Balises de contrôle de la fonction confirmation/correction

Balise de contrôle/Paramètre	Forme abrégée	description	Valeur différée
corrRequest	rqC	Confirmation/correction demandée	FALSE
clearConfComp	clC	Effacer la composition confirmée	Inutile
corrInfoSet	set	Informations de correction introduites	FALSE
compStored	stC	Composition sauvegardée	Inutile
opTrnTopoCnt	-	Valeur calculée du compteur de topographie opérationnel du train	inutile
inutile = cette valeur ne doit pas être utilisée pour le calcul.			

L'état de la fonction confirmation/correction est déterminé par l'ensemble de balises reçues des autres rames (balises de contrôle de confirmation/correction à distance, envoyées par les rames répertoriées dans le répertoire de train avec une valeur cstTopoCnt != 0) et l'ensemble considéré de balises (balises de contrôle de confirmation/correction considérées) comme le montre la Figure 32.



IEC

Anglais	Français
state	état
Legend: clC: clear confirmed composition rqC: correction request set: correction info set stC: composition stored	Légende: clC: clear confirmed composition (suppression d'une composition confirmée) rqC: correction request (demande de correction) set: correction info set (informations de correction prises en compte) stC: composition stored (composition sauvegardée)
remote confirmation/correction control flags	balises de contrôle de confirmation/correction à distance
confirmation/correction function state machine	diagramme d'états de la fonction confirmation/correction
status	statut
confirm unconfirm correct	confirmer, infirmer, corriger
control application	application de contrôle
from remote ECSP (ETBCTRL packet)	depuis l'ECSP distant (paquet ETBCTRL)
to remote ECSP (ETBCTRL packet)	vers l'ECSP distant (paquet ETBCTRL)
own confirmation/correction control flags	balises de contrôle de confirmation/correction considérées

Figure 32 – Diagramme de blocs du diagramme d'états de la fonction confirmation/correction

Toutes ces balises sont échangées cycliquement à l'aide du télégramme ETBCTRL, mais il doit être considéré que, en raison de l'échange asynchrone des télégrammes ETBCTRL, l'ensemble complet de balises de contrôle pourrait ne pas être cohérent durant un certain temps, ce qui conduit à des effets transitoires.

Localement, le diagramme d'états de la fonction confirmation/correction est déclenché par l'application de contrôle ECSP qui envoie une demande de confirmation, de confirmation d'abandon ('infirmation') ou de correction au diagramme d'états. Le diagramme d'états, en retour, envoie des informations sur l'état réel à l'application de contrôle.

6.6.3.2 Séquence de correction

L'interaction pour la correction entre l'ECSP des différentes rames du train est décrite à la Figure 33 sous forme d'exemple d'un train comportant deux rames A et B.

Initialement, les quatre balises de contrôle (stC,set,rqC,clC) sont configurées au (0000) logique³.

La confirmation avec ou sans correction est initiée par un ECSC (l'ECSC A dans l'exemple) en invoquant la primitive de fonction F_confirmRequest. L'ECSP A ajoutera ensuite les informations de confirmation/correction au télégramme ETBCTRL qui informe tous les ECSP (y compris l'ECSP A) de la correction ou de la confirmation exigée (balises de contrôle définies sur (0010)).

Les informations de correction comprennent:

- le nombre de véhicules confirmés dans le train (paramètre 'confVehCnt' du télégramme ETBCTRL);
- une liste ordonnée des véhicules confirmés dans le train (paramètre 'ConfVehList' du télégramme ETBCTRL).

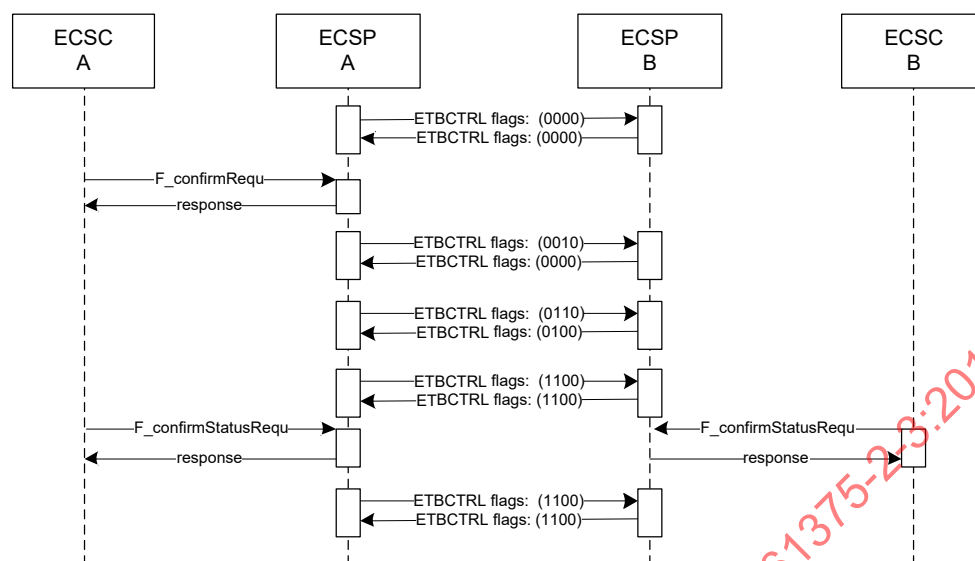
L'ECSP A calculera ensuite un répertoire opérationnel de train confirmé/corrigé, puis définira les balises de contrôle sur (0110).

L'ECSP B calculera également un répertoire opérationnel de train confirmé/corrigé, puis définira les balises de contrôle sur (0100).

Si tous les ECSP ont calculé un répertoire opérationnel de train confirmé/corrigé, l'ECSP passe à l'état 'CompStored' qui est reflété dans les balises de contrôle définies sur (1100).

L'ECSC peut demander un état de confirmation en invoquant la primitive de fonction F_confirmStatusRequest.

³ '0' signifie FALSE, '1' signifie TRUE. Ici: (0000) signifie (FALSE,FALSE,FALSE,FALSE).



IEC

Anglais	Français
response	réponse
ETBCTRL flags	balises ETBCTRL

Figure 33 – Diagramme de séquences du protocole de correction/confirmation (exemple)

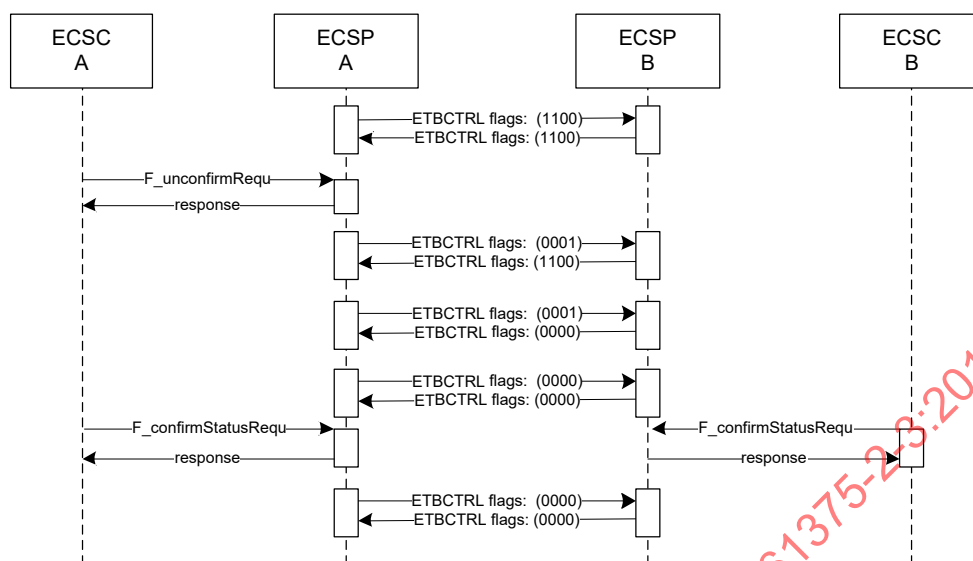
6.6.3.3 Séquence de confirmation d'une composition existante

La séquence de confirmation d'une composition existante est identique à la séquence de correction. La seule différence réside dans le fait que la liste des véhicules confirmés est vide.

6.6.3.4 Séquence d'annulation de composition confirmée

Une composition confirmée peut être annulée en invoquant la primitive de fonction `F_unconfirmRequ`, à quoi l'ECSP doit répondre par `F_unconfirmResp`. Dès réception d'une demande d'annulation, l'ECSP doit activer la balise de contrôle 'cIC' durant trois cycles de données de processus consécutifs (voir Figure 34).

Un ECSP recevant un télégramme ETBCTRL avec la balise de contrôle 'cIC' activée enverra un accusé de réception en définissant ses balises de contrôle sur (0000).



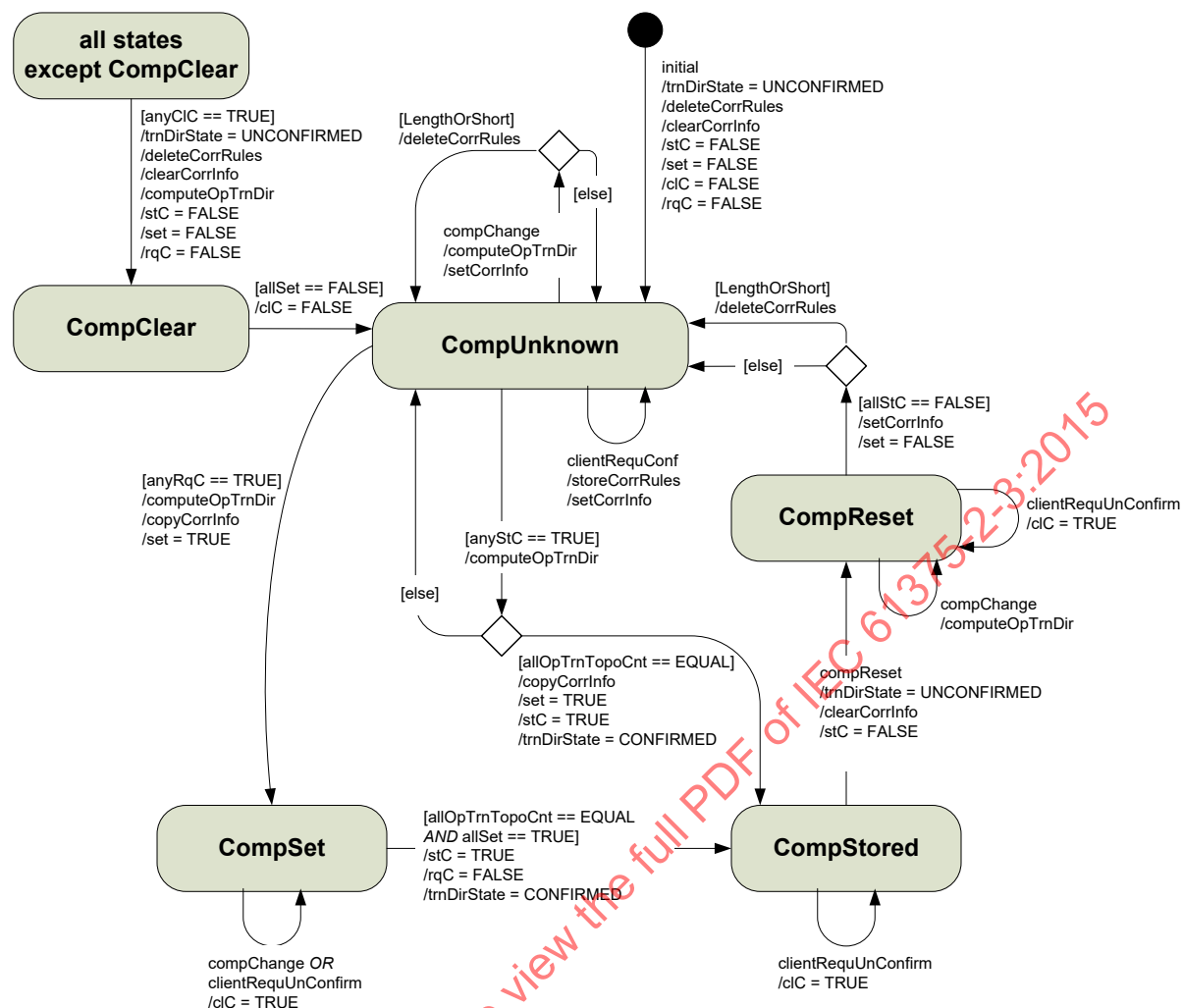
IEC

Anglais	Français
response	réponse
ETBCTRL flags	balises ETBCTRL

Figure 34 – Diagramme de séquences du protocole d'information (exemple)

6.6.4 Diagramme d'états

La fonction confirmation/correction est définie par le diagramme d'états présenté à la Figure 35, à l'aide des explications supplémentaires du Tableau 42 au Tableau 44.



IEC

Anglais	Français
all states	tous les états

Figure 35 – Diagramme d'états de la fonction confirmation/correction

NOTE Pour les transitions déclenchées par la réception d'un télégramme ETBCTRL, seuls les guards sont illustrés par souci de lisibilité.

Tableau 42 – Diagramme d'états de la fonction confirmation/correction: triggers

Trigger	Description
Initial	Alimentation ou relance d'ECSP
clientRequConfirm	Commande de confirmation / correction reçue de l'ECSC
clientRequUnConfirm	Commande reçue de l'ECSC pour infirmer (réinitialiser)
compChange	La composition du train a changé. Cet événement est déclenché par le diagramme d'états ETBCTRL (voir 6.4.5).
compReset	Réinitialisation de la composition confirmée

Tableau 43 – Diagramme d'états de la fonction confirmation/correction: guards

Guard	Description
anyRqC == TRUE	OR logique sur les balises de contrôle 'rqC' dans tous les télégrammes ETBCTRL égal à TRUE
anyCIC == TRUE	OR logique sur les balises de contrôle 'clC' dans tous les télégrammes ETBCTRL égal à TRUE
allStC == FALSE	OR logique sur les balises de contrôle 'stC' dans tous les télégrammes ETBCTRL égal à FALSE
allSet == FALSE	OR logique sur les balises de contrôle 'set' dans tous les télégrammes ETBCTRL égal à FALSE pour une période d'au moins 3 cycles ETBCTRL NOTE Cette condition temporelle doit permettre de s'assurer que l'ECSP également inséré passe bien à l'état CompUnknown.
anyStC	OR logique sur les balises de contrôle 'stC' dans tous les télégrammes ETBCTRL
allOpTrnTopoCnt	Comparer le paramètre 'opTrnTopoCnt' dans tous les télégrammes ETBCTRL EQUAL = valeur identique dans tous les télégrammes
LenghtOrShort	Les véhicules d'extrémité découverts de la nouvelle composition ne sont pas identiques aux véhicules d'extrémité découverts de la dernière composition confirmée (= allongement ou raccourcissement du train).

Tableau 44 – Diagramme d'états de la fonction confirmation/correction: actions

Action	Description
computeOpTrnDir	Calculer le répertoire opérationnel de train en tenant compte des informations de correction obtenues par le biais du télégramme ETBCTRL et mettre à jour le paramètre de télégramme ETBCTRL 'opTrnTopoCnt'.. (voir 6.7.3).
trnDirState	Définir le paramètre TTDB 'trnDirState' sur la valeur définie.
storeCorrRules	Définir les règles de correction obtenues à partir de l'utilisateur en local
deleteCorrRules	Supprimer les règles de correction stockées en local
setCorrInfo	Vérifier la cohérence des règles de correction stockées en local par rapport au répertoire opérationnel de train et ajouter les informations de correction au télégramme ETBCTRL comme défini en 6.7.2.
copyCorrInfo	Cette action est facultative. Copier la liste des véhicules confirmés (paramètres ETBCTRL 'confVehList' et 'confVehCnt') de l'ECSP distributrice dans le télégramme ETBCTRL considéré (seulement si cet ECSP ne distribue pas déjà la liste des véhicules confirmés). Le paramètre ETBCTRL 'rqC' ne doit pas être modifié. NOTE La copie évite que les informations de correction ne soient perdues en cas de défaillance de l'ECSP qui les a initialement distribuées.
clearCorrInfo	Supprimer les informations de correction du télégramme ETBCTRL (supprimer la liste des véhicules confirmés) et définir le paramètre ETBCTRL 'confVehCnt' sur 0.

6.6.5 Défaillance de l'ECSC

En cas de défaillance de l'ECSC (voir 6.3), aucune action spéciale n'est demandée pour l'ECSP.

6.7 Calcul du répertoire opérationnel de train

6.7.1 Généralités

Le calcul du répertoire opérationnel de train est pris en charge par deux actions:

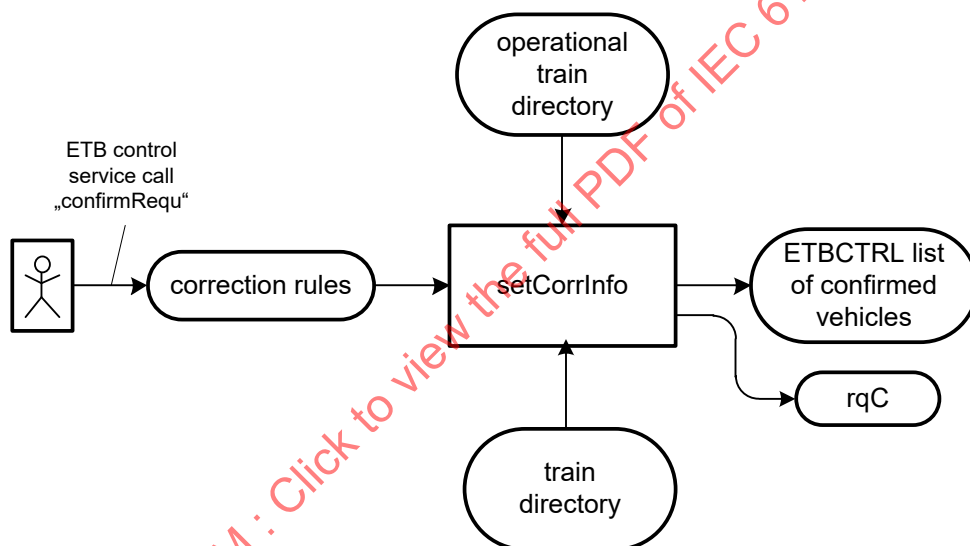
- l'action "setCorrInfo"
- l'action "computeOpTrnDir"

Ces deux actions sont contrôlées par la fonction véhicule de tête (computeOpTrnDir seulement) et la fonction confirmation/correction.

6.7.2 Action setCorrInfo

6.7.2.1 Généralités

Cette action a pour objet de transférer les règles de correction fournies par l'utilisateur dans une liste de véhicules confirmés (confVehList) qui sera jointe au télégramme ETBCTRL. Ce processus est montré à la Figure 36.



IEC

Anglais	Français
operational train directory	répertoire opérationnel de train
ETB control service call	appel de service de contrôle ETB
correction rules	règles de correction
train directory	répertoire de train
ETBCTRL list of confirmed vehicles	liste ETBCTRL des véhicules confirmés

Figure 36 – Diagramme de blocs de l'action "setCorrInfo"

Pour le transfert, les informations du répertoire opérationnel (ex: paramètre TTDB 'vehId') et du répertoire opérationnel de train (ex: paramètre TTDB 'trnVehNo') sont nécessaires.

Ce transfert comporte un contrôle de cohérence, qui permet de vérifier si les règles peuvent être appliquées correctement au répertoire de train donné selon la politique de correction définie en 6.7.2.3.

6.7.2.2 Règles de Confirmation/Correction

Les règles de Confirmation/Correction sont des règles, qui décrivent quelles actions de confirmation ou correction doivent être réalisées sur une configuration de train donnée telle que définie par le répertoire de train.

Les règles définies au Tableau 45 doivent être prises en charge.

Tableau 45 – Règles de Confirmation/Correction

Règle	Description
Confirmer la composition sans correction	La composition de train telle qu'elle est découverte durant l'inauguration du train et représentée par le répertoire de train est confirmée (aucune correction nécessaire).
Insérer N véhicules entre le véhicule portant le VehId X et le véhicule portant le VehId Y	Corriger la composition de train découverte en insérant un saut de N véhicules entre les véhicules découverts X et Y.
Insérer N véhicules avant l'extrémité 1 du véhicule portant le VehId X	Corriger la composition de train découverte en insérant un saut de N véhicules avant le véhicule.
Insérer N véhicules après l'extrémité 2 du véhicule portant le VehId X	Corriger la composition de train découverte en insérant un saut de N véhicules après le véhicule.
Confirmer la composition avec correction	La composition de train incluant les sauts de véhicules doit être confirmée.
Infirmier la composition	Supprimer toutes les corrections apportées et infirmer la composition

NOTE Il ne relève pas du domaine d'application de la présente norme de définir un codage spécifique de ces règles. Une mise en œuvre possible est donnée à l'Annexe E.

6.7.2.3 Politique de correction

La politique de correction définit les conditions dans lesquelles une correction est acceptée et une correction est maintenue après une inauguration de train.

- a) Une correction, autrement dit l'insertion d'un saut de N véhicules dans une composition de train découverte, doit être acceptée si les conditions suivantes sont satisfaites:
 - 1) Le saut n'est pas inséré dans les limites d'une rame (seule une insertion entre les rames est permise).
 - 2) Le nombre total de 63 véhicules n'est pas dépassé.

Dans tous les autres cas, la demande de correction utilisateur doit être refusée.

- b) Si une nouvelle inauguration de train survient après une correction, la correction doit seulement persister et la confirmation doit être maintenue si les conditions suivantes sont satisfaites:
 - 1) Si des véhicules découverts (non présents après la dernière inauguration de train) apparaissent à la position d'un saut, le nombre N de sauts de véhicules doit être égal au nombre de véhicules récemment découverts.
 - 2) Les véhicules d'extrémité découverts de la nouvelle composition sont identiques aux véhicules d'extrémité découverts de l'ancienne composition (aucun allongement ni raccourcissement du train).

Dans tous les autres cas, la composition doit être infirmée.

- c) Si, après une nouvelle inauguration de train, des véhicules intermédiaires préexistants ne sont plus découverts, mais que les véhicules d'extrémité découverts sont identiques aux véhicules d'extrémité découverts précédemment (aucun allongement ni raccourcissement du train), un saut doit être automatiquement inséré en remplacement des véhicules

intermédiaires manquants. Cela ne doit néanmoins être réalisé que si le saut n'est pas inséré dans les limites d'une rame (seule une insertion entre les rames est permise).

6.7.2.4 Exécution d'une fonction

La fonction setCorrInfo doit se comporter comme suit:

S'il y a seulement une règle de confirmation de la composition sans correction et s'il n'y a eu aucune insertion de saut automatique (voir 6.7.2.3, point 3), aucune liste de véhicules confirmés (confVehList) ne doit être jointe au télégramme ETBCTRL (paramètre ETBCTRL 'confVehCnt' = 0) et le paramètre ETBCTRL 'rqC' doit être défini sur TRUE.

S'il y a seulement une règle de confirmation de la composition sans correction et s'il y a eu une insertion de saut automatique, la liste de véhicules confirmés (confVehList) doit être jointe au télégramme ETBCTRL (paramètre ETBCTRL 'confVehCnt' = 0) et le paramètre ETBCTRL 'rqC' doit être défini sur TRUE.

S'il n'y a pas de règles de correction ni de règle de confirmation de la composition sauvegardée, aucune liste de véhicules confirmés (confVehList) ne doit être jointe au télégramme ETBCTRL (paramètre ETBCTRL 'confVehCnt' = 0) et le paramètre ETBCTRL 'rqC' doit être défini sur FALSE.

Si les règles de correction ne sont pas cohérentes avec la politique de correction, aucune liste de véhicules confirmés (confVehList) ne doit être jointe au télégramme ETBCTRL (paramètre ETBCTRL 'confVehCnt' = 0) et le paramètre ETBCTRL 'rqC' doit être défini sur FALSE.

S'il y a des règles de correction ET que les règles de correction sont cohérentes avec la politique de correction, la liste de véhicules confirmés (confVehList) doit être jointe au télégramme ETBCTRL (paramètre ETBCTRL 'confVehCnt' != 0) et le paramètre ETBCTRL 'rqC' doit être défini sur TRUE.

NOTE 1 Le contrôle de cohérence présume que, après la bonne exécution d'une confirmation/correction, la composition de train confirmée est égale à la composition réelle du train et qu'un changement de la composition réelle du train ne peut être réalisé qu'en découplant ou en accouplant des rames (ex: accouplement de deux trains). Cela signifie que, si une composition de train a été mal confirmée ou mal corrigée (ex: un véhicule non utilisable dans le train a été oublié durant la confirmation/correction), le répertoire opérationnel de train peut lui aussi présenter des résultats erronés.

NOTE 2 Pendant une correction, les sauts annoncés dans la composition du train indiquent les emplacements des véhicules non accessibles. À ce moment, il arrive souvent que seul le nombre de véhicules présents dans le saut soit connu (parce qu'ils sont visibles par le conducteur, par exemple), mais pas la manière dont ces véhicules sont associés aux rames. Par exemple, un saut de 3 véhicules pourrait être comblé par une rame de 3 véhicules, une rame de 1 véhicule et une rame de 2 véhicules, ou trois rames de 1 véhicule.

NOTE 3 Il existe une action associée 'copyCorrInfo' qui copie les informations de correction obtenues à partir d'un autre télégramme ETBCTRL dans le télégramme ETBCTRL considéré (voir 6.6.4). Dans ce cas, le paramètre ETBCTRL 'rqC' est défini sur FALSE, mais le paramètre ETBCTRL 'confVehCnt' est défini sur une valeur != 0. La combinaison de ces deux paramètres indique la nature de la confirmation/correction telle qu'elle peut être perçue à partir de la table de vérité suivante:

rqC	confVehCnt	description
FALSE	= 0	Composition infirmée
FALSE	!= 0	Informations de correction copiées
TRUE	= 0	Confirmation demandée
TRUE	!= 0	Correction demandée

EXEMPLES

Quelques exemples-types de résultats de contrôle de cohérence sont illustrés à la Figure 37.

Ces exemples se basent sur une composition de train corrigée telle qu'illustrée au début de la Figure 37. Chaque véhicule est également une rame. La liste des véhicules confirmés (confVehList) qui est jointe au télégramme ETBCTRL ressemblera à ce qui suit:

confVehCnt = 9

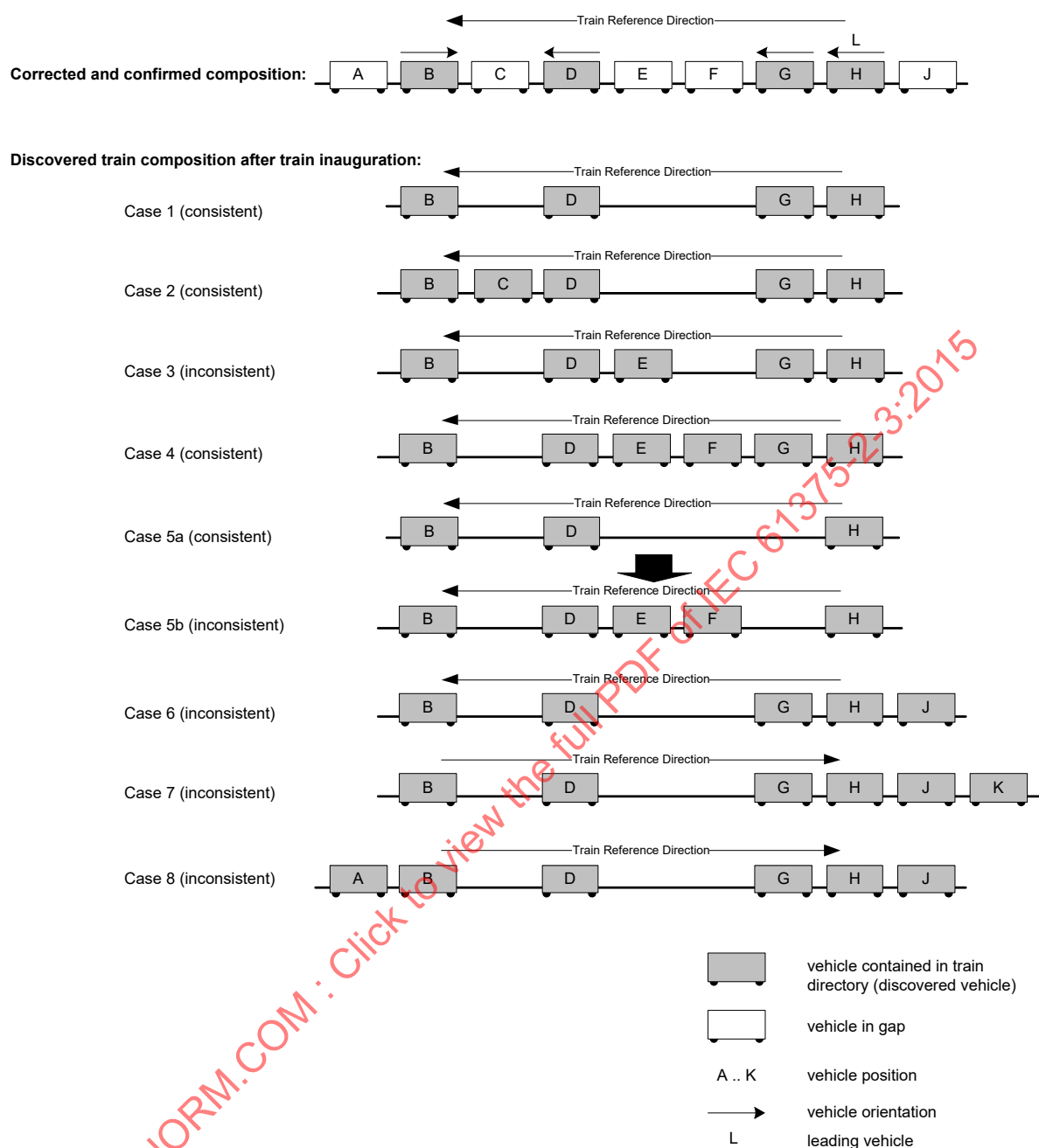
trnVehNo	isLead	leadDir	vehOrient
0	1	0	0
1	1	0	2
0	1	0	0
2	1	0	1
0	1	0	0
0	1	0	0
3	1	0	1
4	2	1	1
0	1	0	0

Le cas 1 correspond exactement à la composition confirmée, dans les cas 2 et 4 le nombre de nouveaux véhicules correspond au nombre de sauts de véhicules à cet endroit, ces cas sont donc cohérents. Dans le cas 3, l'endroit auquel doit être mappé le nouveau véhicule n'est pas évident, cela peut être E ou F, ce qui est donc considéré comme incohérent.

Le cas 5a décrit un cas de disparition, après inauguration, d'un véhicule qui était présent auparavant (créant un saut de trois véhicules à cet endroit: deux sauts de véhicules corrigés plus un véhicule manquant) et deux nouveaux véhicules avec des 'vehld' inconnus sont signalés à ces endroits après l'inauguration suivante (cas 5b). Le saut n'étant pas entièrement comblé, ce cas est considéré incohérent.

Les cas 6 et 8 présentent un saut venant compléter l'extrémité du train, ce qui est incohérent. Il ne peut pas être déterminé si le véhicule représentant le saut est devenu actif ou si un nouveau véhicule a été ajouté, ce qui porte dans ce cas le nombre de véhicules à un total de 10.

Le cas 7 montre un allongement réel du train, mais là encore on ne peut pas déterminer si l'un des nouveaux véhicules comble le saut de véhicule et qu'un seul véhicule a été ajouté ou si le saut de véhicule est encore présent et que deux véhicules ont été ajoutés.



IEC

Anglais	Français
Train Reference Direction	Direction de référence du train
Corrected and confirmed composition:	Composition corrigée et confirmée:
Discovered train composition after train inauguration:	Composition du train découverte après l'inauguration:
Case x (consistent)	Cas x (cohérent)
Case x (inconsistent)	Cas x (incohérent)
vehicle contained in train directory	Véhicule inclus dans le répertoire de train
vehicle in gap	saut de véhicule
vehicle position	position du véhicule
vehicle orientation	orientation du véhicule
leading vehicle	véhicule de tête

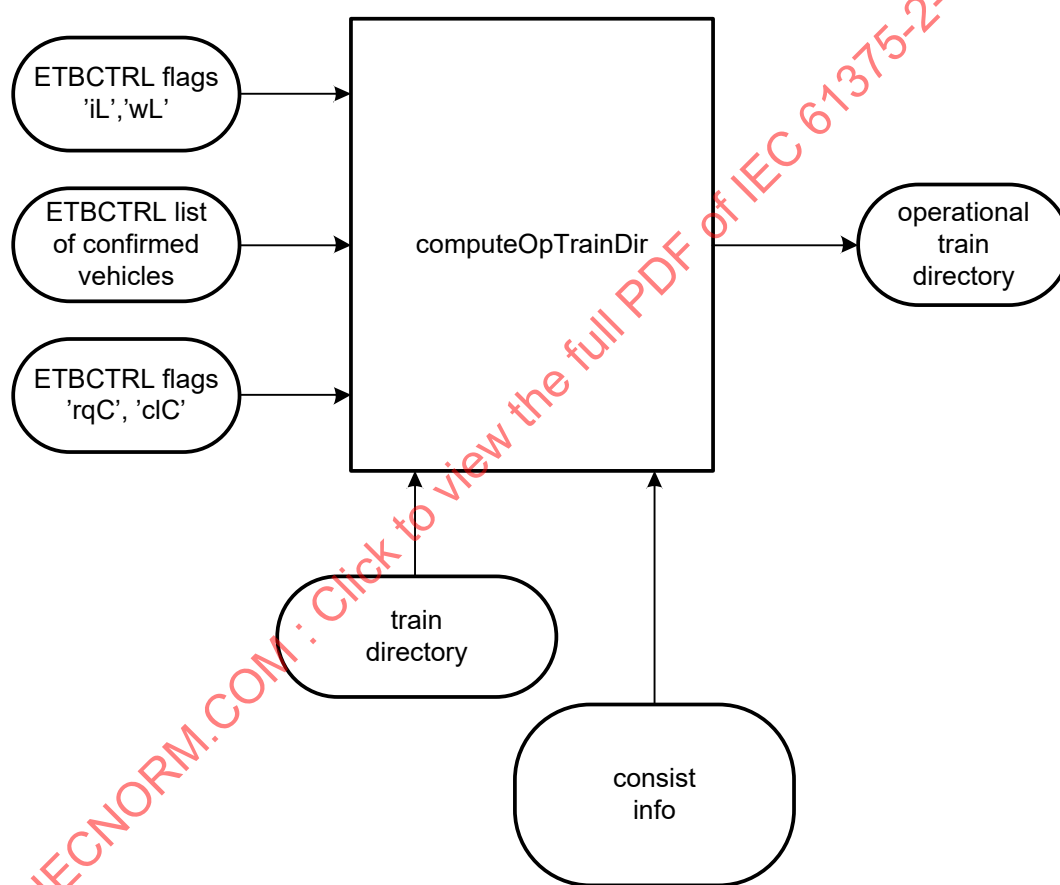
Figure 37 – Exemples de contrôle de cohérence de la composition du train

6.7.3 Action computeOpTrnDir

6.7.3.1 Généralités

L'action computeOpTrnDir doit générer le répertoire opérationnel de train en lisant et en interprétant les informations provenant des quatre sources de données d'entrée (voir Figure 38):

- les informations de correction, obtenues à partir du télégramme ETBCTRL et fournies par la fonction confirmation/correction, notamment l'action 'setCorrInfo';
- les informations concernant un véhicule de tête au sein du train, obtenues à partir du télégramme ETBCTRL et fournies par la fonction véhicule de tête (voir 6.5);
- la séquence réelle de rames dans le train, obtenue à partir du répertoire du train;
- le nombre et la séquence de véhicules dans une rame, obtenue à partir de la liste d'informations de rame.



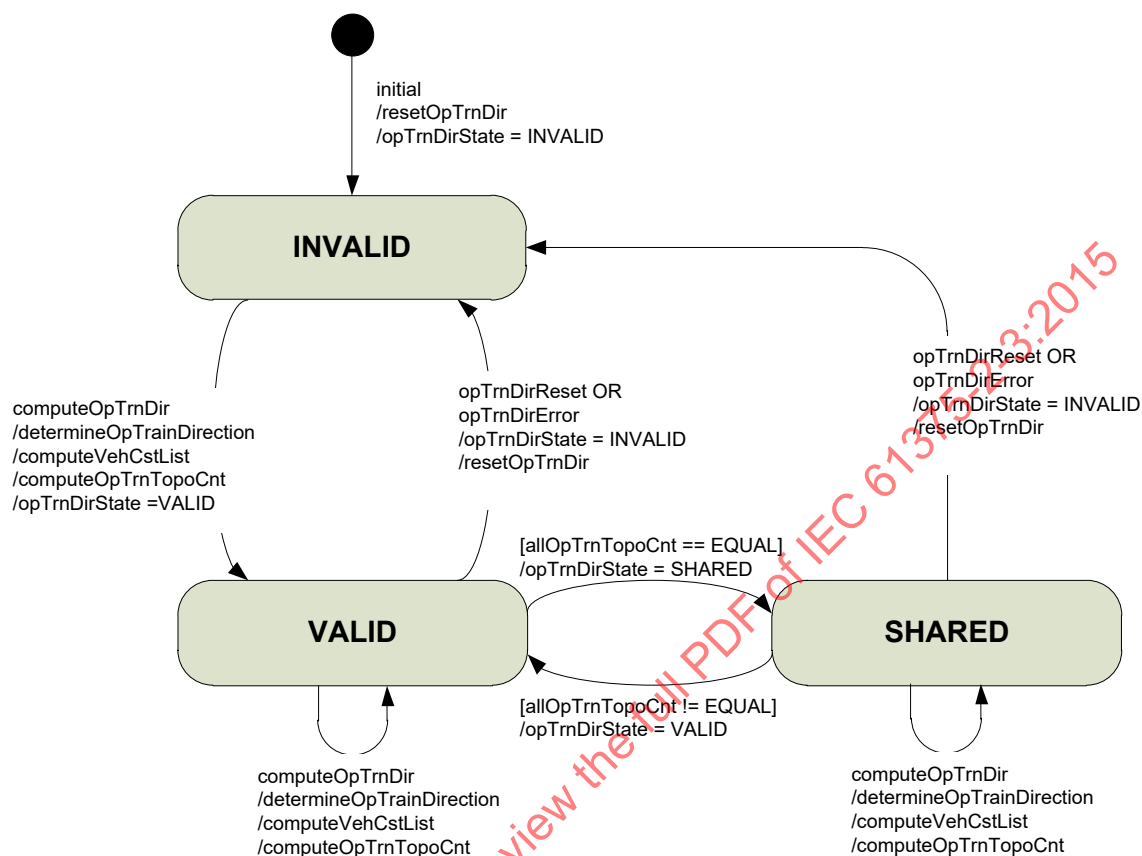
IEC

Anglais	Français
operational train directory	répertoire opérationnel de train
train directory	répertoire de train
consist info	Informations de rame
ETBCTRL flags	balises ETBCTRL
ETBCTRL list of confirmed vehicles	liste ETBCTRL des véhicules confirmés

Figure 38 – Calcul du répertoire opérationnel de train

6.7.3.2 Diagramme d'états de calcul du répertoire opérationnel de train

Le diagramme d'états présenté à la Figure 39 et expliqué en détail du Tableau 46 jusqu'au Tableau 48 définit le calcul du répertoire opérationnel de train.



IEC

Anglais	Français
INVALID	INVALIDE
VALID	VALIDE
SHARED	PARTAGE

Figure 39 – Diagramme d'états computeOpTrnDir

Tableau 46 – Diagramme d'états de calcul du répertoire opérationnel de train: trigger

Trigger	Description
initial	Initialise ou relance l'ECSP
computeOpTrnDir	Un nouveau calcul du répertoire opérationnel de train a été demandé.
opTrnDirReset	Une réinitialisation du répertoire opérationnel de train a été demandée.
opTrnDirError	Une erreur s'est produite, empêchant le calcul d'un répertoire opérationnel de train valide.

Tableau 47 – Diagramme d'états de calcul du répertoire opérationnel de train: guards

Guard	Description
allOpTrnTopoCnt == EQUAL	Des télégrammes ETBCTRL ont été reçus en provenance de tous les ECSP actifs répertoriés dans le répertoire de train local avec un paramètre cstTopoCnt != 0 et tous les télégrammes ETBCTRL valides reçus (y compris le télégramme ETBCTRL considéré) ont une valeur de paramètre 'trnTopoCnt' identique.

Tableau 48 – Diagramme d'états de calcul du répertoire opérationnel de train: action

Action	Description
determineOpTrainDirection	La direction opérationnelle du train doit être déterminée comme spécifié en 4.2.4.3.
computeVehCstList	Une liste ordonnée des rames (paramètre TTDB 'opCstList') et des véhicules (paramètre TTDB 'opVehList') doit être générée en utilisant les informations du répertoire de train et de la liste d'informations de rame. Si le paramètre 'confVehCnt' a une valeur != 0, des sauts de véhicules et des véhicules manquants, selon les informations de correction du télégramme ETBCTRL, doivent être considérés comme défini en 6.7.2. En règle générale, un saut de véhicule doit être compté comme une rame comprenant un seul véhicule.
computeOpTrnTopoCnt	La valeur du compteur de topographie opérationnel du train doit être calculée comme défini en 5.3.3.2.13.
resetOpTrnDir	Le répertoire opérationnel de train et le statut du répertoire opérationnel de train doivent être invalidés en définissant les valeurs de paramètres TTDB suivantes: opTrnTopoCnt = 0 opTrnOrient = inconnu
opTrnDirState	Définit le paramètre TTDB 'opTrnDirState' sur la valeur définie.

NOTE A l'état 'SHARED', tous les ECSP sont synchronisés en utilisant la même valeur OpTrnTopoCnt. Les appareils terminaux peuvent en tirer profit, car ils pourront envoyer des télégrammes TRDP à l'ensemble du train sans risquer la suppression du télégramme par le destinataire en raison d'une incohérence de la valeur opTrnTopoCnt.

EXEMPLE

Avec la composition de train corrigée obtenue à partir de la Figure 37, la liste de véhicules du répertoire opérationnel de train ressemblera au contenu du Tableau 49.

Tableau 49 – Exemple de répertoire opérationnel de train

vehId	opVehNo	isLead	leadDir	trnVehNo	VehOrient	ownOpCstNo
-	1	1	0	0	0	1
'H'	2	2	1	4	2	2
'G'	3	1	0	3	2	3
-	4	1	0	0	0	4
-	5	1	0	0	0	5
'D'	6	1	0	2	2	6
-	7	1	0	0	0	7
'B'	8	1	0	1	1	8
-	9	1	0	0	0	9

6.8 Fonction mode sommeil (option)

6.8.1 Généralités

Cette fonction permet de mettre un train entier en mode faible puissance (= mode sommeil) et de revenir sur demande au fonctionnement normal. Le mode sommeil sera activé si toutes les rames en font la demande et sera désactivé si au moins une rame en fait la demande.

6.8.2 Cas d'utilisation du mode sommeil (informel)

Le cas d'utilisation suivant pour la fonction mode sommeil se base sur les concepts définis dans la fiche UIC 556.

Il convient que les ETBN d'un véhicule individuel puissent satisfaire aux conditions répertoriées au Tableau 50.

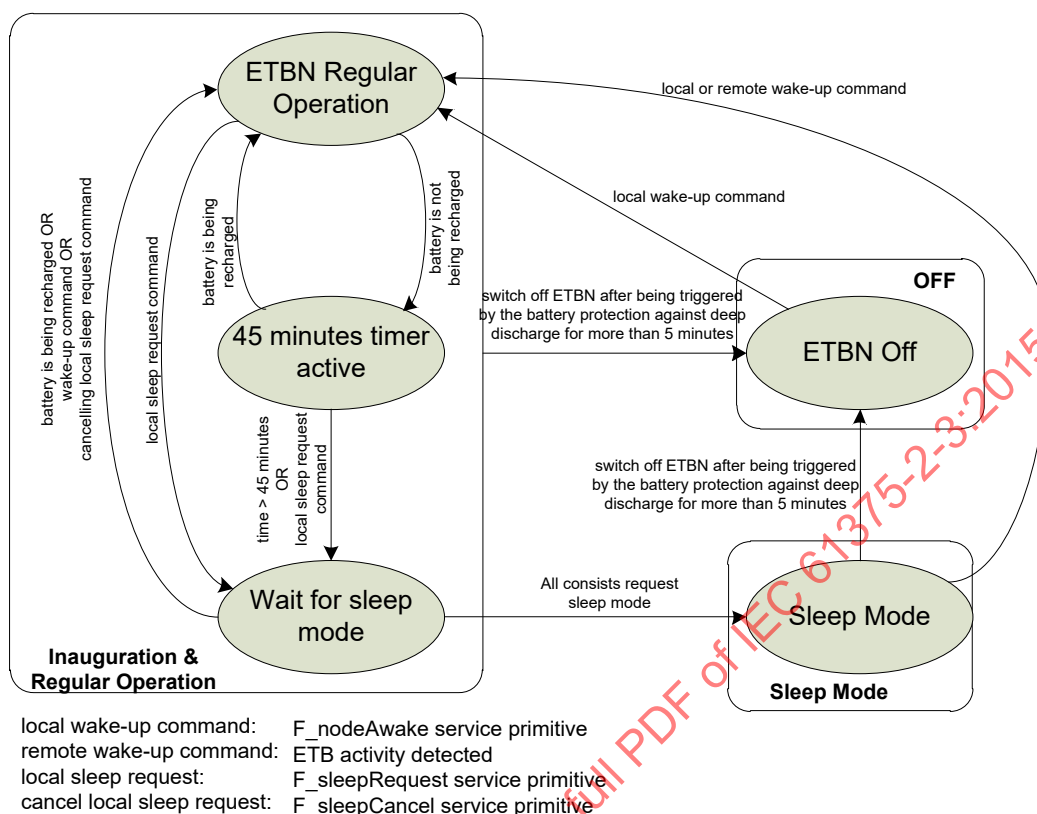
Tableau 50 – Conditions de fonctionnement des ETBN

Non	Condition de fonctionnement	Description
0	OFF	Un ETBN ne peut pas fonctionner lorsque la tension continue du véhicule concerné tombe au-dessous de 70 % de la tension nominale.
1	MODE SOMMEIL	L'ETBN est en mode faible puissance, mais peut recevoir des signaux sur le câble du réseau central de train (ex: trames Ethernet).
2	INAUGURATION	L'ETBN détermine le répertoire du train.
3	FONCTIONNEMENT NORMAL	Echange d'informations sur ETB

Les exigences suivantes déterminent les transitions des conditions de fonctionnement et des conditions internes de l'ETBN:

- La charge de batterie étant disponible, l'ETBN passe en inauguration et en fonctionnement normal.
- Si l'ETBN est en mode sommeil (condition de fonctionnement 1), une action dédiée déclenche le passage en inauguration et en fonctionnement normal (conditions de fonctionnement 2 et 3). Par exemple, dans la fiche UIC 550-1, une telle action est définie par l'appui sur le bouton-poussoir "éclairage voiture" du pupitre de commande.
- En cas de détection d'activité ETB en mode sommeil, l'ETBN passe en inauguration et en fonctionnement normal (conditions de fonctionnement 2 et 3).
- Au bout de 45 min après l'arrêt du chargement des batteries sur l'ensemble du train (ex: coupure de la ligne de train chargeant les batteries à partir de l'unité de traction), l'ETBN passe en mode sommeil (condition de fonctionnement 1) correspondant au concept de mode sommeil spécifié pour l'ETBN dans le présent paragraphe.
- Si la protection de la batterie contre les décharges profondes est activée (voir également fiche UIC 550), l'ETBN est coupé (condition de fonctionnement 0) puisqu'il est alors vraisemblable que toutes les charges télécommandées soient coupées.
- Pour le fonctionnement en mode secours, l'ETBN peut également être mis en marche au-dessous de l'état de coupure le plus faible de la tension minimale, par exemple avec le bouton-poussoir "éclairage voiture" sur le pupitre de commande, à condition que la tension minimale continue (70 % de la tension nominale, IEC 60571) soit disponible.
- Le bit de contrôle "sleepRequest" (demande de passage en sommeil) du télégramme ETBCTRL contrôle la transition entre le fonctionnement des nœuds et l'attente en mode sommeil.

Les états opérationnels correspondants de l'ETBN sont donnés à la Figure 40.



IEC

Anglais	Français
ETBN Regular Operation	Fonctionnement normal de l'ETBN
Battery is being recharged OR wake-up command OR cancelling local sleep request command	Batterie en charge dans la rame OU commande de réveil OU annulation de la commande locale de demande de passage en mode sommeil
Local sleep request command	Commande locale de demande de passage en mode sommeil
Battery is being recharged	Batterie en charge
Battery is not recharged	Batterie non en charge
Time > 45 minutes	Temps > 45 min
Wait for sleep mode	En attente pour passer au mode sommeil
Inauguration & Regular Operation	Inauguration et fonctionnement normal
Local or remote wake-up command	Commande de réveil locale ou à distance
Local wake-up command	Commande de réveil locale
Switch off ETBN after being triggered by the battery protection against deep discharge for more than 5 minutes	Désactivation de l'ETBN après déclenchement par la protection de la batterie contre les décharges profondes pendant plus de 5 min
All consists request sleep mode	Demande de passage en mode sommeil pour toutes les rames
ETBN Off	ETBN désactivé
Sleep mode	Mode sommeil
local wake-up command	commande locale de réveil
remote wake-up command	commande à distance de réveil
local sleep request	demande locale de passage en mode sommeil
cancel local sleep request	annulation de la demande locale de passage en mode sommeil
service primitive	primitive de service
ETB activity detected	activité ETB détectée

Figure 40 – Diagramme d'états du cas d'utilisation "mode sommeil"

L'utilisation de l'état 'ETBN Off' n'est pas spécifiée en détail dans la présente partie de l'IEC 61375, car cela relève d'une initiative locale à la rame qui n'affecte pas l'interopérabilité.

Bien que le cas d'utilisation du mode sommeil décrit ci-dessus soit relatif à un ETBN, la présente partie de la norme définit un concept plus générique qui partage la fonctionnalité de mode sommeil entre l'ECSP et l'ETBN:

- L'ECSP coordonne les demandes de passage en sommeil entre les différentes rames, gère la temporisation et fournit l'interface de contrôle à l'ECSC.
- L'ETBN prend en charge l'arrêt de la transmission, l'activation d'une fonction de détection d'activité sur la ligne en mode sommeil qui détecte les activités sur la ligne et le réveil de la rame en cas de détection d'une activité sur la ligne.

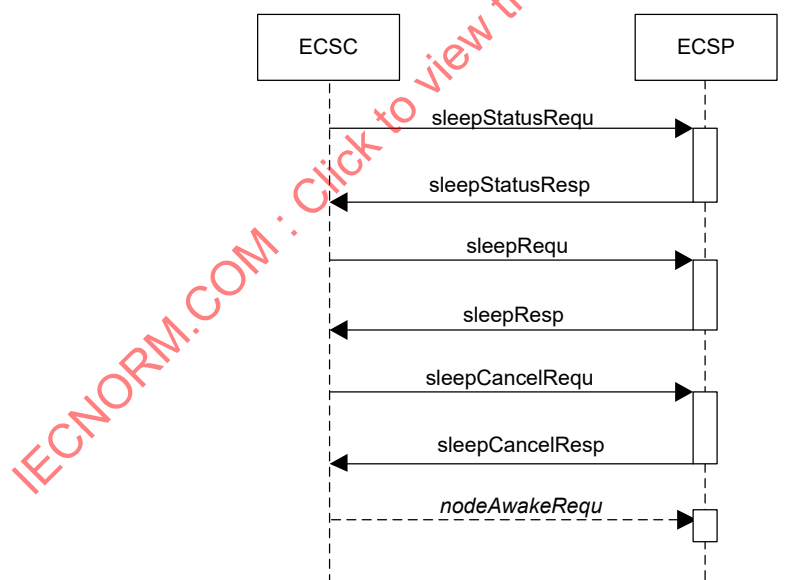
6.8.3 Exclusivité

Toutes les rames d'un train doivent prendre en charge la fonction mode sommeil pour permettre le fonctionnement correct de cette fonction. Le mélange de rames réalisant cette fonction et de rames ne la réalisant pas n'est pas supporté.

6.8.4 Primitives de fonction

Les primitives de fonction sont fournies par l'ECSP et peuvent être invoquées par l'ECSC local pour contrôler le mode sommeil.

Les primitives de fonction définies dans le présent paragraphe (du Tableau 51 au Tableau 54) et présentées à la Figure 41 doivent être prises en charge.



IEC

Figure 41 – Diagramme de séquences de la fonction mode sommeil

Tableau 51 – Primitives de la fonction mode sommeil: F_sleepStatus

Primitive de fonction:	F_sleepStatus
Description:	Lire l'état des demandes de passage en sommeil
Paramètre de demande:	-
Paramètre de réponse:	nombre de rames demandant le passage en sommeil état de la fonction mode sommeil considéré

Tableau 52 – Primitives de la fonction mode sommeil: F_sleepRequest

Primitive de fonction:	F_sleepRequest
Description:	Demande (locale) de passage en sommeil (définit le bit sleepRequest dans ETBCTRL-VDP)
Paramètre de demande:	-
Paramètre de réponse:	sleepRequest = SET/UNSET (ACTIVE / NON ACTIVE)

Tableau 53 – Primitives de la fonction mode sommeil: F_sleepCancel

Primitive de fonction:	F_sleepCancel
Description:	Annule une demande (locale) de passage en sommeil (réinitialise le bit sleepRequest dans ETBCTRL-VDP)
Paramètre de demande:	-
Paramètre de réponse:	sleepRequest = SET/UNSET (ACTIVE / NON ACTIVE)

Tableau 54 – Primitives de la fonction mode sommeil: F_nodeAwake

Primitive de fonction:	F_nodeAwake
Description:	Demande locale de retour en fonctionnement normal (train en éveil). Cette fonction peut être déclenchée par une action locale, par exemple en actionnant un bouton "éclairage voiture" sur le pupitre de commande ou à distance par l'intermédiaire d'un système de communication radio sol-trains. Selon la mise en œuvre du mode sommeil, cette commande peut être dirigée vers l'ECSP ou vers l'un des ETBN ou tous les ETBN.
Paramètre de demande:	-
Paramètre de réponse:	-

6.8.5 Protocole d'ECSP à ECSP

6.8.5.1 Généralités

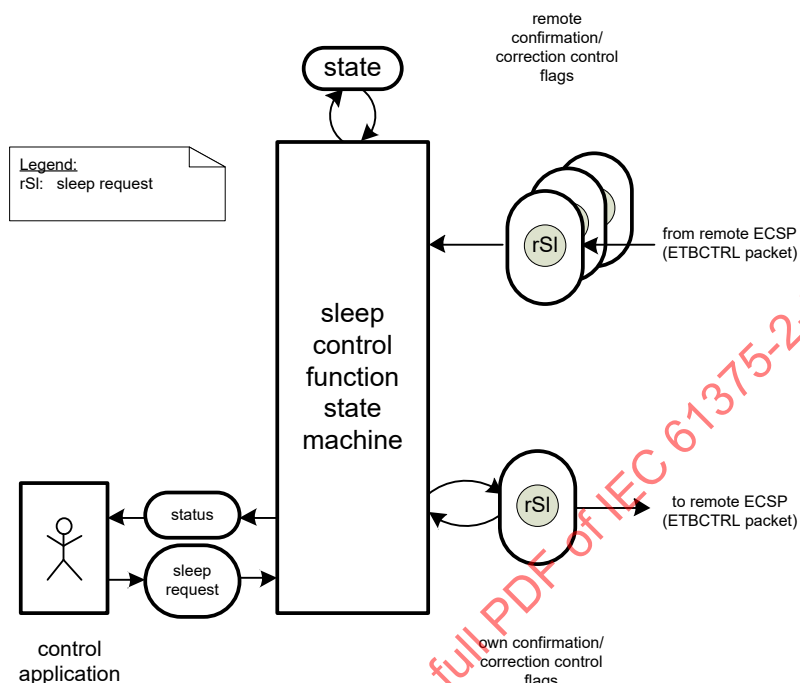
Le présent paragraphe définit le protocole du service lié à l'échange de données entre les ECSP homologues.

La fonction mode sommeil utilise un paramètre ('balise') du télégramme ETBCTRL pour le contrôle (voir Tableau 55).

Tableau 55 – Balises de contrôle de la fonction mode sommeil

control flag	Forme abrégée	description	valeur par défaut
'sleepRequest'	'rSI'	Demande de passage en sommeil	FALSE

L'état de la fonction mode sommeil est déterminé par les balises reçues des autres rames (balises de contrôle de la fonction mode sommeil à distance, envoyées par les rames répertoriées dans le répertoire de train avec une valeur $cstTopoCnt \neq 0$) et la balise considérée (balise de contrôle de la fonction mode sommeil considérée) comme le montre la Figure 42.



IEC

Anglais	Français
state	état
Legend: rSI: sleep request	Légende: rSI: sleep request (demande de passage en mode sommeil)
sleep control function state machine	diagramme d'états de la fonction mode sommeil
remote confirmation/correction control flags	balises de contrôle de confirmation/correction à distance
from remote ECSP (ETBCTRL packet)	depuis l'ECSP distant (paquet ETBCTRL)
to remote ECSP (ETBCTRL packet)	vers l'ECSP distant (paquet ETBCTRL)
control application	application de contrôle
status	statut
sleep request	demande de passage en mode sommeil
own confirmation/correction control flags	balises de contrôle de confirmation/correction considérées

Figure 42 – Diagramme de blocs du diagramme d'états de la fonction mode sommeil

6.8.5.2 Séquence de sommeil

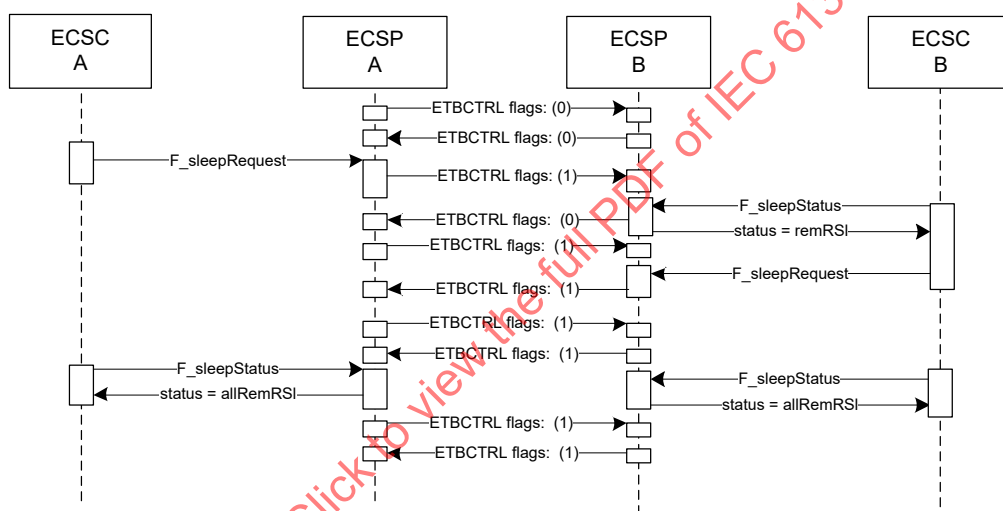
L'interaction pour la fonction mode sommeil entre l'ECSP des différentes rames du train est décrite à la Figure 43 sous forme d'exemple d'un train comportant deux rames A et B.

Initialement, la balise de contrôle pour la fonction mode sommeil (rSI) est configurée à la valeur logique (0)⁴.

La demande de passage en sommeil est initialisée par un ECSC (l'ECSC A dans le présent exemple) en invoquant la primitive de fonction F_sleepRequest. L'ECSP local définira alors sa balise de contrôle 'rSI', signalant ainsi qu'il est préparé pour le mode sommeil, ce qui aura pour conséquence d'informer tous les ECSP (y compris l'ECSP considéré) de la demande. L'ECSC distant obtiendra ces informations avec son interrogation d'état (primitive de fonction F_sleepStatus), puis décidera de se préparer au mode sommeil ou non (c'est une décision de l'application). En cas de préparation pour le mode sommeil, l'ECSC distant doit demander le mode sommeil à son ECSP local, lequel activera la balise de contrôle 'rSI' en conséquence. Si tous les ECSP ont leur balise de contrôle 'rSI' activée, le mode sommeil peut être activé.

NOTE 1 La présente partie de l'IEC 61375 ne définit pas la manière dont l'ECSP gère une rame en mode faible puissance (mode sommeil). Cette tâche relève de la conception de la rame.

NOTE 2 Une interface de fonction mode sommeil possible entre l'ECSP et l'ETBN est définie à l'Annexe E.



IEC

Figure 43 – Diagramme de séquences du protocole de demande de passage en sommeil (exemple)

6.8.5.3 Diagramme d'états

Le diagramme d'états ETBN pour la fonction mode sommeil est présenté à la Figure 44 et dans les Tableaux 56 à 58.

NOTE Un ECSP qui détecte tous les bits 'sleepRequest' activés continue d'envoyer un télégramme ETBCTRL pendant un certain temps afin de s'assurer que tous les autres ETBN l'ont également détecté (état WaitForSleepMode). Après cela, l'ETBN cesse d'émettre en passant en mode de "by-pass actif" (état PrepareForSleepMode). Il y reste un certain temps pour s'assurer que tous les autres ETBN ont bien atteint le même état. Puis il passe en sommeil.

⁴ (0) signifie (FALSE).

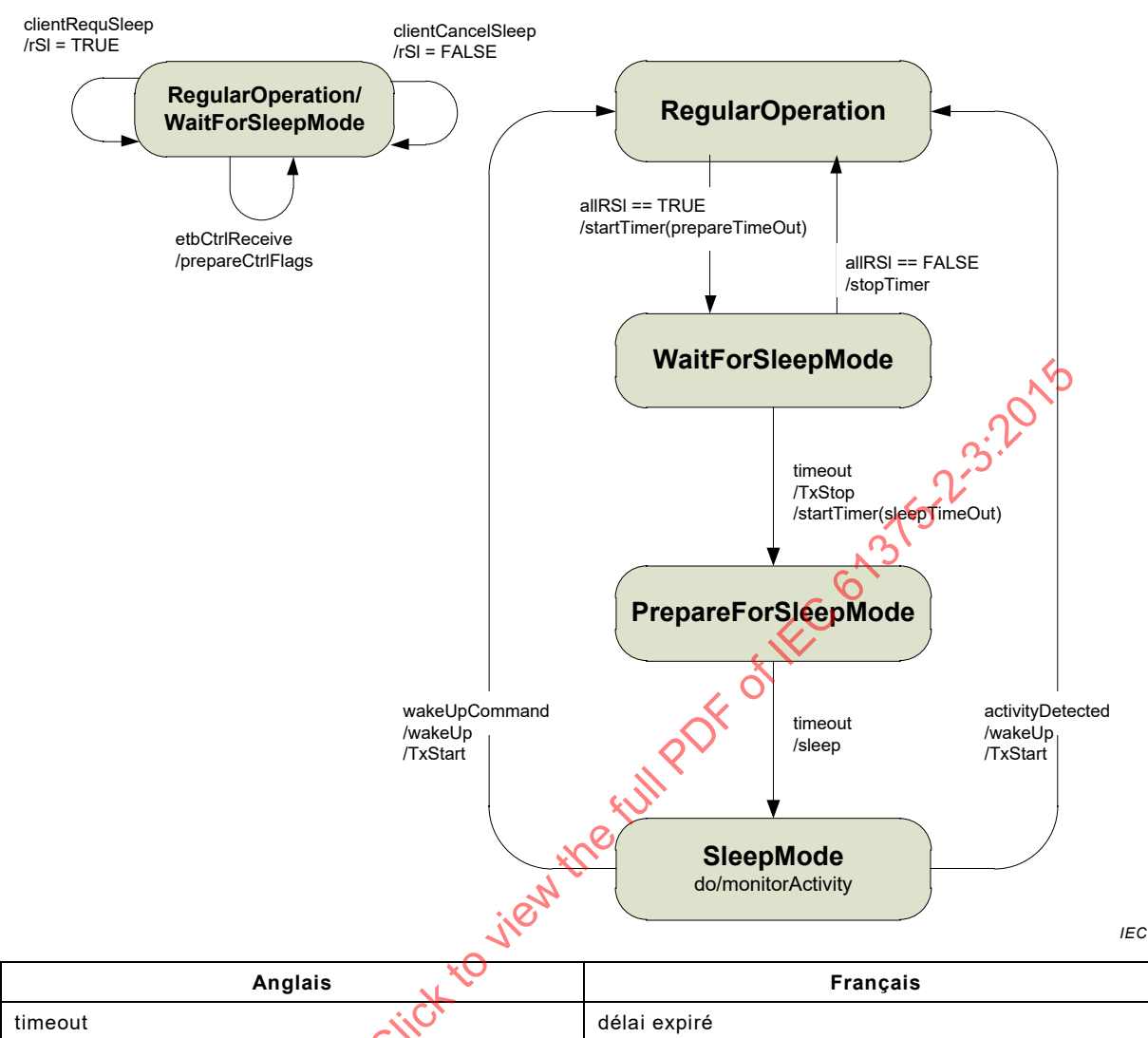


Figure 44 – Diagramme d'états de la fonction mode sommeil

Tableau 56 – Diagramme d'états de la fonction mode sommeil: triggers

Trigger	Description
clientRequSleep	Commande locale de demande de passage en sommeil reçue
clientCancelSleep	Commande locale d'annulation de demande de passage en sommeil reçue
activityDetected	Activité détectée sur l'ETB (trames HELLO)
wakeUpCommand	Commande locale de réveil
etbCtrlReceive	Nouveau télégramme ETBCTRL reçu
timeout	Délai expiré

Tableau 57 – Diagramme d'états du destinataire de MD: guards

Guards	Description
allRSI	AND logique sur les balises de contrôle rSI dans tous les télégrammes ETBCTRL

Tableau 58 – Diagramme d'états de la fonction mode sommeil: actions

Action	Description
prepareCtrlFlags	Récupère la balise de contrôle rSI à partir des télégrammes ETBCTRL reçus et la stocke en local.
startTimer	Démarre un minuteur avec les valeurs d'expiration suivantes: prepareTimeOut = (5,0 ± 1,0) s sleepTimeOut = (10,0 ± 1,0) s
stopTimer	Arrête le minuteur.
TxStop	L'ETBN doit interrompre l'envoi sur l'ETB, mais l'ETBN figurant dans le nœud intermédiaire doit poursuivre la transmission des trames Ethernet entre la direction 1 et la direction 2. L'ETBN doit pouvoir recevoir des trames Ethernet aux fins de détection d'activité sur la ligne. NOTE Une mise en œuvre possible consiste à basculer l'ETBN en mode bypass.
Sleep	Bascule la rame en mode faible puissance (selon la conception de la rame).
TxStart	L'ETBN reprend (active) l'envoi sur l'ETB.
monitorActivity	L'ETBN doit surveiller l'activité sur l'ETB (réception de trames HELLO). A la réception, l'ETBN doit déclencher le trigger 'activityDetected'.
wakeup	Quitte le mode de faible puissance. L'ETBN débute l'inauguration du train (envoi de trames HELLO et TOPOGRAPHY).

6.8.5.4 Défaillance de l'ECSC

En cas de défaillance de l'ECSC (voir 6.3), aucune action spéciale n'est demandée pour l'ECSP.

Annexe A (normative)

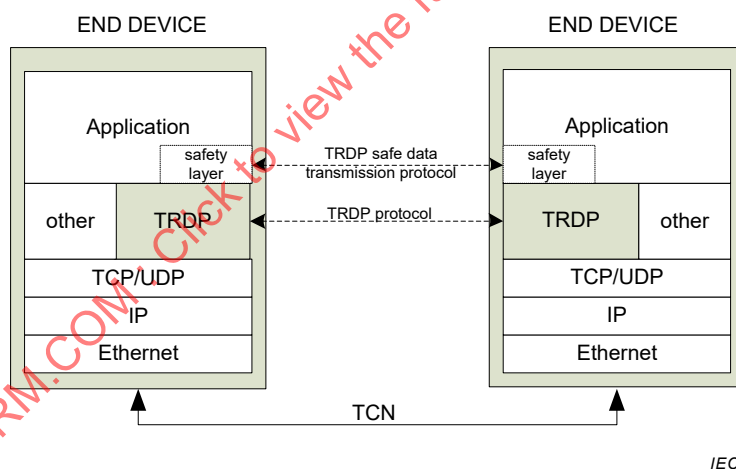
Protocole temps réel de données de train (TRDP)

A.1 Généralités

La présente annexe définit le protocole temps réel de données de train (TRDP) devant être utilisé pour l'échange des données de processus TCN et des données de messagerie TCN sur l'ETB. Le TRDP est exécuté par une couche TRDP placée au-dessus de la couche transport TCP/UDP (voir Figure A.1). En option, cette couche peut être étendue par une couche sécurité qui fournit une transmission de bout en bout des données de processus critiques pour la sécurité et des données de messagerie entre deux appareils terminaux quelconques du réseau.

Toutes les données TRDP sérielles sont définies en big-endian et avec alignement des octets. Le tri éventuel des données utilisateur nécessite d'être effectué par une primitive de service spécifique, appelée depuis la couche application avant d'appeler les primitives de service TRDP pour l'envoi des données. De même, un rétablissement éventuel des données utilisateur nécessite d'être effectué au niveau applicatif après réception de données de la couche TRDP.

NOTE En option, une couche sécurité fait également usage des données au format sériel (big-endian et alignement des octets).



Anglais	Français
END DEVICE	APPAREIL TERMINAL
Application	Application
safety layer	couche sécurité
other	autre
TRDP safe data transmission protocol	Protocole de transmission sécurisé de données TRDP
TRDP protocol	Protocole TRDP

Figure A.1 – Vue d'ensemble de la pile de protocole

A.2 Couches inférieures

A.2.1 Couche liaison de données

Le TRDP utilise l'ETB tel que défini dans l'IEC 61375-2-5 pour la communication des données avec les trains.

A.2.2 Couche réseau

Le TRDP se base sur l'IP (RFC 791) pour l'échange des données entre les fonctions situées dans les différentes rames d'un train.

L'adressage IP des fonctions et appareils est défini dans l'IEC 61375-2-5, ainsi que dans la présente partie de l'IEC 61375. Les affectations d'adresses IP peuvent changer après les inaugurations de trains.

Le transfert de données entre l'ETB et le réseau de rame est assuré par une fonctionnalité passerelle de l'ETBN telle que définie dans l'IEC 61375-2-5. À titre de recommandation, il convient que cette passerelle transfère les données entre l'ETB et le réseau de rame uniquement si les données sont cohérentes avec la vision réelle du réseau central de train (définie par la valeur 'etbTopoCnt') et (éventuellement) la vision opérationnelle du train (définie par la valeur 'opTrnTopoCnt'). La cohérence peut être vérifiée en comparant les valeurs des paramètres 'etbTopoCnt' et 'opTrnTopoCnt' contenus dans les télégrammes TRDP à celles stockées dans l'ETBN. En cas de discordance, il convient de supprimer les télégrammes.

NOTE 1 Le filtrage des données incohérentes avec l'ETBN réduit la probabilité que des données incohérentes soient transférées à la destination. Par contre, cela ne libère pas la destination de la responsabilité d'effectuer un contrôle de la cohérence elle-même.

NOTE 2 Une mise en œuvre possible consistera à définir des règles de pare-feu correspondantes pour le routeur IP au sein de l'ETBN (si un routeur IP est disponible).

A.2.3 Couche transport

Le protocole TRDP utilise les services des protocoles de la couche transport de l'UDP (RFC 768) et TCP (RFC 793) pour la communication des données.

Les données de processus TRDP doivent être envoyées avec UDP. La taille des paquets de données de processus TRDP doit être limitée à la taille d'une trame Ethernet.

Les données de messagerie TRDP peuvent être envoyées avec UDP ou avec TCP. Le choix doit être effectué par l'utilisateur lorsque le transfert des données de messagerie est invoqué.

NOTE Les paquets de données de messagerie dépassant la taille de MTU Ethernet (environ 1 500 octets) peuvent être transmis par TCP afin d'éviter des problèmes de fragmentation de paquet. Cependant, TCP se limite aux communications de point à point.

Pour la communication interopérable de données, les affectations de ports de destination (ports prédéfinis) définies au Tableau A.1 doivent être utilisées:

Tableau A.1 – Affectations de ports UDP/TCP

Protocole	Port de destination
Données de processus	17224
Données de messagerie (UDP)	17225
Données de messagerie (TCP)	17225

Pour recevoir des télégrammes de données de processus et recevoir une notification de données de messagerie UDP, des télégrammes de demande et de confirmation, le port prédéfini doit être utilisé.

Pour recevoir des télégrammes de réponse à des données de messagerie UDP, le port auquel a été envoyée la demande concernée doit être utilisé.

Pour envoyer des télégrammes de données de processus et envoyer une notification de données de messagerie UDP, des télégrammes de demande et de confirmation, un port source privé différent du port prédéfini doit être utilisé.

Pour envoyer des télégrammes de réponse à des données de messagerie UDP, n'importe quel port source différent de celui à partir duquel a été reçue la demande peut être utilisé.

Les connexions TCP doivent être établies entre un port source différent du port prédéfini et le port prédéfini en tant que destinataire.

Des numéros de port prédéfinis différents peuvent être utilisés pour des raisons propres au projet.

Il convient de fournir les numéros de port prédéfinis utilisés à la pile de communication en tant que paramètre de configuration.

A.3 Calcul du FCS TRDP

Pour le calcul de CRC de FCS TRDP, il est fait appel au CRC32 selon l'IEEE 802.3. Le calcul de CRC doit être fait sur les données préparées pour transmission (au format big-endian et avec alignement des octets). Le CRC lui-même doit être joint au format little-endian (octet de poids faible en premier).

Sur la base de la table ci-dessous, l'algorithme suivant dans la notation du langage de programmation C est utilisé, avec la valeur de départ 'FFFFFFFF'H (voir Figure A.2 et Figure A.3):

```

/*****
 * GLOBAL FUNCTION DEFINITIONS
 */

/**
 * @internal
 * Computes and returns a 32-bit FCS.
 *
 * @param buf Input buffer
 * @param len Length of input buffer
 * @param fcs Initial (seed) value for the FCS calculation
 *
 * @return Computed fcs value
 */
static UNSIGNED32 fcs32(const UNSIGNED8 buf[], UNSIGNED32 len, UNSIGNED32 fcs)
{
    UNSIGNED32 i;

    for (i=0; i < len; i++)
    {
        fcs = (fcs >> 8)^fcstab[(fcs ^ buf[i]) & 0xff];
    }
    return (fcs ^ 0xffffffff);
}

```

Figure A.2 – Calcul d'une FCS

```

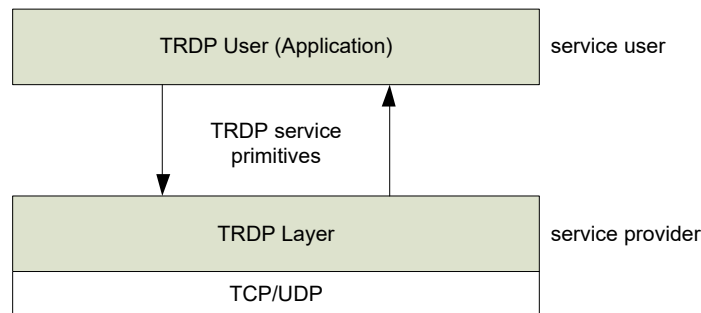
/* Le générateur polynomial de FCS-32:
 * x**0 + x**1 + x**2 + x**4 + x**5 + x**7 + x**8 + x**10 + x**11 +
 * x**12 + x**16 + x**22 + x**23 + x**26 + x**32.
 */
static const UINSGINED32 fcstab[256] =
{
    0x00000000, 0x77073096, 0xee0e612c, 0x990951ba,
    0x076dc419, 0x706af48f, 0xe963a535, 0x9e6495a3,
    0x0edb8832, 0x79dcb8a4, 0xe0d5e91e, 0x97d2d988,
    0x09b64c2b, 0x7eb17cbd, 0xe7b82d07, 0x90bf1d91,
    0x1db71064, 0x6ab020f2, 0xf3b97148, 0x84be41de,
    0x1dad47d, 0x6ddde4eb, 0xf4d4b551, 0x83d385c7,
    0x136c9856, 0x646ba8c0, 0xfd62f97a, 0x8a65c9ec,
    0x14015c4f, 0x63066cd9, 0xfa0f3d63, 0x8d080df5,
    0x3b6e20c8, 0x4c69105e, 0xd56041e4, 0xa2677172,
    0x3c03e4d1, 0x4b04d447, 0xd20d85fd, 0xa50ab56b,
    0x35b5a8fa, 0x42b2986c, 0xdbb8c9d6, 0xacb9f940,
    0x32d86ce3, 0x45df5c75, 0xdcd60dcf, 0xabd13d59,
    0x26d930ac, 0x51de003a, 0xc8d75180, 0xbfbd06116,
    0x21b4f4b5, 0x56b3c423, 0xcfb9599, 0xb8bda50f,
    0x2802b89e, 0x5f058808, 0xc60cd9b2, 0xb10be924,
    0x2f6f7c87, 0x58684c11, 0xc1611dab, 0xb6662d3d,
    0x76dc4190, 0x01db7106, 0x98d220bc, 0xefd5102a,
    0x71b18589, 0x06b6b51f, 0x9fbfe4a5, 0xe8b8d433,
    0x7807c9a2, 0x0f00f934, 0x9609a88e, 0xe10e9818,
    0x7f6a0dbb, 0x086d3d2d, 0x91646c97, 0xe6635c01,
    0x6b6b51f4, 0x1c6c6162, 0x856530d8, 0xf262004e,
    0x6c0695ed, 0x1b01a57b, 0x8208f4c1, 0xf50fc457,
    0x65b0d9c6, 0x12b7e950, 0x8bbeb8ea, 0xfcb9887c,
    0x62dd1ddf, 0x15da2d49, 0x8cd37cf3, 0xfbd44c65,
    0x4db26158, 0x3ab551ce, 0xa3bc0074, 0xd4bb30e2,
    0x4adfa541, 0x3dd895d7, 0xa4d1c46d, 0xd3d6f4fb,
    0x4369e96a, 0x346ed9fc, 0xad678846, 0xda60b8d0,
    0x44042d73, 0x33031de5, 0xaa0a4c5f, 0xdd0d7cc9,
    0x5005713c, 0x270241aa, 0xbe0b1010, 0xc90c2086,
    0x5768b525, 0x206f85b3, 0xb966d409, 0xce61e49f,
    0x5edef90e, 0x29d9c998, 0xb0d09822, 0xc7d7a8b4,
    0x59b33d17, 0x2eb40d81, 0xb7bd5c3b, 0xc0ba6cad,
    0xedb88320, 0x9abfb3b6, 0x03b6e20c, 0x74b1d29a,
    0xead54739, 0x9dd277af, 0x04d8b2615, 0x73dc1683,
    0xe3630b12, 0x94643b84, 0x0dd66a3e, 0x7a6a5aa8,
    0xe40ecf0b, 0x9309ff9d, 0x0a00ae27, 0x7d079eb1,
    0xf00f9344, 0x8708a3d2, 0x1e01f268, 0x6906c2fe,
    0xf762575d, 0x806567cb, 0x196c3671, 0x6e6b06e7,
    0xfed41b76, 0x89d32be0, 0x10da7a5a, 0x67dd4acc,
    0xf9b9df6f, 0x8ebeeeff9, 0x17b7be43, 0x60b08ed5,
    0xd6d6a3e8, 0xa1d1937e, 0x38d8c2c4, 0x4fdff252,
    0xd1bb67f1, 0xa6bc5767, 0x3fb506dd, 0x48b2364b,
    0xd80d2bda, 0xaf0a1b4c, 0x36034af6, 0x41047a60,
    0xdf60efc3, 0xa867df55, 0x316e8eef, 0x4669be79,
    0xcb61b38c, 0xbcc6681a, 0x256fd2a0, 0x5268e236,
    0xcc0c7795, 0xbb0b4703, 0x220216b9, 0x5505262f,
    0xc5ba3bbe, 0xb2bd0b28, 0x2bb45a92, 0x5cb36a04,
    0xc2d7ffa7, 0xb5d0cf31, 0x2cd99e8b, 0x5bdeae1d,
    0x9b64c2b0, 0xec63f226, 0x756aa39c, 0x026d930a,
    0x9c0906a9, 0xeb0e363f, 0x72076785, 0x05005713,
    0x95bf4a82, 0xe2b87a14, 0x7bb12bae, 0x0cb61b38,
    0x92d28e9b, 0xe5d5be0d, 0x7cdcefb7, 0x0bdbdf21,
    0x86d3d2d4, 0xf1d4e242, 0x68ddb3f8, 0x1fda836e,
    0x81be16cd, 0xf6b9265b, 0x6fb077e1, 0x18b74777,
    0x88085ae6, 0xff0f6a70, 0x66063bca, 0x11010b5c,
    0x8f659eff, 0xf862ae69, 0x616bffd3, 0x166ccf45,
    0xa00ae278, 0xd70dd2ee, 0x4e048354, 0x3903b3c2,
    0xa7672661, 0xd06016f7, 0x4969474d, 0x3e6e77db,
    0xaed16a4a, 0xd9d65adc, 0x40df0b66, 0x37d83bf0,
    0xa9bcae53, 0xdeb9ec5, 0x47b2cf7f, 0x30b5ffe9,
    0xbdbdf21c, 0xcabac28a, 0x53b39330, 0x24b4a3a6,
    0xbad03605, 0xcd70693, 0x54de5729, 0x23d967bf,
    0xb3667a2e, 0xc4614ab8, 0x5d681b02, 0x2a6f2b94,
    0xb40bbe37, 0xc30c8ea1, 0x5a05df1b, 0x2d02ef8d
};

```

Figure A.3 – Table FCS

A.4 Interaction entre utilisateur TRDP et couche TRDP

La couche TRDP doit fournir les services de communication de données de processus et de données de messagerie à l'utilisateur TRDP à la couche supérieure (application) comme illustré à la Figure A.4. Cette interface de service sera définie de manière abstraite dans les articles suivants pour la communication des données de processus et des données de messagerie.



IEC

Anglais	Français
TRDP User (Application)	Utilisateur TRDP (Application)
service user	utilisateur de service
TRDP service primitives	Primitives de service TRDP
TRDP Layer	couche TRDP
service provider	fournisseur de service

Figure A.4 – Modèle de service TRDP

A.5 Identificateur de communication (ComId)

Toutes les communications TRDP utilisent un ComId, qui est transmis dans l'en-tête de chaque PDU. Le ComId est un identificateur unique pour la structure de données utilisateur de la PDU.

Avec l'adresse IP source et l'adresse IP destinataire, le ComId forme un identificateur unique de la PDU au sein du train.

Le ComId dépend de l'application, mais il convient de le normaliser dans les profils d'application. Il convient qu'il soit la référence vis-à-vis des paramètres suivants du télégramme des données de processus:

- structure (données) (relation point à point)
- URI source
- URI destinataire
- paramètres de communication (TTL, QoS)
- délai expiré
- comportement relatif à la validité (comportement en cas de données invalides/expiration du délai) (seulement pour les PD_PDU)
- durée de cycle (seulement pour les PD-PDU)
- redondance (la source est redondante) (seulement pour les PD-PDU)
- paramètres de transmission sécurisée de données (si utilisés, seulement pour les PD-PDU)

Le ComId doit être défini sur la valeur zéro (ComId = 0) dans les cas suivants:

- pour les PDU comportant des données utilisateur non spécifiées;
- dans les messages de confirmation de données de messagerie;
- dans les messages d'erreur de données de messagerie;

Les ComId 1..999 ne doivent pas être utilisés par l'application, car ils sont réservés à un usage spécifique tel que défini dans le Tableau A.2.

Tableau A.2 – ComId réservés

ComId	Description
0	PDU non spécifiée
1	télégramme ETBCTRL
2	Message de notification CSTINFO
3	Message de notification CSTINFOCTRL
4 – 9	ComId supplémentaires réservés pour le cadre de communication et le service de contrôle ETB
10	Echo TRDP
11 – 29	ComId supplémentaires réservés pour le cadre de communication et le service de contrôle ETB
30	ComId supplémentaires réservés pour les données statistiques TRDP
31	TRDP – commande de demande de statistiques
32-34	ComId supplémentaires réservés pour les données statistiques TRDP
35	TRDP – données statistiques globales
36	TRDP – données statistiques d'abonnement
37	TRDP – données statistiques d'édition
38	TRDP – données statistiques de redondance
39	TRDP – données statistiques de rattachement
40	TRDP – données statistiques de récepteur UDP
41	TRDP – données statistiques de récepteur TCP
50 – 79	ComId supplémentaires réservés pour le cadre de communication et le service de contrôle ETB
80	Essai de conformité – télégramme de contrôle
81	Essai de conformité – télégramme de statut
82	Essai de conformité – télégramme de demande de confirmation
83	Essai de conformité – télégramme de réponse de confirmation
84	Essai de conformité – télégramme de demande opTrnDir
85	Essai de conformité – télégramme de réponse opTrnDir
86	Essai de conformité – télégramme de demande écho
87	Essai de conformité – télégramme de réponse écho
88	Essai de conformité – télégramme de notification écho
89 – 99	ComId supplémentaires réservés pour l'essai de conformité
100	TTDB – télégramme de statut du répertoire opérationnel de train
101	TTDB – notification du répertoire opérationnel de train
102	TTDB – demande d'informations sur le répertoire de train
103	TTDB – réponse d'informations sur le répertoire de train
104	TTDB – demande d'informations de rame

ComId	Description
105	TTDB – réponse d'informations de rame
106	TTDB – demande d'informations sur le répertoire de réseau de train
107	TTDB – réponse d'informations sur le répertoire de réseau de train
108	TTDB – demande d'informations sur le répertoire opérationnel de train
109	TTDB – réponse d'informations sur le répertoire opérationnel de train
110	TTDB – demande complète d'informations de rame
111	TTDB – réponse complète d'informations de rame
112 – 119	ComId supplémentaires réservés pour la TTDB
120	ECSP – télégramme de contrôle
121	ECSP – télégramme de statut
122	ECSP – demande de confirmation/correction
123	ECSP – réponse de confirmation/correction
124 – 129	ComId supplémentaires réservés pour l'ECSP
130	ETBN – demande de contrôle
131	ETBN – réponse de statut
132	ETBN – demande sur le répertoire de réseau de train
133	ETBN – réponse sur le répertoire de réseau de train
134 – 139	ComId supplémentaires réservés pour l'ETBN
140	DNS TCN – télégramme de demande de résolution (requête)
141	DNS TCN – télégramme de réponse de résolution
142 – 149	ComId supplémentaires réservés pour le DNS TCN
150 – 199	ComId supplémentaires réservés pour le cadre de communication et le service de contrôle ETB
200 – 999	ComId supplémentaires réservés pour les autres normes

A.6 Données de processus

A.6.1 Modèle de communication

Le protocole de données de processus TRDP définit l'échange de PD-PDU pour le transfert de données de processus.

Les PD-PDU doivent être transmises cycliquement ou sur demande entre un éditeur et un ou plusieurs abonnés en utilisant un service TRDP sans connexion et non confirmé.

A.6.2 Rôles

Les partenaires de la communication dans un transfert de données de processus peuvent avoir différents rôles:

éditeur: source des données de processus

abonné: destinataire des données de processus

demandeur: demande à l'éditeur ou aux éditeurs d'éditer leurs données de processus

L'éditeur et le demandeur coïncident dans les modèles de communication de type push.

Un abonné peut également être un demandeur dans les modèles de communication de type pull.

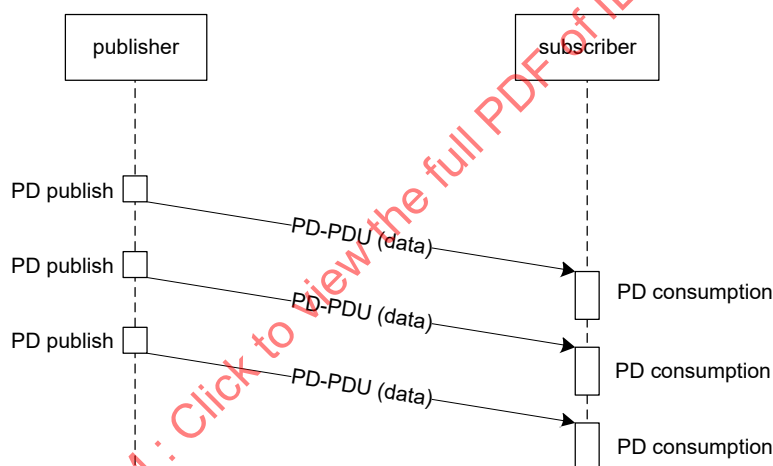
A.6.3 Modèle de communication

A.6.3.1 Modèle de communication de type push

L'échange des données de processus doit prendre en charge le modèle de communication de type push suivant, comme le définit l'IEC 61375-1:

- point à point, sans accusé de réception, le destinataire connaissant la source (voir Figure A.5);
- point à multipoint, cyclique sans accusé de réception, la source connaissant le destinataire, par exemple des groupes de redondance (voir Figure A.6);
- point à multipoint, cyclique sans accusé de réception, la source ne connaissant pas le destinataire (voir Figure A.6).

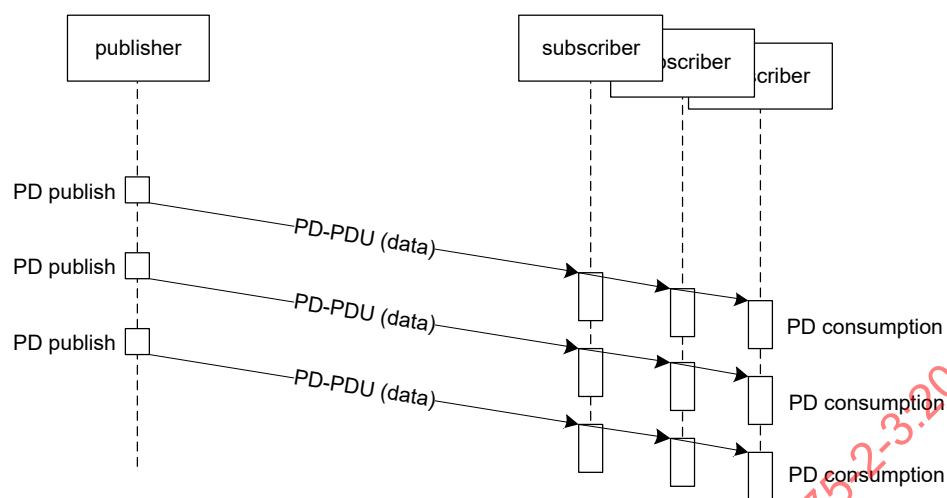
NOTE Du point de vue communication, il n'y aura pas de différence dans un modèle de communication point à multipoint pour savoir lorsque la source connaît le destinataire ou non. Il est donné ici par souci d'exhaustivité seulement.



IEC

Anglais	Français
publisher	éditeur
subscriber	abonné
PD publish	édition de PD (données de processus)
PD-PDU (data)	PD-PDU (données)
PD consumption	consommation de PD (données de processus)

Figure A.5 – Modèle de type push de données de processus (point à point)



IEC

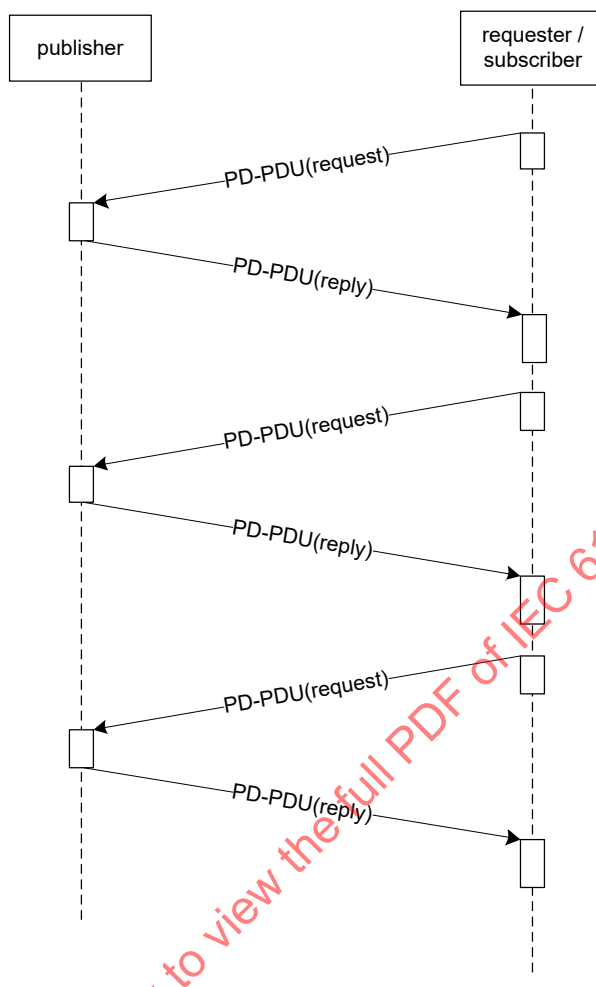
Anglais	Français
publisher	éditeur
subscriber	abonné
PD publish	édition de PD (données de processus)
PD-PDU (data)	PD-PDU (données)
PD consumption	consommation de PD (données de processus)

Figure A.6 – Modèle de type push de données de processus (point à multipoint)

A.6.3.2 Modèle de communication de type pull

L'échange des données de processus doit prendre en charge le modèle de communication de type pull suivant, comme le définit l'IEC 61375-1:

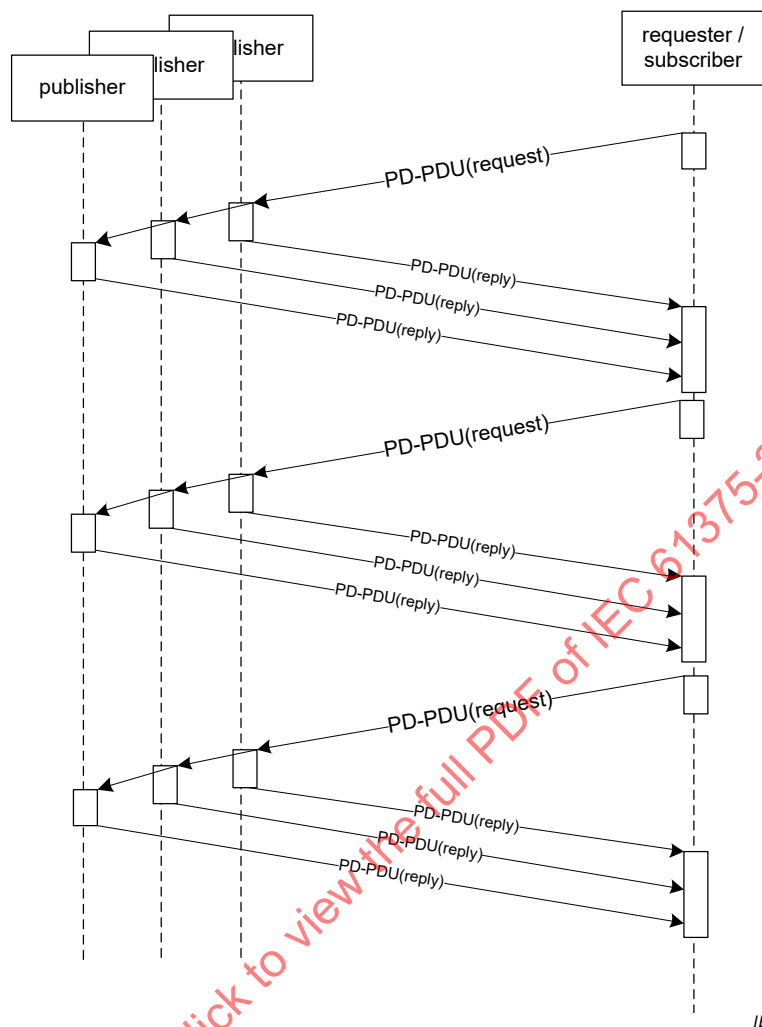
- point à point, sans accusé de réception, le destinataire connaissant la source (voir Figure A.7);
- multipoint à point, sans accusé de réception, le destinataire ne connaissant pas la source (voir Figure A.8);
- point à multipoint, sans accusé de réception, le destinataire connaissant la source (voir Figure A.9). Dans ce cas, un abonné dédié demande à l'éditeur prédéfini d'émettre sa PD-PDU;
- multipoint à multipoint, sans accusé de réception, le destinataire ne connaissant pas la source (voir Figure A.10). Dans ce cas, un abonné dédié demande à un ou plusieurs éditeurs inconnus d'émettre leur PD-PDU.



IEC

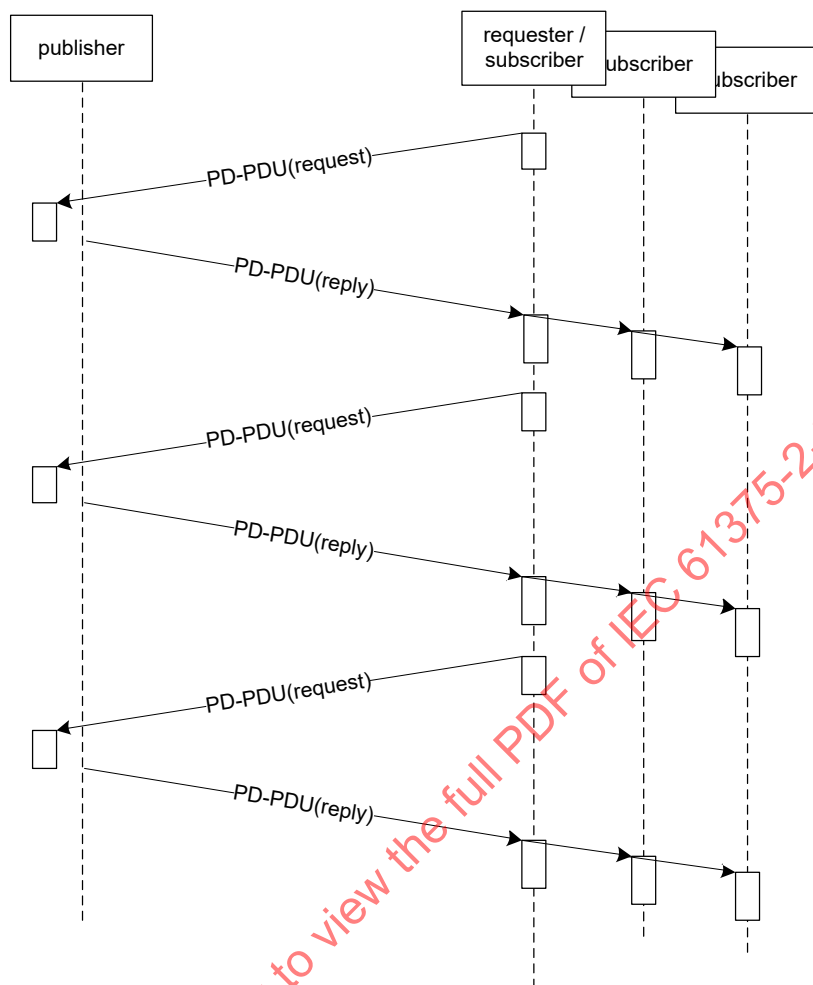
Anglais	Français
publisher	éditeur
requester/subscriber	demandeur/abonné
PD-PDU(request)	PD-PDU (demande)
PD-PDU(reply)	PD-PDU (réponse)

**Figure A.7 – Modèle de type pull de données de processus
(point à point, le destinataire connaissant la source)**



Anglais	Français
publisher	éditeur
requester/subscriber	demandeur/abonné
PD-PDU(request)	PD-PDU (demande)
PD-PDU(reply)	PD-PDU (réponse)

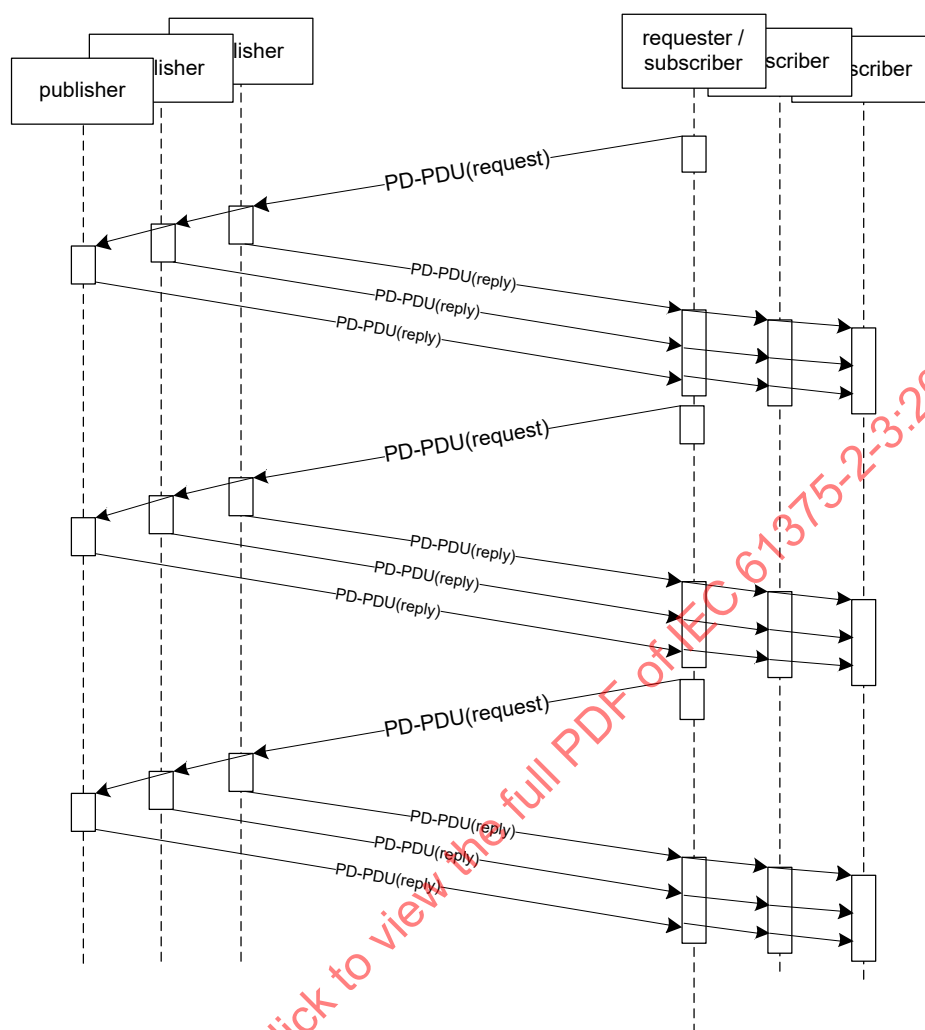
**Figure A.8 – Modèle de type pull de données de processus
(multipoint à point, le destinataire ne connaissant pas la source)**



IEC

Anglais	Français
publisher	éditeur
requester/subscriber	demandeur/abonné
subscriber	abonné
PD-PDU(request)	PD-PDU (demande)
PD-PDU(reply)	PD-PDU (réponse)

**Figure A.9 – Modèle de type pull de données de processus
(point à multipoint, le destinataire connaissant la source)**



IEC

Anglais	Français
publisher	éditeur
requester/subscriber	demandeur/abonné
subscriber	abonné
PD-PDU(request)	PD-PDU (demande)
PD-PDU(reply)	PD-PDU (réponse)

**Figure A.10 – Modèle de type pull de données de processus
(multipoint à multipoint, le destinataire ne connaissant pas la source)**

A.6.4 Adressage

Un éditeur/abonné doit utiliser une adresse de monodiffusion IP pour s'adresser à un abonné/éditeur connu.

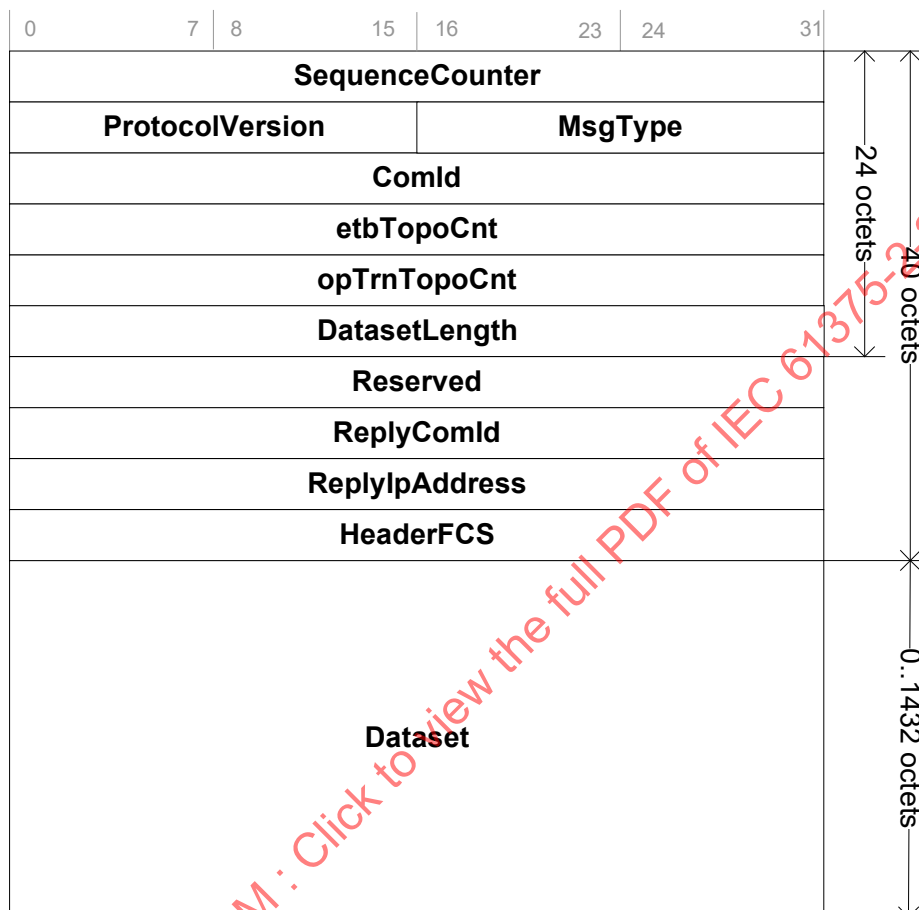
Un éditeur/abonné doit utiliser une adresse de multidiffusion IP pour adresser un groupe d'abonnés/éditeurs connus (ex: groupes de redondance).

Un éditeur/abonné doit utiliser une adresse de multidiffusion IP pour s'adresser à des abonnés/éditeurs inconnus.

Les plages d'adresses pouvant être utilisées pour la communication de données de processus sont définies dans l'IEC 61375-2-5 et en 5.4.5.

A.6.5 PD-PDU

La structure d'une PD-PDU est définie à la Figure A.11 et dans la notation ASN.1. Le Tableau A.3 donne des explications supplémentaires.



IEC

Figure A.11 – PD-PDU

```

PD_PDU ::= RECORD
{
    sequenceCounter  UINT32          -- message sequence counter
    protocolVersion  VERSION         -- protocol version
    msgType          UINT16          -- type of the message
    ComId            UINT32          -- communication identifier
    etbTopoCnt       UINT32          -- etbTopoCnt value
    opTrnTopoCnt     UINT32          -- opTrnTopoCnt value
    datasetLength    UINT32          -- length of the array 'dataset'
    reserved01       UINT32          -- reserved for future use (= 0)
    replyComId       UINT32          -- requested communication identifier
    replyIpAddress   UINT32          -- reply IP address
    headerFcs        UINT32          -- header checksum
    dataset           UINT8 [datasetLength] -- user data set
}
    
```

Tableau A.3 – Paramètres des PD-PDU

Paramètre	Description	Valeur
sequenceCounter	Le compteur de séquence: - doit être géré pour envoyer des télégrammes par ComId/MsgType en direction de chaque demandeur/éditeur; - doit être géré (stocké) pour les télégrammes de processus reçus par SourceIPAddr/ComId/MsgType à chaque éditeur/abonné; - doit être incrémenté à chaque envoi du télégramme de processus. Les télégrammes envoyés en parallèle via des sous-réseaux distincts sont envoyés avec le même compteur de séquence pour détecter les duplications côté récepteur; - peut servir à la surveillance de la couche communication (envoi de données de processus). Pour vérifier que l'application continue de mettre à jour les données de processus (données utilisateur), une surveillance peut être effectuée au moyen du protocole de transmission sécurisé de données (SDTv2, voir Annexe B) ou peut être mise en œuvre par l'application (ex: avec un signe de vie).	Calculée, valeur de départ 0
protocolVersion	La version du protocole doit être ainsi composée: - Octet de poids fort: valeur mainVersion, incrémentée à chaque modification incompatible - Octet de poids faible: valeur subVersion, incrémentée à chaque modification compatible EXEMPLE: '0102'H = version de protocole 1.2	Fixe
msgType	Type de télégramme 'Pr' = Demande de données de processus 'Pp' = Réponse à des données de processus 'Pd' = Données d'un message de données de processus 'Pe' = Données d'un message de données de processus (Erreur) NOTE Encodé au format ASCII afin d'améliorer la lecture avec les outils d'analyse de protocoles ouverts (ex: Wireshark) Le paramètre MsgType 'Pe' étant réservé aux notifications, il convient de ne pas l'envoyer sur le bus.	'5072'H ('Pr') '5070'H ('Pp') '5064'H ('Pd') '5065'H ('Pe')
ComId	Identificateur du dataset utilisateur. Voir également A.5.	Définie par l'utilisateur
etbTopoCnt	Le compteur de topographie ETB: - doit être utilisé (adressage du train) conformément à l'IEC 61375-2-5 (paramètre 'etbTopoCnt'); - doit être défini par l'utilisateur; - doit être défini pour toutes les communications sur l'ETB; - doit être défini si un paramètre opTrnTopoCnt valide est défini; - est facultatif dans tous les autres cas. Il doit être défini sur la valeur 0 (= non valide) s'il n'est pas utilisé.	0..2³²-1 Définie par l'utilisateur
opTrnTopoCnt	Le compteur de topographie opérationnel du train: - doit être utilisé tel que défini en 5.3.3; - doit être défini par l'utilisateur; - doit être défini pour la communication entre les fonctions qui utilisent les informations du répertoire opérationnel de train (ex: opérations sélectives latérales); - doit être défini lorsque l'appareil source a utilisé le répertoire opérationnel de train pour récupérer	

Paramètre	Description	Valeur
	l'adresse IP destinataire (ex: résolution de l'URI 'vcu.leadVeh.anyCst.anyCITrn.ITrn'); est facultatif dans tous les autres cas. Il doit être défini sur la valeur 0 (= non valide) s'il n'est pas utilisé.	
datasetLength	La longueur du dataset: - doit être la longueur des données utilisateur en nombre d'octets sans octets de remplissage; - doit représenter l'information primaire au sujet de la taille des données utilisateur. NOTE En cas de datasets de taille fixe, il s'agit d'une information redondante parce que la longueur des données est déjà définie dans le ComId.	0..1432 Calculée
reserved01	Réservé à une utilisation future	mis à 0
replyComId	Le ComId demandé: - doit être utilisée uniquement dans une demande de données de processus; - doit être utilisé en tant que ComId dans la réponse; - si le paramètre est mis à 0, le ComId de la demande doit être utilisé pour la réponse; - si le paramètre est mis à 0 et que le ComId de la demande est également à 0, la réponse doit être envoyée comme une PDU non spécifiée.	défini par l'utilisateur dans la demande de données de processus, mis à 0 dans les autres types de télégrammes.
replyIpAddress	L'adresse IP de réponse: - doit être utilisée uniquement dans une demande de données de processus; - doit être utilisée en tant qu'adresse IP dans la réponse; - si le paramètre est à 0, l'adresse IP de la source de cette demande doit être utilisée pour la réponse.	défini par l'utilisateur pour une demande, sinon mis à 0.
headerFCS	La séquence de contrôle de la trame d'en-tête: - doit être calculée pour l'en-tête de PD-PDU; - ne doit pas comprendre le HeaderFCS lui-même.	Calculée selon A.3
dataset	Les données utilisateur Lorsque la longueur du dataset utilisateur n'est pas un multiple de 4, il convient d'ajouter les octets d'une valeur de 0 (zéro) jusqu'à ce que ce soit le cas (octets de remplissage).	

A.6.6 Interaction entre application et couche de protocole TRDP

A.6.6.1 Primitives de service

La couche TRDP doit fournir un service de communication de données de processus pour l'utilisateur TRDP à la couche supérieure. Les primitives de service répertoriées dans le Tableau A.4 doivent être fournies.

NOTE Les primitives de service sont définies de manière abstraite afin de ne pas limiter les mises en œuvre. La spécification de l'interface de service ne relève pas du domaine d'application de la présente partie de la norme.

Tableau A.4 – Primitives de service TRDP

Rôle	Primitive de service	Paramètre	Description
éditeur	PD.publish		Utilisateur TRDP → couche TRDP L'utilisateur du service transfère un nouveau paquet de données de processus.
		Handle	Handle pour cette édition, renvoyé par la couche TRDP
		etbTopoCnt	comme défini dans le Tableau A.3
		opTrnTopoCnt	comme défini dans le Tableau A.3
		ComId	comme défini dans le Tableau A.3
		DatasetLength	Longueur réelle des données utilisateur à envoyer comme défini dans le Tableau A.3
		Dataset	Mémoire tampon avec données utilisateur à envoyer comme défini dans le Tableau A.3
		CycleTime	Durée de cycle (txTime) pour l'envoi des données de processus Unité: 10^{-6} s (µs)
		DestinationIpAddress[]	Adresse(s) IP de destination où envoyer les données de processus
		redundant	FALSE signifie que les données de processus sont envoyées immédiatement, TRUE signifie que les données de processus ne sont envoyées que si des ComId redondants sont activés.
	PD.unPublish		Utilisateur TRDP → couche TRDP Interruption de l'édition
		Handle	Handle, fourni par la couche TRDP
	PD.putData		
		Handle	Handle, fourni par la couche TRDP
		DatasetLength	Longueur réelle des données utilisateur à envoyer comme défini dans le Tableau A.3
		Dataset	Mémoire tampon avec données utilisateur à envoyer comme défini dans le Tableau A.3
	PD.activateRed		Utilisateur TRDP → couche TRDP L'utilisateur du service active l'édition des ComId marqués redondants.
	PD.deactivateRed		Utilisateur TRDP → couche TRDP L'utilisateur du service désactive l'édition des ComId marqués redondants.
Demandeur	PD.request		Utilisateur TRDP → couche TRDP La couche TRDP envoie une demande PD-PDU.request à un ou plusieurs éditeurs. Appelée

Rôle	Primitive de service	Paramètre	Description
			cycliquement par l'application.
		Handle	Handle pour cet abonnement du ComId de réponse, retourné par la couche TRDP
		etbTopoCnt	comme défini dans le Tableau A.3
		opTrnTopoCnt	comme défini dans le Tableau A.3
		ComId	ComId à envoyer, comme défini dans le Tableau A.3
		DatasetLength	Longueur des données à envoyer, comme défini dans le Tableau A.3
		DataSet	Longueur des données utilisateur à envoyer, comme défini dans le Tableau A.3
		DestinationIpAddress[]	Adresse(s) IP de destination où envoyer la demande de données de processus
		redundant	FALSE signifie que les données de processus sont envoyées immédiatement, TRUE signifie que les données de processus ne sont envoyées que si des ComId redondants sont activés.
		ReplyComId	comme défini dans le Tableau A.3
		ReplyIpAddress	comme défini dans le Tableau A.3
Abonné	PD.subscribe		Utilisateur TRDP → couche TRDP L'utilisateur s'abonne aux données de processus en utilisant des filtres possibles pour ComId, plage SourceIpAddress, DestinationIpAddress.
		Handle	Handle pour cet abonnement, retourné par la couche TRDP
		etbTopoCnt	comme défini dans le Tableau A.3
		opTrnTopoCnt	comme défini dans le Tableau A.3
		ComId	comme défini dans le Tableau A.3
		DatasetLength	comme défini dans le Tableau A.3
		DataSet	comme défini dans le Tableau A.3
		SourceIpAddress	Adresse IP source, générée à partir de l'URI respectif grâce au DNS. Définit l'adresse IP inférieure, en cas de plage d'adresses IP (voir 5.4.4.6.2). Elle est définie sur 0 si aucun filtrage n'est demandé
		SourceIpAddress2	Définit l'adresse IP supérieure, en cas de plage d'adresses IP (voir 5.4.4.6.2). Adresse IP source, générée à partir de l'URI respectif grâce au DNS. Elle est définie sur 0 si elle n'est

Rôle	Primitive de service	Paramètre	Description
			pas utilisée.
		DestinationIpAddress	Adresse IP destinataire, générée à partir de l'URI respectif grâce au DNS. Elle est définie sur 0 si aucun filtrage n'est demandé
		Subnet	Sélectionne le ou les sous-réseaux auquel ou auxquels s'abonner, un ou tous les réseaux redondants étant désignés
		Timeout	Délai d'expiration (rxTime) pour les données de processus à recevoir Unité: 10^{-6} s (μ s)
	PD.unsubscribe		Utilisateur TRDP → couche TRDP L'utilisateur se désabonne des données de processus en utilisant des filtres possibles pour ComId, SourceIpAddress, DestinationIpAddress.
		Handle	Handle retourné par la couche TRDP dans PD.subscribe
	PD.indicate		Couche TRDP → utilisateur TRDP La couche TRDP indique de nouvelles données de processus ou une expiration du délai. La valeur d'expiration est donnée avec PD.subscribe.
		Handle	Handle retourné par la couche TRDP dans PD.subscribe
		IndType	- données de processus ('Pd') - réponse à données de processus ('Pp') - demande de données de processus ('Pr') - erreur de données de processus ('Pe')
		etbTopoCnt	Compteur topologique reçu, comme défini dans le Tableau A.3
		opTrnTopoCnt	Compteur topologique reçu, comme défini dans le Tableau A.3
		ComId	ComId reçu, comme défini dans le Tableau A.3
		DatasetLength	Taille des données utilisateur reçues, comme défini dans le Tableau A.3
		Dataset	Données utilisateur reçues, comme défini dans le Tableau A.3
		SequenceCounter	Compteur de séquence reçu, comme défini dans le Tableau A.3
		SourceIpAddress	Adresse IP source reçue
		DestinationIpAddress	Adresse IP destinataire reçue, comme défini dans le Tableau A.3
		ReplyComId	ComId de réponse reçu, comme défini dans le Tableau A.3

Rôle	Primitive de service	Paramètre	Description
		ReplyIpAddress	Adresse IP de réponse reçue, comme défini dans le Tableau A.3
		Statut En cas d'erreur TRDP ('Pe'), la valeur est fournie par la couche TRDP. 1 – expiration du délai 6 – absence d'inscription 5 – aucune mémoire 7 – données non valides	0 OK < 0 NOK
	PD.poll		Utilisateur TRDP → couche TRDP L'utilisateur interroge la couche pour recevoir de nouvelles données de processus au lieu d'utiliser l'indication.
		Handle	Handle retourné par la couche TRDP dans PD.subscribe
		IndType	- données de processus ('Pd') - réponse à données de processus ('Pp') - demande de données de processus ('Pr') - erreur de données de processus ('Pe')
		etbTopoCnt	Compteur topologique reçu, comme défini dans le Tableau A.3
		opTrnTopoCnt	Compteur topologique reçu, comme défini dans le Tableau A.3
		ComId	ComId reçu, comme défini dans le Tableau A.3
		DatasetLength	Taille des données utilisateur reçues, comme défini dans le Tableau A.3
		Dataset	Données utilisateur reçues, comme défini dans le Tableau A.3
		SequenceCounter	Compteur de séquence reçu, comme défini dans le Tableau A.3
		SourceIpAddress	Adresse IP source reçue
		DestinationIpAddress	Adresse IP destinataire reçue
		ReplyComId	ComId de réponse reçu, comme défini dans le Tableau A.3
		ReplyIpAddress	Adresse IP de réponse reçue, comme défini dans le Tableau A.3
		Statut En cas d'erreur TRDP ('Pe'), la valeur est fournie par la couche TRDP. 1 – expiration du délai 5 – aucune mémoire 6 – absence d'inscription 7 – données non valides	0 OK < 0 NOK

A.6.6.2 Accès utilisateur aux données de processus

L'utilisateur TRDP doit avoir deux possibilités pour récupérer des données de processus: soit par un mécanisme d'interrogation, généralement utilisé sous forme de tâches utilisateur cycliques, soit par le mécanisme d'indication dans lequel la couche TRDP envoie une notification à l'utilisateur lorsque de nouvelles données sont disponibles ou lorsqu'il y a une expiration du délai.

A.6.6.3 Séquence d'interactions – Modèle de type pull de données de processus

Le diagramme de séquences de la Figure A.12 définit la séquence permise aux primitives de service pour le modèle de type pull de données de processus. La primitive de service PD.request et le télégramme de demande associé ne sont appelés que par un demandeur.

NOTE 1 Les lignes pointillées dans les diagrammes de séquences sont utilisées pour indiquer des options.

Les données de processus sont préparées cycliquement par l'éditeur et doivent être remises à la couche TRDP en appelant la primitive PD.putData.

Pour recevoir une demande, l'éditeur doit s'abonner pour cela.

Tant que le premier télégramme de demande correspondant aux critères de filtrage de l'abonnement n'a pas été reçu, les données associées figurant dans le tampon de réception du répondeur doivent être marquées comme non valides.

Des PDU déjà émises cycliquement doivent pouvoir faire l'objet d'une demande.

Les compteurs de topographie d'un télégramme de demande entrant doivent être vérifiés conformément à A.6.7.

La couche TRDP doit répondre à chaque télégramme de demande entrant en utilisant les données de processus disponibles, l'adresse IP de réponse et le ComId de réponse fournis dans le télégramme de demande.

Si l'adresse IP de réponse du télégramme de demande est 0, l'adresse IP source du télégramme de demande doit être utilisée comme adresse IP destinataire pour la réponse.

Pour recevoir la réponse correspondant à une demande, le demandeur nécessite de s'abonner à cela. L'adresse IP destinataire, l'adresse IP source et la plage d'adresses IP source sont des critères de filtrage possibles.

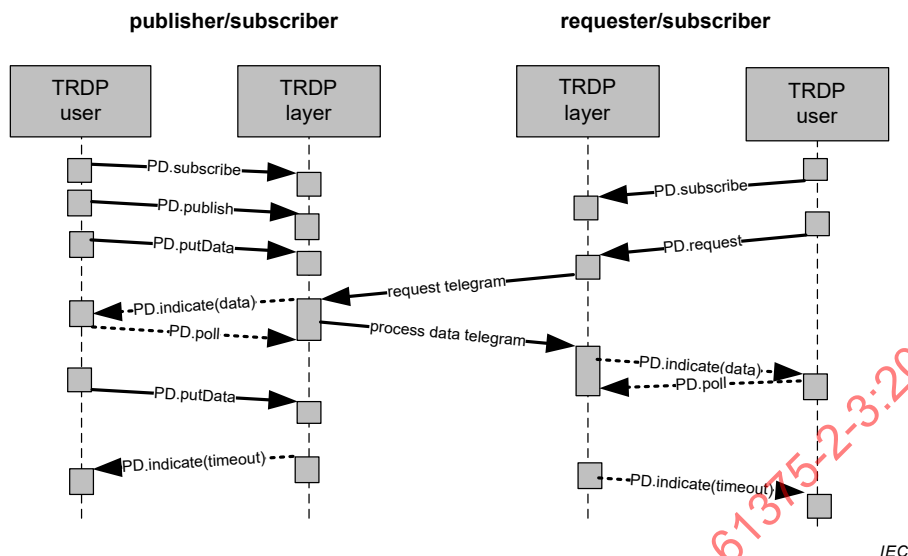
Tant que le premier télégramme de réponse correspondant aux critères de filtrage de l'abonnement n'a pas été reçu, les données associées figurant dans le tampon de réception de l'abonné doivent être marquées comme non valides. Avant d'envoyer le télégramme de demande, les données de réponse associées figurant dans le tampon de réception de l'abonné doivent être marquées comme non valides.

NOTE 2 Cela permet d'éviter que l'utilisateur récupère des données de réponse périmées après avoir demandé de nouvelles données de réponse.

La supervision des expirations de délai à la couche TRDP de l'abonné doit reprendre après l'envoi du télégramme de demande.

La demande peut comporter des données utilisateur.

Avant d'envoyer une demande, la couche TRDP doit comparer les compteurs de topologie soumis avec la demande aux compteurs de topologie réels. Au moins un des cas répertoriés dans le Tableau A.5 pour les compteurs de topographie doit être satisfait.



Anglais	Français
publisher/subscriber	éditeur/abonné
requester/subscriber	demandeur/abonné
TRDP user	utilisateur TRDP
TRDP layer	couche TRDP
request telegram	télégramme de demande
process data telegram	télégramme de données de processus

Figure A.12 – Diagramme de séquences d'interaction pour le modèle de type pull de données de processus

A.6.6.4 Séquence d'interactions – Modèle de type push de données de processus

Le diagramme de séquences de la Figure A.13 définit la séquence permise des primitives de service pour le modèle de type push de données de processus.

Les données de processus sont préparées cycliquement par l'éditeur et doivent être remises à la couche TRDP en appelant la primitive PD.putData.

La couche TRDP de l'éditeur doit envoyer les données dans le cycle configuré à l'adresse configurée.

La couche TRDP de l'éditeur doit envoyer les données uniquement après avoir comparé aux compteurs de topographie réels les compteurs de topographie stockés en local soumis avec l'édition. Au moins un des cas répertoriés dans le Tableau A.5 pour les compteurs de topographie doit être satisfait.

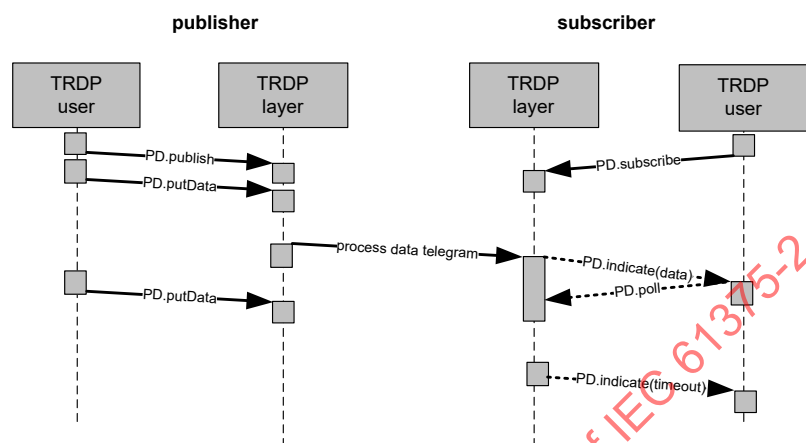
Tout abonné peut s'abonner au télégramme de processus en utilisant le ComId, l'adresse IP destinataire et l'adresse IP source du télégramme de processus comme critères de filtrage possibles.

La supervision des expirations de délai à la couche TRDP de l'abonné doit être démarrée après l'abonnement.

La supervision de l'expiration du délai de la couche TRDP de l'abonné doit reprendre après réception de la PD-PDU correspondante.

La couche TRDP de l'abonné doit comparer les compteurs de topographie du télégramme reçu aux compteurs de topographies réels et aux compteurs de topographie soumis avec l'abonnement. Au moins un des cas répertoriés dans le Tableau A.5 pour les compteurs de topographie doit être satisfait.

Tant que le premier télégramme valide correspondant aux critères de filtrage n'a pas été reçu, les données doivent être marquées comme non valides.



IEC

Anglais	Français
publisher	éditeur
subscriber	abonné
TRDP user	utilisateur TRDP
TRDP layer	couche TRDP
process data telegram	télégramme de données de processus

Figure A.13 – Diagramme de séquences d'interaction pour le modèle de type push de données de processus

A.6.6.5 Gestion de la redondance des données de processus

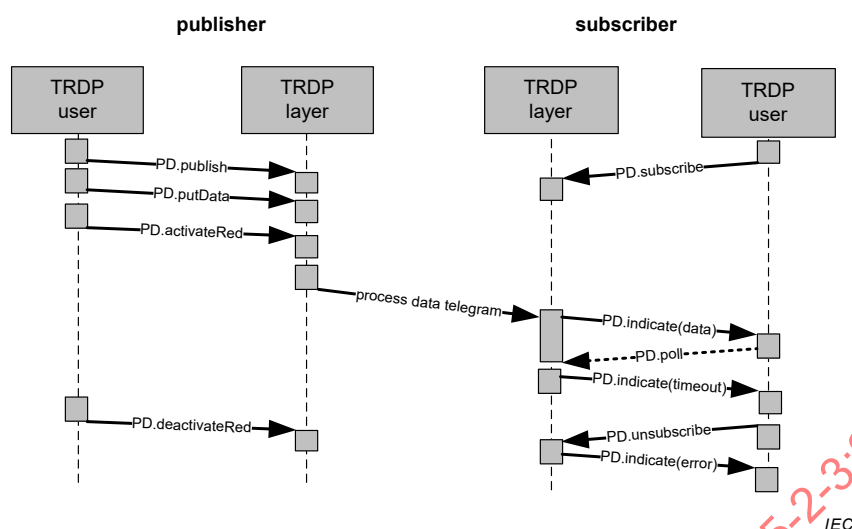
Le diagramme de séquences de la Figure A.14 montre la séquence des primitives de service pour la gestion de la redondance du modèle de type push.

Les primitives de service pour la gestion de la redondance doivent être utilisées de la même façon pour l'éditeur du modèle de type pull et pour l'éditeur du modèle de type push.

Si un appareil redondant passe à l'état de maître de redondance, il doit appeler la primitive PD.activate pour commencer l'édition des données de processus associées aux ComId marqués comme redondants.

Si un appareil redondant passe à l'état d'esclave de redondance, il doit appeler la primitive PD.deactivate pour interrompre l'édition des données de processus associées aux ComId marqués comme redondants.

Au démarrage de TRDP, les éditeurs de données de processus redondantes (identifiées par des ComId marqués comme redondants) doivent être initialisés en mode esclave de redondance (édition désactivée).



Anglais	Français
publisher	éditeur
subscriber	abonné
TRDP user	utilisateur TRDP
TRDP layer	couche TRDP
process data telegram	télégramme de données de processus

Figure A.14 – Diagramme de séquences d'interaction pour la gestion des données de processus redondantes

A.6.7 Contrôle du compteur de topographie

Avant l'envoi d'un télégramme, les compteurs de topographie du télégramme doivent être comparés aux compteurs de topographie réels afin de s'assurer que l'application émettrice partage la vision réelle du réseau central de train et la vision opérationnelle du train.

Après réception d'un télégramme, les valeurs du compteur de topographie doivent être vérifiées afin de s'assurer que l'éditeur/le demandeur et l'abonné partagent une vision du réseau central de train et une vue opérationnelle du train identiques. Pour ce contrôle, les valeurs du compteur de topographie reçues sont comparées aux valeurs du compteur de topographie soumises dans la primitive de service PD.subscribe.

Le contrôle du compteur de topographie est réussi si au moins un des cas répertoriés au Tableau A.5 est vrai.

Tableau A.5 – Contrôle du compteur de topographie

Editeur	Valeurs réelles du compteur de topographie		Valeurs stockées en local du compteur de topographie soumises avec l'édition	
Abonné	Valeurs réelles du compteur de topographie		Valeurs du compteur de topographie du télégramme reçu	
	Valeurs du compteur de topographie du télégramme reçu		Valeurs stockées en local du compteur de topographie soumises avec l'abonnement	
Cas	etbTopoCnt	opTrnTopoCnt	etbTopoCnt	opTrnTopoCnt
1	tout	tout	0	0
2	0	égal à	0	égal à
3	égal à	tout	égal à	0
4	égal à	égal à	égal à	égal à
5 (1)	0	0	égal à	égal à
(1) Le cas s'applique uniquement à un abonné qui compare les valeurs du compteur de topographie du télégramme reçu aux valeurs stockées en local du compteur de topographie soumises avec l'abonnement. Utilisé pour la réception de télégrammes locaux. Légende: 0 = la valeur du compteur de topographie définie sur 0 (inutile) any = le compteur de topographie a une valeur comprise dans la plage $0..(2^{32}-1)$ equal = la valeur reçue et la valeur locale du compteur de topographie sont valides ($1..(2^{32}-1)$) et identiques				

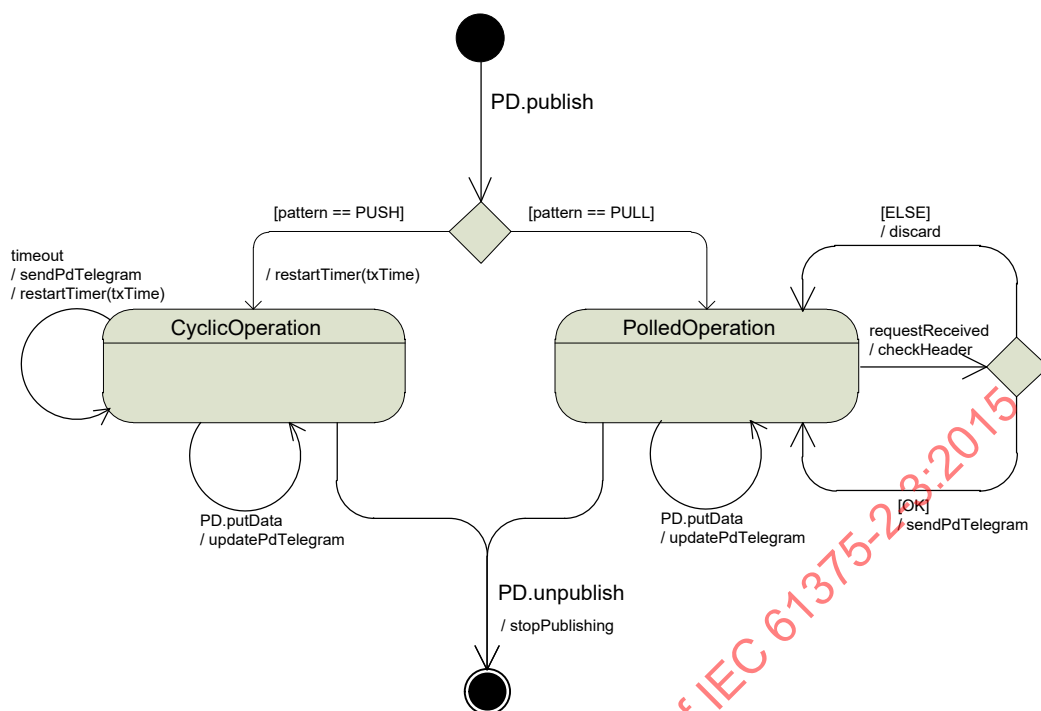
A.6.8 Diagramme d'états**A.6.8.1 Edition de PD-PDU(données)**

Un éditeur doit éditer des PD-PDU(données) lorsqu'une durée de cycle définie s'achève (modèle de type push) et/ou lorsqu'il reçoit une demande d'un abonné dédié ou d'un appareil de communication indépendant.

Pour ne pas surcharger les abonnés, l'éditeur de PD-PDU envoyées par push doit appliquer un mécanisme de modélisation du trafic pour avoir une distribution uniforme des PD-PDU dans le temps.

L'édition de PD-PDU(données) est définie par le diagramme d'états illustré à la Figure A.15.

NOTE En cas de demande de type pull pour une PD-PDU déjà éditée en utilisant le modèle de type push, la réponse correspondante est envoyée à l'adresse donnée par la demande indépendamment de l'adresse et du cycle du modèle de type push.



IEC

Anglais	Français
timeout	délaï expiré
pattern == PUSH	modèle == PUSH
pattern == PULL	modèle == PULL

Figure A.15 – Diagramme d'états de l'éditeur des PD

Les guards, les triggers, les actions et les états sont décrits du Tableau A.6 au Tableau A.9.

Tableau A.6 – Diagramme d'états de l'éditeur des PD: guards

Guards	Description
pattern	Configuration du modèle de communication pour l'échange PD-PDU Valeurs: PUSH PULL
checkTopoCounts	Compare les compteurs de topologie stockés en local soumis avec l'édition aux compteurs de topographie réels. Au moins un des cas répertoriés dans le Tableau A.5 pour les compteurs de topographie est satisfait.

Tableau A.7 – Diagramme d'états de l'éditeur des PD: triggers

Triggers	Description
PD.publish	L'utilisateur a appelé la primitive de service PD.publish.
PD.unpublish	L'utilisateur a appelé la primitive de service PD.unpublish.
PD.putData	Un nouveau paquet de données de processus a été préparé par l'application à des fins d'édition (primitive de service PD.putData).
requestReceived	Un télégramme de demande (type 'Pr') a été reçu.
délaï expiré	Le temps txTime a expiré.

Tableau A.8 – Diagramme d'états de l'éditeur des PD: actions

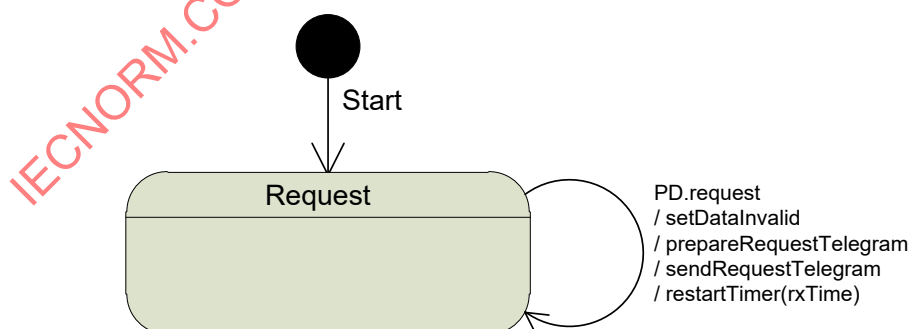
Actions	Description
updatePdTelegram	Prend le paquet de données de processus de l'application et génère un télégramme de données de processus à des fins d'édition.
checkHeader	Contrôle la validité du télégramme de demande de PD reçu: - HeaderFCS - ProtocolVersion Contrôle l'existence d'un éditeur pour ce ComId Si un éditeur existe, contrôle si au moins un des cas énumérés dans le Tableau A.5 pour les compteurs de topographie est rempli. OK = tous les contrôles ont été validés.
sendPdTelegram	Edite (envoie) le télégramme de données de processus.
restartTimer(txTime)	Redémarre le minuteur txTime = durée de cycle d'envoi des PD telle que définie par la primitive de service PD.publish.
stopPublishing	Interruption d'édition sur demande, interruption du minuteur.
discard	Supprime le télégramme.

Tableau A.9 – Diagramme d'états de l'éditeur des PD: états

Etat	Description
CyclicOperation	L'éditeur envoie des PD-PDU déclenchées par le minuteur de façon cyclique.
PolledOperation	L'éditeur envoie des PD-PDU déclenchées par une primitive PD-PDU (demande).

A.6.8.2 Demande d'édition d'une primitive PD-PDU (réponse)

Un abonné dédié ou un appareil de communication indépendant peut demander à un ou plusieurs éditeurs d'envoyer leurs PD-PDU. La demande de PD-PDU est définie dans le diagramme d'états de la Figure A.16.



IEC

Anglais	Français
Start	Lancement
Request	Demande

Figure A.16 – Demandeur du diagramme d'états des PD

Les triggers, les actions et les états sont décrits du Tableau A.10 au Tableau A.13.

Tableau A.10 – Diagramme d'états de l'éditeur des PD: guards

Guards	Description
checkTopoCounts	Compare les compteurs de topologie soumis avec la demande aux compteurs de topographie réels. Au moins un des cas répertoriés dans le Tableau A.5 pour les compteurs de topographie doit être satisfait.

Tableau A.11 – Diagramme d'états du demandeur des PD: triggers

Triggers	Description
PD.request	La primitive de service PD.request a été appelée par l'application.

Tableau A.12 – Diagramme d'états du demandeur des PD: actions

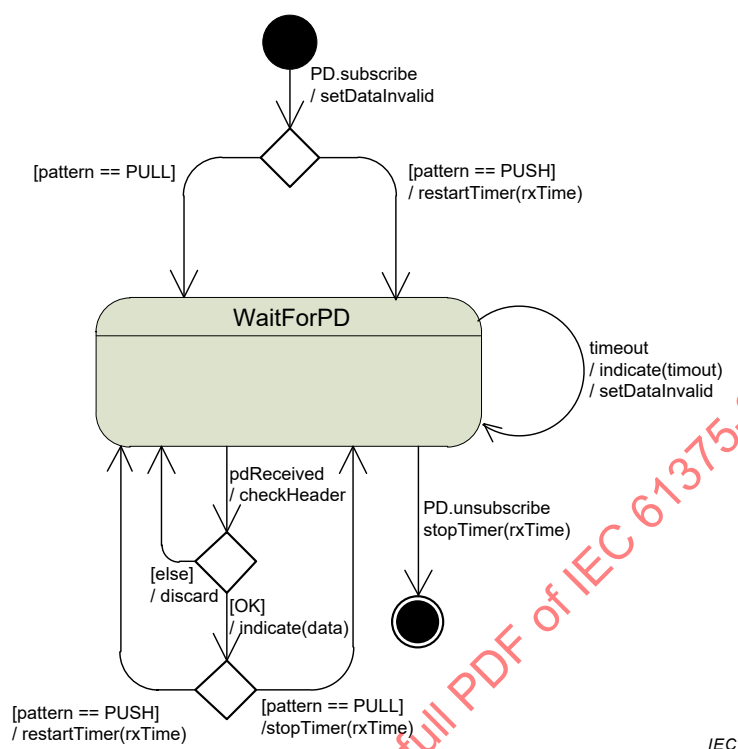
Actions	Description
prepareRequestTelegram	Prépare un télégramme de demande pour un ou plusieurs éditeurs.
sendRequestTelegram	Envoie un télégramme de demande à un ou plusieurs éditeurs.
restartTimer	Démarre (Redémarre) le minuteur temps = rXTime (valeur du délai d'expiration donnée par la primitive de service PD.subscribe).
setDataInvalid	Définit les données relatives comme initialisées/non valides dans le tampon de réception.

Tableau A.13 – Diagramme d'états du demandeur des PD: état

Etat	Description
Demande	Le demandeur envoie un télégramme de demande. L'envoi de ce dernier est déclenché lorsque l'application utilise la primitive PD.request.

A.6.8.3 Réception d'une primitive PD-PDU(réponse/données)

Voir Figure A.17.



Anglais	Français
pattern == PUSH	modèle == PUSH
pattern == PULL	modèle == PULL

Figure A.17 – Abonné au diagramme d'états des PD

Les triggers, les actions et les états sont décrits du Tableau A.14 au Tableau A.17.

Tableau A.14 – Diagramme d'états de l'abonné aux PD: triggers

Triggers	Description
PD.subscribe	La primitive de service PD.subscribe a été appelée par l'application.
pdReceived	Un télégramme de processus ou de demande a été reçu.
Timeout	Le temps rxTime a expiré.
PD.unsubscribe	La primitive de service PD.unsubscribe a été appelée par l'application.

Tableau A.15 – Diagramme d'états de l'abonné aux PD: guards

Guards	Description
pattern	Configuration du modèle de communication pour l'échange PD-PDU Valeurs: PUSH PULL
checkTopoCounts	Compare les compteurs de topographie du télégramme reçu aux compteurs de topographie soumis avec l'abonnement. Au moins un des cas répertoriés dans le Tableau A.5 pour les compteurs de topographie doit être satisfait.

Tableau A.16 – Diagramme d'états de l'abonné aux PD: actions

Actions	Description
indicate (données)	Informe l'application de la réception d'un nouveau télégramme de processus ou de demande (primitive de service PD.indicate).
indicate (délai d'expiration)	Informe l'application de l'expiration du délai (primitive de service PD.indicate).
checkHeader	Contrôle si la FC, la version et le type de l'en-tête sont corrects. Contrôle les critères de filtrage. OK = tous les contrôles ont été validés.
setDataInvalid	Configure les PD comme initialisées/non valides.
stopTimer (minuteur)	Interrompt le minuteur relatif à l'abonnement.
restartTimer (temps)	Redémarre le minuteur. temps = rxTime (valeur du délai d'expiration donnée par la primitive de service PD.subscribe). Le minuteur doit être démarré pour la première fois lorsque l'abonnement est effectué. Pour le modèle de type PUSH, il doit être redémarré à la réception de chaque télégramme de données de processus. Pour le modèle de type PULL, il doit être redémarré à la réception de chaque demande.
discard	Supprime le télégramme.

Tableau A.17 – Diagramme d'états de l'abonné aux PD: état

Etat	Description
WaitForPD	Attend jusqu'à la réception d'un nouveau télégramme de processus.

A.7 Données de messagerie

A.7.1 Modèle de communication

Le protocole TRDP définit l'échange des données de messagerie entre un appelant et un ou plusieurs répondeurs sur un service TRDP confirmé. Celui-ci étant destiné uniquement aux données de messagerie en temps réel, la longueur d'un télégramme est limitée à 64 Ko. L'appelant envoie un message de demande (avec ou sans données utilisateur) au(x) répondeur(s), lesquels doivent retourner un message de réponse (avec ou sans données utilisateur) lorsque l'appelant en question l'exige. L'appelant doit ensuite mettre fin à la réception de la réponse en envoyant une confirmation lorsque le répondeur l'exige.

NOTE Le transfert de données est ainsi "garanti" puisque l'appelant/le répondeur est averti de la bonne réception et du traitement des données de messagerie.

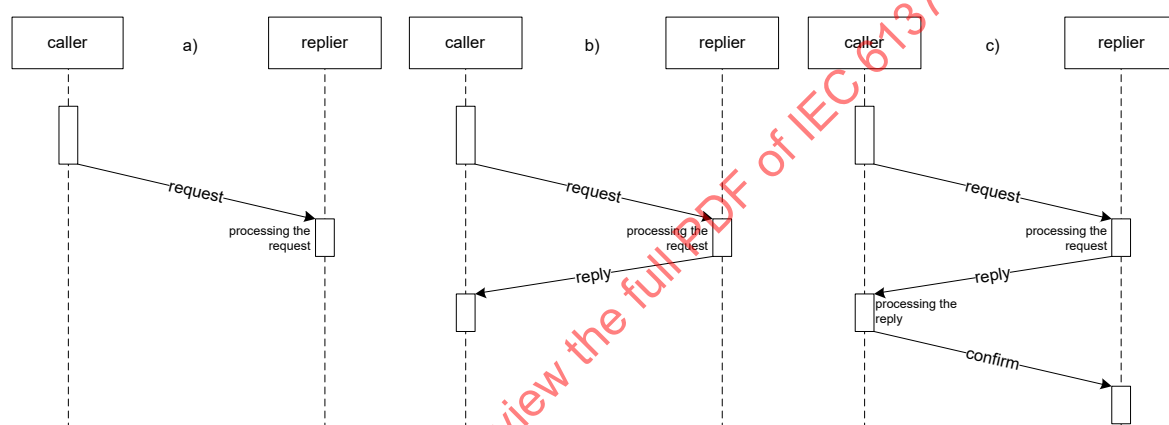
Un appelant doit pouvoir déterminer à partir du type de demande si une réponse est planifiée.

Un répondeur doit pouvoir déterminer à partir du type de réponse si une confirmation est planifiée.

En conséquence, le protocole TRDP doit proposer trois options de transfert de données de messagerie (Figure A.18):

- demande sans réponse ('notification');
- demande avec réponse sans confirmation ('demande sans confirmation');
- demande avec réponse et confirmation ('demande avec confirmation').

Le protocole TRDP doit proposer deux mécanismes de transfert de données de messagerie (UDP ou TCP). Les différentes primitives de service de ces deux options ne doivent pas être mélangées.



IEC

Anglais	Français
caller	appelant
replier	répondeur
request	demande
processing the request	traitement de la demande
reply	réponse
processing the reply	traitement de la réponse
confirm	confirmation

Figure A.18 – Options de transfert de données de messagerie

A.7.2 Rôles

Les partenaires de communication dans un transfert de données de messagerie peuvent avoir différents rôles:

- appelant: la source des données de messagerie dans le modèle de communication de type push
le destinataire des données de messagerie dans le modèle de communication de type pull
- répondeur: la source des données de messagerie dans le modèle de communication de type pull
le destinataire des données de messagerie dans le modèle de communication de type push

A.7.3 Modèle de communication

A.7.3.1 Modèle de communication de type push

Conformément à l'IEC 61375-1, l'échange de données de messagerie doit prendre en charge les modèles de communication de type push ci-dessous (voir la note pour les exceptions):

- a) point à point, apériodique avec acquittement, la source connaît le destinataire;
- b) point à point, apériodique sans acquittement, la source connaît le destinataire;
- c) point à multipoint, apériodique avec acquittement, la source connaît le destinataire;
- d) point à multipoint, apériodique sans acquittement, la source connaît le destinataire;
- e) point à multipoint, apériodique avec acquittement, la source ne connaît pas le destinataire;
- f) point à multipoint, apériodique sans acquittement, la source ne connaît pas le destinataire.

NOTE e) n'est pas exigé dans l'IEC 61375-1.

A.7.3.2 Modèle de communication de type pull

Conformément à l'IEC 61375-1, l'échange de données de messagerie doit prendre en charge les modèles de communication de type pull ci-dessous (voir la note pour les exceptions):

- a) point à point, apériodique avec acquittement, le destinataire connaissant la source;
- b) point à point, apériodique sans acquittement, le destinataire connaissant la source;
- c) point à multipoint, apériodique avec acquittement, le destinataire connaissant la source;
- d) point à multipoint, apériodique sans acquittement, le destinataire connaissant la source;
- e) point à multipoint, apériodique au premier acquittement, le destinataire ne connaissant pas la source;
- f) point à multipoint, apériodique sans acquittement, le destinataire ne connaissant pas la source.

NOTE e) n'est pas exigé dans l'IEC 61375-1.

A.7.4 Adressage

L'appelant doit utiliser une adresse de mono- ou multidiffusion IP pour l'adressage au(x) répondeur(s) connu(s).

L'appelant doit utiliser une adresse de multidiffusion IP pour l'adressage aux répondeurs inconnus.

L'appelant peut utiliser une adresse de multidiffusion IP pour l'adressage à un groupe de redondance de répondeurs connus.

Le répondeur doit répondre à l'adresse de monodiffusion de l'appelant.

Les plages d'adresses pouvant être utilisées pour la communication de données de messagerie sont définies dans l'IEC 61375-2-5.

A.7.5 MD-PDU

La structure d'une MD-PDU est définie à la Figure A.19 puis dans la notation ASN.1. Le Tableau A.18 donne des explications supplémentaires.

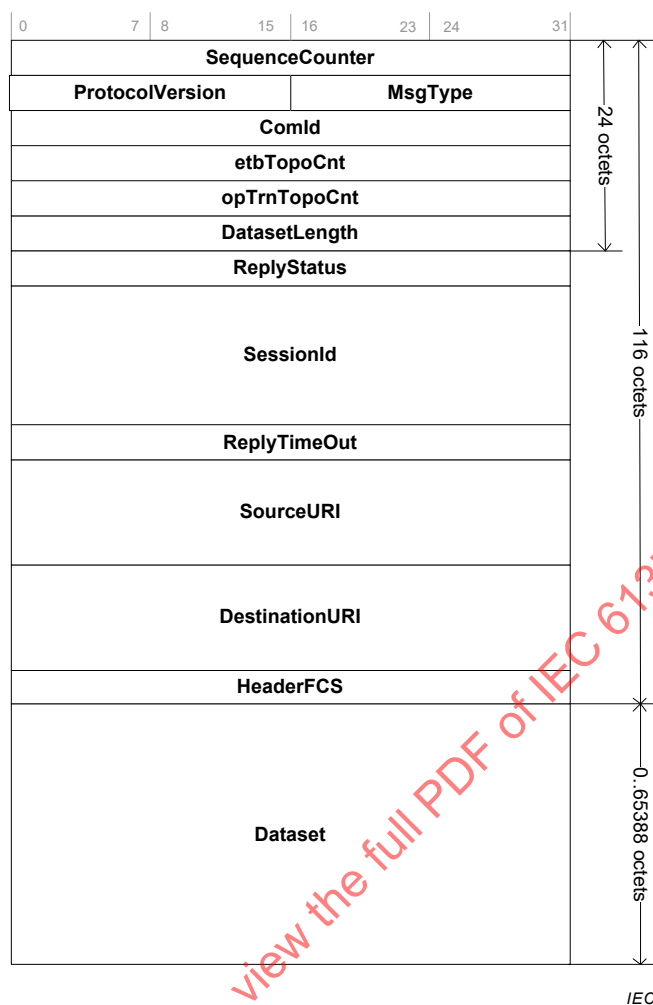


Figure A.19 – MD-PDU

```

MD_PDU ::= RECORD
{
    sequenceCounter  UINT32          -- message sequence counter
    protocolVersion  VERSION         -- protocol version
    msgType          UINT16          -- type of the message
    comId            UINT32          -- communication identifier
    etbTopoCnt       UINT32          -- etbTopoCnt value
    opTrnTopoCnt     UINT32          -- opTrnTopoCnt value
    datasetLength    UINT32          -- length of the array 'dataset'
    replyStatus      INT32           -- reply status indication
    sessionId        UINT32[4]       -- session identifier
    replyTimeout     UINT32          -- reply timeout
    sourceUri        CHAR[32]        -- source URI (user part)
    destinationUri   CHAR[32]        -- destination URI (user part)
    headerFcs        UINT32          -- header checksum
    dataset          ARRAY[] of UINT8 -- user data set
}

```

Tableau A.18 – Paramètres des MD-PDU

Paramètre	Description	Valeur
sequenceCounter	Le compteur de séquences - doit être incrémenté à chaque répétition du message (est remis à 0 en cas de débordement); - doit être renvoyé avec le message de réponse; - doit commencer par la valeur: 0	Calculée
protocolVersion	La version du protocole doit être ainsi composée: - octet de poids fort: valeur MainVersion, incrémentée à chaque modification incompatible; - octet de poids faible: valeur SubVersion, incrémentée à chaque modification compatible. EXEMPLE: '0102'H = version de protocole 1.2	Fixe
msgType	Type de télégramme 'Mn' = notification (demande sans réponse) 'Mr' = demande de MD avec réponse 'Mp' = réponse de MD sans confirmation 'Mq' = réponse de MD avec confirmation 'Mc' = confirmation de MD 'Me' = erreur de MD NOTE Encodé au format ASCII afin d'améliorer la lecture avec les outils d'analyse de protocoles ouverts (ex: Wireshark)	'4D6E'H ('Mn') '4D72'H ('Mr') '4D70'H ('Mp') '4D71'H ('Mq') '4D63'H ('Mc') '4D65'H ('Me')
ComId	Identificateur du dataset utilisateur. Voir également A.5. Une PDU non spécifiée (ComId=0) doit être prise en charge. NOTE L'adresse IP source pourrait être modifiée par la NAT, voir IEC 61375-2-5.	Définie par l'utilisateur
etbTopoCnt	Le compteur de topographie: - doit être utilisé (adressage du train) conformément à la partie de la norme IEC 61375-2-5 (paramètre 'etbTopoCnt'); - doit être défini par l'utilisateur; - doit être défini pour toutes les communications sur l'ETB; - doit être défini si un paramètre opTrnTopoCnt valide est défini; - est facultatif dans tous les autres cas. Il doit être défini sur la valeur 0 (= non valide) s'il n'est pas utilisé.	0..2 ³² -1 Définie par l'utilisateur
opTrnTopoCnt	Le compteur de topographie: - doit être utilisé tel que défini en 5.3.3; - doit être défini par l'utilisateur; - doit être défini pour la communication entre les fonctions qui utilisent les informations du répertoire opérationnel de train (ex: opérations sélectives latérales); - doit être défini lorsque l'appareil source a utilisé le répertoire opérationnel de train pour récupérer l'adresse IP destinataire (ex: résolution de l'URI 'vcu.leadVeh.anyCst.anyCITrn.ITrn'); - est facultatif dans tous les autres cas. Il doit être défini sur la valeur 0 (= non valide) s'il n'est pas utilisé.	

Paramètre	Description	Valeur
datasetLength	Longueur du dataset utilisateur en octets, hors octets de remplissage.	0..65 388 Calculée
replyStatus	La valeur d'état doit être définie par le répondeur afin de rendre compte du résultat d'exécution d'un message de demande ou par l'appelant lors de l'envoi d'une confirmation. Le résultat d'exécution est communiqué par l'application à l'origine de la réponse puis transmis à l'application à l'origine de la demande avec le message de réponse. En cas de réponse d'erreur TRDP ('Me'), la valeur est fournie par le protocole TRDP: • -1 – réservé • -2 – abandon de la session • -3 – aucune instance de répondeur (côté répondeur) • -4 – aucune mémoire (côté répondeur) • -5 – aucune mémoire (locale) • -6 – aucune réponse • -7 – réponses partielles • -8 – aucune confirmation • -9 – réservé • -10 – échec de l'envoi	< 0: NOK 0: OK > 0: état de la réponse utilisateur
sessionId	L'identification de la session: • doit identifier une session "demande-réponse" ou "demande-réponse-confirmation", laquelle se décompose en une session d'appelant et une session de réponse; • doit identifier une session "demande-erreur" lorsque la couche TRDP du répondeur renvoie un message d'erreur; • doit être calculée du côté appelant et réutilisée du côté répondeur (récepteur); • doit être utilisée du côté appelant pour associer un message de réponse ou d'erreur au message de demande initial; • doit être utilisée du côté répondeur pour identifier une retransmission de la demande lorsque le message de réponse n'a pas été reçu ainsi que le message de confirmation; • doit être un UUID conformément à la RFC 4122, version en cours.	Calculée
replyTimeout	La valeur d'expiration du délai de réponse doit être établie pour définir le temps de réponse planifié dans une session de demande/réponse. Unité: 10^{-6} s (μs) Doit être définie sur 0 pour les télégrammes de type 'Mn', 'Mp', 'Mc' et 'Me'	1 .. $(2^{32}-1)$ 0 == temps infini
sourceURI	L'URI source: • doit être utilisé pour l'adressage fonctionnel; • doit être une chaîne terminée par un caractère nul; • doit utiliser des octets de remplissage finaux définis sur 0; • doit contenir uniquement la "section utilisateur" sans la "section hôte" et le symbole "@"; • peut être une chaîne vide (toutes les valeurs définies sur 0).	
destinationURI	L'URI destinataire:	

Paramètre	Description	Valeur
	- doit être utilisé pour l'adressage fonctionnel - doit être une chaîne terminée par un caractère nul; - doit utiliser des octets de remplissage finaux définis sur 0; - doit contenir uniquement la "section utilisateur" sans la "section hôte" et le symbole "@"; - peut être une chaîne vide (toutes les valeurs définies sur 0).	
headerFCS	La séquence de contrôle de la trame d'en-tête: - doit être calculée pour l'en-tête de MD; - ne doit pas intégrer le paramètre HeaderFCS.	Calculée selon A.3
dataset	Données utilisateur. Lorsque la longueur du dataset utilisateur n'est pas un multiple de 4, des octets d'une valeur de 0 (zéro) doivent être ajoutés jusqu'à ce que ce soit le cas (octets de remplissage).	

A.7.6 Interaction entre l'application et la couche TRDP

A.7.6.1 Primitives de service

La couche TRDP doit fournir des services pour la communication de données de messagerie à l'utilisateur TRDP. Les primitives de service répertoriées dans le Tableau A.19 et le Tableau A.20 doivent être fournies.

NOTE Les primitives de service sont définies de façon abstraite afin de ne pas limiter la mise en œuvre. La spécification de l'interface de service ne relève pas du domaine d'application de la présente partie de la norme.

Tableau A.19 – Primitives de service TRDP: appelant

Primitive de service	Paramètres	Direction/Description
MD.request		Utilisateur TRDP → couche TRDP L'utilisateur TRDP souhaite envoyer un message de demande (notification seule ou qui attend une réponse).
	MsgType	- Envoie un message de notification ('Mn'). - Envoie un message de demande ('Mr').
	UserIdentifier	Identificateur renvoyé avec la primitive MD.indication. Il est utilisé par l'utilisateur pour associer la primitive MD.request à la primitive MD.indication.
	SessionId	Identificateur de session utilisé et renvoyé par la couche TRDP.
	TransProtocol	Protocole de couche transport (UDP ou TCP).
	NumRepliers	Nombre de répondeurs lorsque le message de demande est envoyé à un ou plusieurs répondeurs connus (ex: '1' s'il est envoyé à un répondeur). Doit être défini sur '0' lorsque le nombre de répondeurs est inconnu ou qu'un message de notification est envoyé.
	etbTopoCnt	comme défini dans le Tableau A.18
	opTrnTopoCnt	comme défini dans le Tableau A.18
	ComId	comme défini dans le Tableau A.18
	DatasetLength	comme défini dans le Tableau A.18
	Dataset	comme défini dans le Tableau A.18

Primitive de service	Paramètres	Direction/Description
	SourceURI	comme défini dans le Tableau A.18
	DestinationURI	comme défini dans le Tableau A.18
	DestinationIpAddress	Adresse IP destinataire, générée à partir de l'URI respectif grâce au DNS.
	ReplyTimeOut	comme défini dans le Tableau A.18
	MaxNumRetries	Nombre maximal de nouvelles tentatives si NumRepliers == 1. Plage de valeurs 0..2
MD.indicate		Couche TRDP → utilisateur TRDP La couche TRDP envoie des informations sur un événement (réception de données, expiration du délai ou erreur).
	IndType	- Réception d'un message de réponse ('Mp'). - Réception d'un message de requête ('Mq'). - Erreur de la couche TRDP ('Me').
	UserIdentifier	Identificateur envoyé avec la primitive MD.request.
	SessionId	Identificateur de session utilisé par la couche TRDP.
	NoOfRepl	Nombre de réponses reçues.
	etbTopoCnt	Valeur obtenue à partir du message reçu. Elle est définie sur 0 si elle est inconnue (renvoi d'erreur).
	opTrnTopoCnt	Valeur obtenue à partir du message reçu. Elle est définie sur 0 si elle est inconnue (renvoi d'erreur).
	ComId	comme défini dans le Tableau A.18
	DatasetLength	comme défini dans le Tableau A.18
	User Dataset	comme défini dans le Tableau A.18
	SourceURI	comme défini dans le Tableau A.18
	DestinationURI	comme défini dans le Tableau A.18
	SourceIpAddress	Adresse IP source.
	DestinationIpAddress	Adresse IP destinataire.
	ReplyTimeOut	comme défini dans le Tableau A.18
	ReplyStatus	comme défini dans le Tableau A.18
MD.confirm		Utilisateur TRDP → couche TRDP L'utilisateur TRDP souhaite envoyer un message de confirmation.
	SessionId	Identificateur de session renvoyé par la primitive MD.indication. Il est utilisé par la couche TRDP pour associer la primitive MD.confirm à la session.
	ReplyStatus	comme défini dans le Tableau A.18
MD.abort		Utilisateur TRDP → couche TRDP L'utilisateur TRDP souhaite abandonner une session ouverte (par exemple du fait de l'inauguration du TCN).
	SessionId	Identificateur de session utilisé par la couche TRDP.

Tableau A.20 – Primitives de service TRDP: répondeur

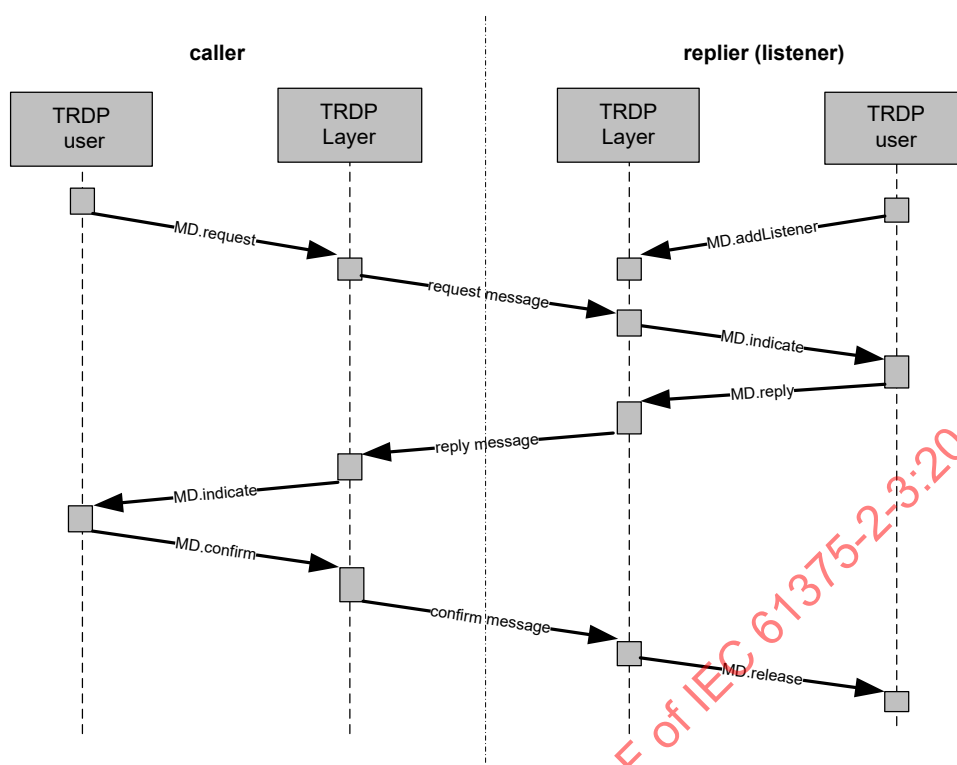
Primitive de service	Paramètres	Direction/Description
MD.addListener		Utilisateur TRDP → couche TRDP L'utilisateur TRDP souhaite préparer la couche TRDP à la réception de messages de notification/demande (par exemple en réunissant certains groupes de multidiffusion définis et en fournissant un tampon de télégramme). Des filtres peuvent être définis pour les paramètres ComId, SourceIpAddress, plage SourceIpAddress, DestinationIpAddress, SourceURI et DestinationURI.
	Handle	Handle pour ce récepteur, renvoyé par la couche TRDP.
	ComId	Seules les notifications/demandes avec ce(s) paramètre(s) sont acceptées.
	etbTopoCnt	Valeur réelle de ce paramètre.
	opTrnTopoCnt	Valeur réelle de ce paramètre.
	SourceURI[]	Comme défini dans le Tableau A.18, seules les notifications/demandes provenant de cet (ces) URI source sont acceptées. Seule la section utilisateur est prise en considération, sauf en cas d'adressage de groupe où la section hôte l'est également. Voir également A.7.6.3.
	DestinationURI	Comme défini dans le Tableau A.18, seules les notifications/demandes provenant de cet ou de ces URI destinataires sont acceptées. Seule la section utilisateur est prise en considération, sauf en cas d'adressage de groupe où la section hôte l'est également. Voir également A.7.6.3.
	SourceIpAddress	Adresse IP source, générée à partir de l'URI respectif grâce au DNS. Définit l'adresse IP inférieure, en cas de plage d'adresses IP (voir 5.4.4.6.2).
	SourceIpAddress2	Définit l'adresse IP supérieure, en cas de plage d'adresses IP (voir 5.4.4.6.2). Adresse IP source, générée à partir de l'URI respectif grâce au DNS. Elle est définie sur 0 si elle n'est pas utilisée.
	DestinationIpAddress	Adresse IP destinataire, générée à partir de l'URI respectif grâce au DNS.
MD.updateListener	UserIdentifier	Identificateur à renvoyer avec la primitive MD.indicate.
		Utilisateur TRDP → couche TRDP Suite à une modification de la topologie du train, l'utilisateur TRDP souhaite que la couche TRDP mette à jour les paramètres du récepteur puis les paramètres relatifs à l'appartenance à un groupe de multidiffusion.
	Handle	Handle pour ce récepteur.
	etbTopoCnt	Valeur réelle de ce paramètre.
	opTrnTopoCnt	Valeur réelle de ce paramètre.
	SourceIpAddress	Adresse IP source, générée à partir de l'URI respectif grâce au DNS. Définit l'adresse IP inférieure, en cas de plage d'adresses IP (voir 5.4.4.6.2).

Primitive de service	Paramètres	Direction/Description
	SourcelpAddress2	Définit l'adresse IP supérieure, en cas de plage d'adresses IP (voir 5.4.4.6.2). Adresse IP source, générée à partir de l'URI respectif grâce au DNS. Elle est définie sur 0 si elle n'est pas utilisée.
	DestinationIpAddress	Adresse IP destinataire, générée à partir de l'URI respectif grâce au DNS.
MD.remListener		Utilisateur TRDP → couche TRDP L'utilisateur TRDP souhaite que la couche TRDP supprime un récepteur.
	Handle	Handle renvoyée par la couche TRDP à la primitive MD.addListener.
MD.indicate		Couche TRDP → utilisateur TRDP La couche TRDP envoie des informations sur un événement.
	IndType	- Réception d'un message de demande ('Mr'). - Réception d'un message de notification ('Mn'). - Erreur de la couche TRDP ('Me').
	UserIdentifier	Identificateur envoyé avec la primitive MD.addListener.
	SessionId	Identificateur de la session créée par la couche TRDP.
	etbTopoCnt	Valeur obtenue à partir du message reçu. Elle est définie sur 0 si elle est inconnue (renvoi d'erreur).
	opTrnTopoCnt	Valeur obtenue à partir du message reçu. Elle est définie sur 0 si elle est inconnue (renvoi d'erreur).
	ComId	Valeur obtenue à partir du message reçu. Elle est définie sur 0 si elle est inconnue (renvoi d'erreur).
	DatasetLength	comme défini dans le Tableau A.18
	Dataset	comme défini dans le Tableau A.18
	SourceURI	comme défini dans le Tableau A.18
	DestinationURI	comme défini dans le Tableau A.18
	SourcelpAddress	Adresse IP source.
	DestinationIpAddress	Adresse IP destinataire. Peut également être une adresse de multidiffusion IP.
	ReplyTimeOut	comme défini dans le Tableau A.18
	ReplyStatus	comme défini dans le Tableau A.18
MD.reply		Utilisateur TRDP → couche TRDP L'utilisateur TRDP souhaite envoyer un message de réponse.
	MsgType	- Envoie un message de réponse ('Mp') - Envoie un message de requête ('Mq')
	SessionId	Identificateur de session indiqué avec la primitive MD.indication. Il est utilisé par la couche TRDP pour associer la primitive MD.reply à la session de réponse.
	ComId	comme défini dans le Tableau A.18
	DatasetLength	comme défini dans le Tableau A.18
	UserDataset	comme défini dans le Tableau A.18

Primitive de service	Paramètres	Direction/Description
	SourceURI	comme défini dans le Tableau A.18
	SourcelpAddress	Adresse IP source, générée à partir de l'URI respectif grâce au DNS.
	ReplyTimeOut	comme défini dans le Tableau A.18
	ReplyStatus	comme défini dans le Tableau A.18
MD.release		Couche TRDP → utilisateur TRDP La couche TRDP envoie des informations sur la réception d'un message de confirmation, une expiration de délai ou une erreur relative.
	MsgType	- Réception d'un message de confirmation ('Mc'). - Erreur de la couche TRDP ('Me').
	SessionId	Identificateur de session utilisé par la couche TRDP.
	etbTopoCnt	Valeur obtenue à partir du message reçu. Elle est définie sur 0 si elle est inconnue (renvoi d'erreur).
	opTrnTopoCnt	Valeur obtenue à partir du message reçu. Elle est définie sur 0 si elle est inconnue (renvoi d'erreur).
	ComId	comme défini dans le Tableau A.18
	SourceURI	comme défini dans le Tableau A.18
	SourcelpAddress	Adresse IP source.
	ReplyStatus	comme défini dans le Tableau A.18

A.7.6.2 Séquence d'interaction

Le diagramme de séquences de la Figure A.20 définit la séquence des primitives de service.



IEC

Anglais	Français
caller	appelant
replier (listener)	répondeur (récepteur)
TRDP user	utilisateur TRDP
TRDP Layer	couche TRDP
request message	message de demande
reply message	message de réponse
confirm message	message de confirmation

Figure A.20 – Diagramme de séquences d'interaction

A.7.6.3 Règles de filtrage

La primitive de service MD.addListener permet de définir de manière facultative les paramètres SourceURI et DestinationURI pour filtrer les télégrammes de MD reçus. Pour le filtrage, les règles suivantes doivent être appliquées:

- tout paramètre DestinationURI vide dans un télégramme de MD doit être reçu par un récepteur;
- un paramètre SourceURI vide dans un télégramme de MD n'a pas de signification particulière;
- une entrée de filtre DestinationURI vide dans la primitive de service MD.addListener signifie que le récepteur doit recevoir les télégrammes de MD, peu importe le paramètre DestinationURI défini en leur sein;
- une entrée de filtre SourceURI vide dans la primitive de service MD.addListener signifie que le récepteur doit recevoir les télégrammes de MD, peu importe le paramètre SourceURI défini en leur sein;
- deux récepteurs doivent disposer de paramètres de filtrage disjonctifs pour les paramètres ComId, SourceURI, DestinationURI, SourceIpAddress et DestinationIpAddress. En cas d'utilisation de paramètres de filtrage vides, les

deux récepteurs doivent différer en ce qui concerne au moins l'un de leurs paramètres de filtrage non vides.

- f) les télégrammes de MD avec un paramètre DestinationURI vide peuvent être reçus par plusieurs récepteurs avec différents filtres DestinationURI sur un seul appareil. Le protocole TRDP doit gérer la duplication des données reçues pour les différents récepteurs.

NOTE Un télégramme de MD envoyé à une adresse IP destinataire de monodiffusion avec un paramètre DestinationURI vide peut devenir une adresse de multidiffusion, c'est-à-dire que plusieurs réponses peuvent être reçues.

A.7.7 Contrôle du compteur de topographie

Avant l'envoi d'un télégramme, les compteurs de topographie du télégramme doivent être comparés aux compteurs de topographie réels afin de s'assurer que l'application émettrice partage la vision réelle du réseau central de train et la vision opérationnelle du train.

Après réception d'un télégramme, les valeurs du compteur de topographie doivent être vérifiées afin de s'assurer que l'appelant et le répondeur partagent une vision du réseau central de train et une vue opérationnelle du train identiques. Pour ce contrôle, les valeurs du compteur de topographie reçues sont d'abord comparées aux valeurs réelles du compteur de topographie puis à celles soumises dans la primitive de service MD.addListener.

Le contrôle du compteur de topographie est réussi si au moins un des cas répertoriés au est vrai.

Tableau A.21 – Contrôle du compteur de topographie

Appelant	Valeurs réelles du compteur de topographie		Valeurs du compteur de topographie du télégramme à envoyer	
Répondeur	Valeurs réelles du compteur de topographie		Valeurs du compteur de topographie du télégramme reçu	
	Valeurs du compteur de topographie du télégramme reçu		Valeurs stockées en local du compteur de topographie soumises avec l'ajout d'un récepteur	
Cas	etbTopoCnt	opTrnTopoCnt	etbTopoCnt	opTrnTopoCnt
1	tout	tout	0	0
2	0	égal à	0	égal à
3	égal à	tout	égal à	0
4	égal à	égal à	égal à	égal à
5(1)	0	0	égal à	égal à
(1) Le cas s'applique uniquement à un répondeur qui compare les valeurs du compteur de topographie du télégramme reçu aux valeurs stockées en local du compteur de topographie soumises avec l'ajout d'un écouteur. Utilisé pour la réception de télégrammes locaux. Légende: 0 = la valeur du compteur de topographie définie sur 0 (inutile) any = le compteur de topographie a une valeur comprise dans la plage $0..(2^{32}-1)$ equal = la valeur reçue et la valeur locale du compteur de topographie sont valides ($1..(2^{32}-1)$) et identiques				

A.7.8 Diagramme d'états de protocole de MD

A.7.8.1 Couche TRDP: appelant

Lorsqu'un utilisateur TRDP (appelant) souhaite envoyer un message, ses compteurs de topographie doivent être comparés aux compteurs de topographie réels. Si aucun des cas répertoriés dans le Tableau A.21 n'a la valeur TRUE, le message doit être ignoré.

Lorsqu'un utilisateur TRDP (appelant) souhaite envoyer une notification, un message de notification (MsgType 'Mn') doit être communiqué. L'identificateur de session du message de notification doit être défini sur 0.

Lorsqu'un utilisateur TRDP (appelant) souhaite envoyer un message de demande, une session d'appelant doit être ouverte et un message de demande (MsgType 'Mr') doit être communiqué.

Pour garantir l'unicité de l'identificateur de session, chaque session demande-réponse/demande-réponse-confirmation doit être identifiée par un UUID de 16 octets conformément à la RFC 4122, version en cours. L'identificateur de session est calculé du côté appelant dans la couche TRDP puis transmis via chaque message avant d'être utilisé des côtés appelant et répondeur afin d'identifier l'appelant correspondant et la session de réponse.

La valeur du délai d'expiration du ou des messages de réponse doit être définie par l'utilisateur TRDP (appelant).

NOTE Si des nouvelles tentatives sont sélectionnées, le délai d'expiration au niveau de la couche application peut être défini jusqu'à une valeur égale à $(\text{maxNumRetries} + 1) \times \text{replyTimeout}$.

Au cours de la session d'appelant, la couche TRDP doit attendre les messages de réponse entrants (MsgType 'Mp', 'Mq' ou 'Me'). Tous les messages de réponse entrants relatifs à la session d'appelant (identifiés par l'identificateur de session reçu) doivent aussitôt être communiqués à l'utilisateur TRDP.

Lorsque le nombre planifié de réponses entrantes est atteint et qu'il ne reste aucune confirmation en attente de la part de l'utilisateur TRDP (appelant), la session d'appelant doit être clôturée.

Lorsque le demandeur souhaite obtenir une confirmation (MsgType "Mq"), la couche TRDP doit enclencher le minuteur selon le délai d'expiration de confirmation de la réponse spécifié et attendre la confirmation de l'utilisateur TRDP (appelant).

La confirmation doit utiliser l'URI source reçu dans le message de réponse en tant qu'URI destinataire.

Après avoir obtenu la confirmation de l'utilisateur TRDP (appelant), la couche TRDP doit envoyer un message de confirmation (MsgType 'Mc') au répondeur.

Le message de confirmation ne doit pas contenir de données utilisateur et doit uniquement être envoyé en monodiffusion.

L'utilisateur TRDP (appelant) doit veiller à envoyer la confirmation selon le délai défini par le paramètre ReplyTimeOut de la réponse.

Lorsque le nombre de réponses entrantes est atteint et que le minuteur du délai de confirmation expire alors que certaines confirmations de l'utilisateur TRDP (appelant) sont toujours en attente, ce dernier doit en être informé et la session d'appelant doit être clôturée.

Lorsque le minuteur du délai de réponse de la session d'appelant expire en raison de l'absence de réponse alors que le nombre de répondeurs prévu est de 1, la couche TRDP doit, selon la valeur de paramètre 'MaxNumRetries' donnée, répéter la demande jusqu'à deux fois avant d'informer l'utilisateur TRDP (appelant) de l'absence de réponse et de clôturer la session d'appelant.

Lorsque le minuteur du délai de réponse expire et que le nombre de répondeurs prévu est supérieur à 1 alors que certaines réponses sont toujours en attente, la couche TRDP doit informer l'utilisateur TRDP (appelant) de l'expiration du délai et du nombre de réponses manquantes. La session d'appelant doit être clôturée lorsque toutes les confirmations demandées ont été envoyées ou que le minuteur du délai correspondant a expiré.

Lorsque le nombre de répondeurs est inconnu (paramètre NoOfRepliers = 0) et que le minuteur du délai de réponse expire, le nombre total de réponses doit être communiqué à l'utilisateur TRDP (appelant) et la session d'appelant doit être clôturée.

En cas d'utilisation du protocole TCP, aucune retransmission des messages de demande ne doit être effectuée. En cas de retransmission TCP ou de poursuite de la transmission, un service de niveau supérieur ou l'application doivent être utilisés. Le protocole TRDP doit permettre à l'interface de poursuivre la transmission après que la connexion ait été perdue.

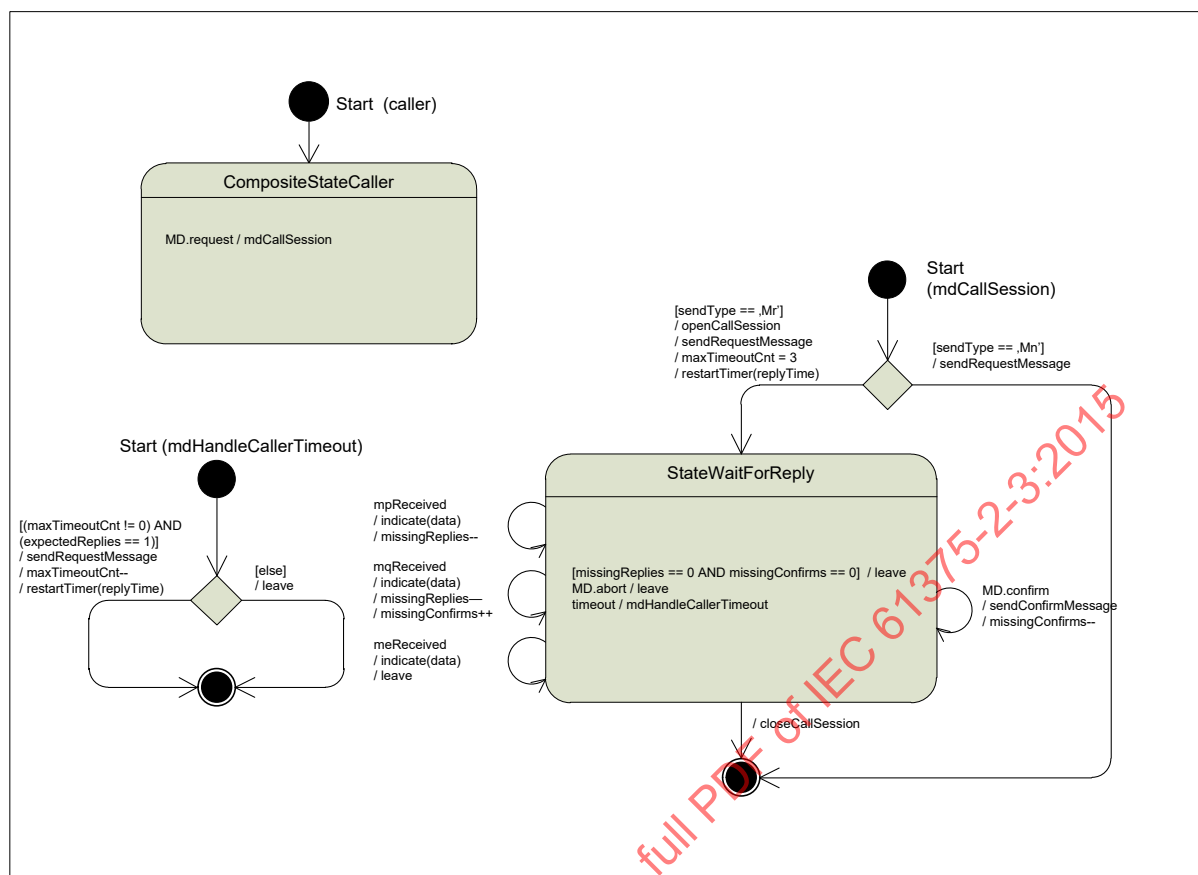
Lorsque la couche TRDP reçoit un message de réponse (MsgType 'Mp', 'Mq' ou 'Me') sans avoir ouvert de session d'appelant pour l'identificateur en question, le message doit être supprimé.

Les valeurs du compteur de topographie obtenues pendant l'ouverture de la session d'appelant doivent être utilisées dans l'ensemble de la session (pour les messages de demande et de confirmation).

Lorsqu'un utilisateur TRDP reçoit un message de réponse avec des valeurs du compteur de topographie différentes des valeurs prévues (voir Tableau A.5), le message doit être supprimé.

Si un utilisateur TRDP abandonne une session (ouverte), par exemple après une modification de la topologie du train, les minuteurs en fonctionnement doivent être interrompus, l'identificateur de session doit être détruit et la session doit être clôturée.

Le comportement de la couche TRDP des répondeurs est décrit à la Figure A.21 et des explications sont apportées du Tableau A.22 au Tableau A.25.



IEC

Anglais	Français
Start (caller)	Lancement (appelant)

Figure A.21 – Diagramme d'états de l'appelant de MD de la couche TRDP

Tableau A.22 – Diagramme d'états de l'appelant de MD: triggers

Triggers	Description
MD.request	L'utilisateur a appelé la primitive de service MD.request.
mpReceived	Réception d'une réponse valide sans confirmation demandée, voir A.7.8.3.
mqReceived	Réception d'une réponse valide avec confirmation demandée, voir A.7.8.3.
meReceived	Réception d'un message d'erreur de MD valide, voir A.7.8.3.
MD.confirm	L'utilisateur a appelé la primitive de service MD.confirm.
MD.abort	L'utilisateur a appelé la primitive de service MD.abort pour clôturer la session ouverte (par exemple après une modification des compteurs de topographie).
délai expiré	Expiration du délai de réponse (aucun message de réponse reçu dans le délai défini par le paramètre replyTimeout)

Tableau A.23 – Diagramme d'états de l'appelant de MD: guards

Guards	Description
sendType	Type de télégramme de demande de MD demandé par l'utilisateur dans la primitive de service MD.request.
expectedReplies	Nombre de réponses prévues indiquées par la primitive de service MD.request (paramètre NoOfRepliers').
missingReplies	Compteur de réponses manquantes.

Guards	Description
missingConfirms	Compteur de confirmations manquantes.
maxTimeoutCnt	Compteur de délais expirés. Sa valeur doit être diminuée au fur et à mesure que les délais expirent.
checkMessage	Vérifie si le CRC, la version de protocole, le type de protocole, la longueur et le compteur de topographie du message reçu sont corrects (au moins un des cas répertoriés dans le Tableau A.21 pour les compteurs de topographie est vrai).
checkTopoCounts	Compare les compteurs de topologie du message reçu aux compteurs de topographie réels. Au moins un des cas répertoriés dans le Tableau A.21 pour les compteurs de topographie est satisfait.

Tableau A.24 – Diagramme d'états de l'appelant de MD: actions

Actions	Description
mdCallSession	Gère les états d'une session d'appelant.
openCallSession	Crée une session d'appelant.
closeCallSession	Clôture la session d'appelant.
mdGetCallSessionState	Renvoie l'état de la session d'appelant pour un paramètre sessionId donné.
sendRequestMessage	Envoie le message de demande. Si l'envoi du message de demande est répété après le délai d'expiration du paramètre du télégramme, le paramètre 'SequenceCounter' doit être incrémenté.
sendConfirmMessage	Envoie le message de confirmation.
Indicate	Informe l'utilisateur sur un message de réponse ou d'erreur reçu en invoquant la primitive de service MD.indicate.
restartTimer(replyTime)	Démarre un minuteur à partir de la valeur replyTimeout définie par l'utilisateur (primitive de service MD.request).
discard	Ignore le message reçu sans autre traitement.

Tableau A.25 – Diagramme d'états de l'appelant de MD: états

Etat	Description
CompositeStateCaller	Etat composite de l'appelant.
StateWaitForReply	Etat d'une session d'appelant ayant envoyé une demande et étant actuellement en attente de la ou des réponses et des éventuelles confirmations demandées par l'application.

A.7.8.2 Couche TRDP: répondeur (récepteur)

Chaque utilisateur TRDP (récepteur) qui souhaite recevoir des MD doit s'enregistrer en tant que récepteur de celles envoyées à l'URI en question (en mono- ou multidiffusion) ou de celles d'un paramètre ComId spécifique.

Tous les messages entrants doivent être contrôlés au regard des compteurs de topographie réels. Les messages qui ne correspondent pas aux compteurs de topographie réels (voir Tableau A.21) doivent être supprimés.

Tous les messages entrants doivent être contrôlés au regard des récepteurs enregistrés. Tout message à un récepteur non enregistré ou à un récepteur ayant prévu une autre valeur des compteurs de topographie doit être supprimé.

En cas de message de demande monodiffusé à un récepteur non enregistré ou à un récepteur ayant prévu une autre valeur relative aux compteurs de topographie, un message d'erreur (MsgType = 'Me') doit être envoyé en indiquant l'erreur avec replyStatus == -3 (aucune instance de répondeur).

Lorsqu'un message de notification (MsgType 'Mn') est reçu, celui-ci doit être transféré au récepteur concerné.

Lorsqu'un message de demande (MsgType 'Mr') est reçu, une session de réponse doit être ouverte à partir de l'identificateur de session reçu, le minuteur du délai d'expiration doit être déclenché selon le délai de réponse spécifié, ledit message doit être transféré au récepteur concerné, et la couche TRDP doit attendre la primitive MD.reply du récepteur.

Si, en raison d'un manque de ressources, une session de réponse ne peut pas être ouverte, le message reçu doit être supprimé et un message d'erreur (MsgType = 'Me') doit être envoyé en indiquant l'erreur avec replyStatus == -4 (aucune mémoire (côté répondeur)).

Lorsqu'un message de demande (MsgType 'Mr') est reçu pour une session de réponse ouverte avec l'identificateur correspondant alors que la primitive MD.reply du récepteur est en attente de réception, ce message de demande doit être supprimé.

Après avoir reçu la primitive MD.reply de l'utilisateur TRDP (récepteur), la couche TRDP doit envoyer à l'appelant (en fonction de la primitive MD.reply du récepteur) un message de réponse avec (MsgType 'Mq') ou sans (MsgType 'Mp') confirmation.

Un message de réponse doit être envoyé uniquement en monodiffusion en utilisant l'adresse IP et l'URI source du message de demande associé en tant qu'adresse IP destinataire et en tant qu'URI destinataire.

La primitive MD.reply d'un récepteur doit utiliser comme adresse IP source l'adresse IP du récepteur et comme URI source l'URI unique (section utilisateur) du récepteur.

Lorsqu'un message de réponse sans confirmation (MsgType 'Mp') a été envoyé, la session de réponse doit être clôturée.

Lorsqu'un message de réponse avec confirmation (MsgType 'Mq') a été envoyé, le minuteur du délai d'expiration de la session de réponse doit être redémarré selon le délai de confirmation spécifié.

Si un message de demande (MsgType 'Mr') est reçu avec un compteur de séquences différent pour une session de réponse déjà ouverte à la suite de la réception par la couche TRDP de la primitive MD.reply de l'utilisateur TRDP (répondeur), la couche TRDP doit répéter l'envoi du message de réponse (MsgType = 'Mq') à l'appelant avec incrémentation du compteur de séquences. Après renvoi du message de réponse, le minuteur du délai d'expiration de la session de réponse doit être redémarré selon le délai de confirmation spécifié.

Lorsqu'un message de demande (MsgType 'Mr') est reçu pour une session de réponse déjà ouverte et le même compteur de séquences que celui reçu précédemment, ce message de demande doit être supprimé.

Lorsqu'un message de confirmation (MsgType 'Mc') est reçu pour une session de réponse ouverte avec l'identificateur de session reçue, le récepteur correspondant doit en être informé et la session doit être clôturée.

Lorsqu'un message de confirmation (MsgType 'Mc') est reçu sans session de réponse ouverte avec l'identificateur correspondant, ce message doit être supprimé.

Lorsque le minuteur du délai d'une session de réponse expire, celle-ci doit être clôturée. En cas d'expiration du délai lié à l'absence de confirmation ou de primitive MD.reply du récepteur, ce dernier doit en être informé.

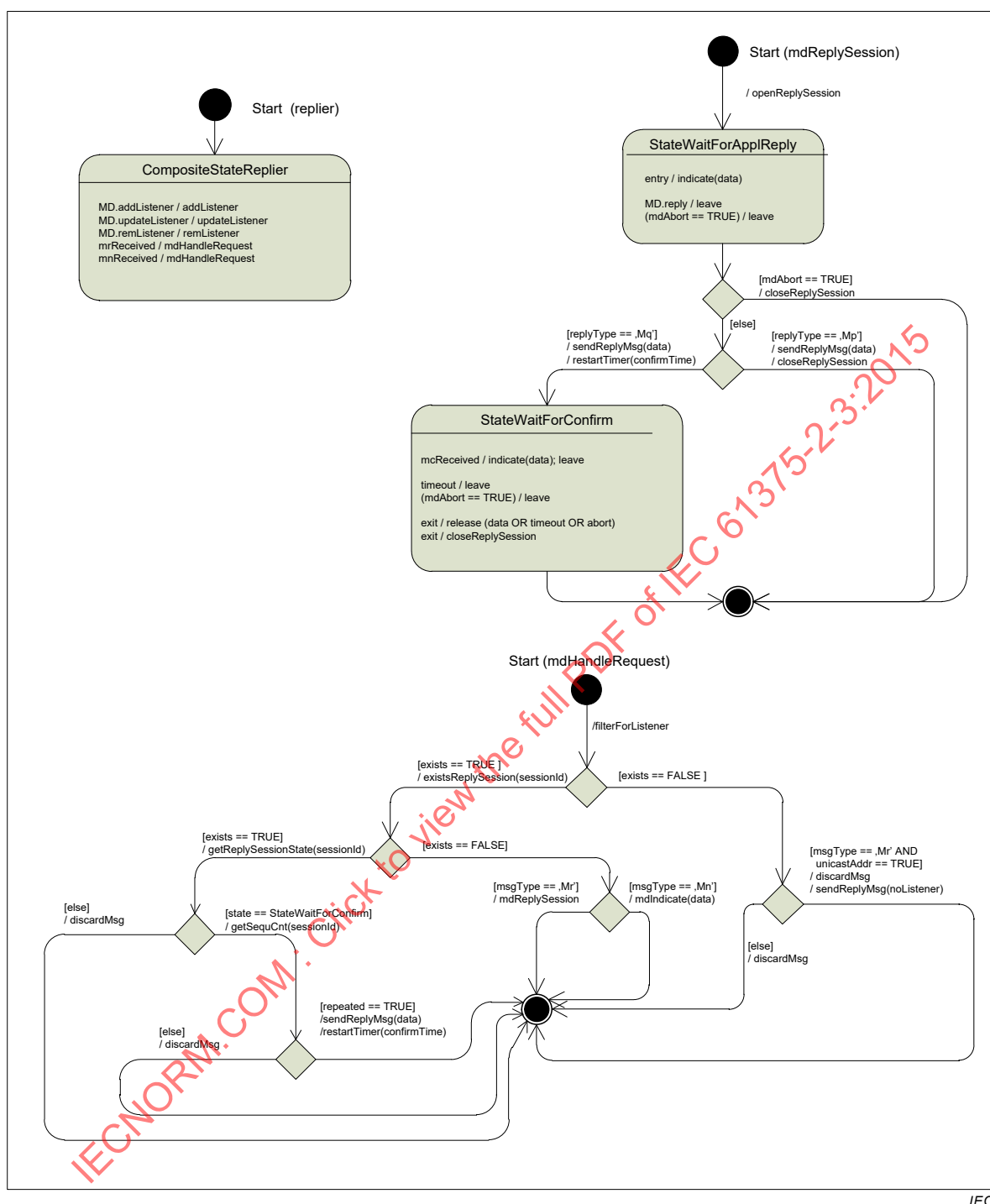
Les valeurs du compteur de topographie obtenues dans une primitive de service MD.addListener doivent être utilisées pour les futures sessions de réponses, sauf si elles sont mises à jour par l'utilisateur (primitive de service MD.updateListener).

Lorsqu'un utilisateur TRDP reçoit un message de demande avec des valeurs du compteur de topographie différentes des valeurs prévues (voir Tableau A.21), le message doit être supprimé.

Au cours d'une session ouverte, si un utilisateur TRDP supprime ou met à jour un récepteur (par exemple après une modification de la topologie du train), les minuteurs en fonctionnement doivent être interrompus, l'identificateur de session doit être détruit et la session doit être clôturée.

Le comportement de la couche TRDP des répondeurs est décrit à la Figure A.22 et des explications sont apportées du Tableau A.26 au Tableau A.29.

IECNORM.COM : Click to view the full PDF of IEC 61375-2-3:2015



IEC

Anglais	Français
Start (replier)	Lancement (répondeur)

Figure A.22 – Diagramme d'états du répondeur de MD de la couche TRDP

Tableau A.26 – Diagramme d'états du répondeur de MD: triggers

Triggers	Description
MD.addListener	L'utilisateur a appelé la primitive de service MD.addListener.
MD.updateListener	L'utilisateur a appelé la primitive de service MD.updateListener.
MD.removeListener	L'utilisateur a appelé la primitive de service MD.removeListener.
mrReceived	Réception d'une demande valide, voir A.7.8.3.
mnReceived	Notification d'une demande valide, voir A.7.8.3.
mcReceived	Réception d'une confirmation valide, voir A.7.8.3.
MD.Reply	L'utilisateur a appelé la primitive de service MD.reply.
délai expiré	Expiration du délai de confirmation (aucun message de confirmation reçu dans le délai défini par le paramètre replyTimeout).

Tableau A.27 – Diagramme d'états du répondeur de MD: guards

Guards	Description
existsReplySession (sessionId)	Il existe une session de réponse pour un identificateur de session.
existsListener	Il existe un récepteur pour la demande donnée qui correspond aux compteurs de topographie (voir Tableau A.21).
replyType	Type de télégramme de réponse de MD demandé par l'utilisateur dans la primitive de service MD.reply.
stateReplySession(sessionId)	État de la session du répondeur pour un identificateur de session donné
repeated	Indication de répétition du message de demande.
unicastAddr	L'adresse destinataire est une adresse de monodiffusion.
checkMessage	Vérifie si le CRC, la version de protocole, le type de protocole, la longueur et le compteur de topographie du message reçu sont corrects (au moins un des cas répertoriés dans le Tableau A.21 pour les compteurs de topographie est vrai).
checkTopoCounts	Compare les compteurs de topographie du message reçu aux compteurs de topographie réels (au moins un des cas répertoriés dans le Tableau A.21 pour les compteurs de topographie est vrai).

Tableau A.28 – Diagramme d'états du répondeur de MD: actions

Actions	Description
mdReplySession	Gère les états d'une session de réponse.
mdHandleRequest	Gère le message de demande reçu.
openReplySession	Crée une session de réponse.
closeReplySession	Clôture la session de réponse.
sendReplyMessage	Envoie le message de réponse.
indicate	Informe l'utilisateur sur un message de demande ou de confirmation reçu en invoquant la primitive de service MD.indicate.
release	Informe l'utilisateur sur un message de confirmation ou un délai d'expiration reçu en invoquant la primitive de service MD.release.
addListener	Ajoute un récepteur pour des critères de filtrage donnés. mdAbort = FALSE
updateListener	Met à jour les paramètres du récepteur. mdAbort = TRUE
removeListener	Supprime un récepteur.

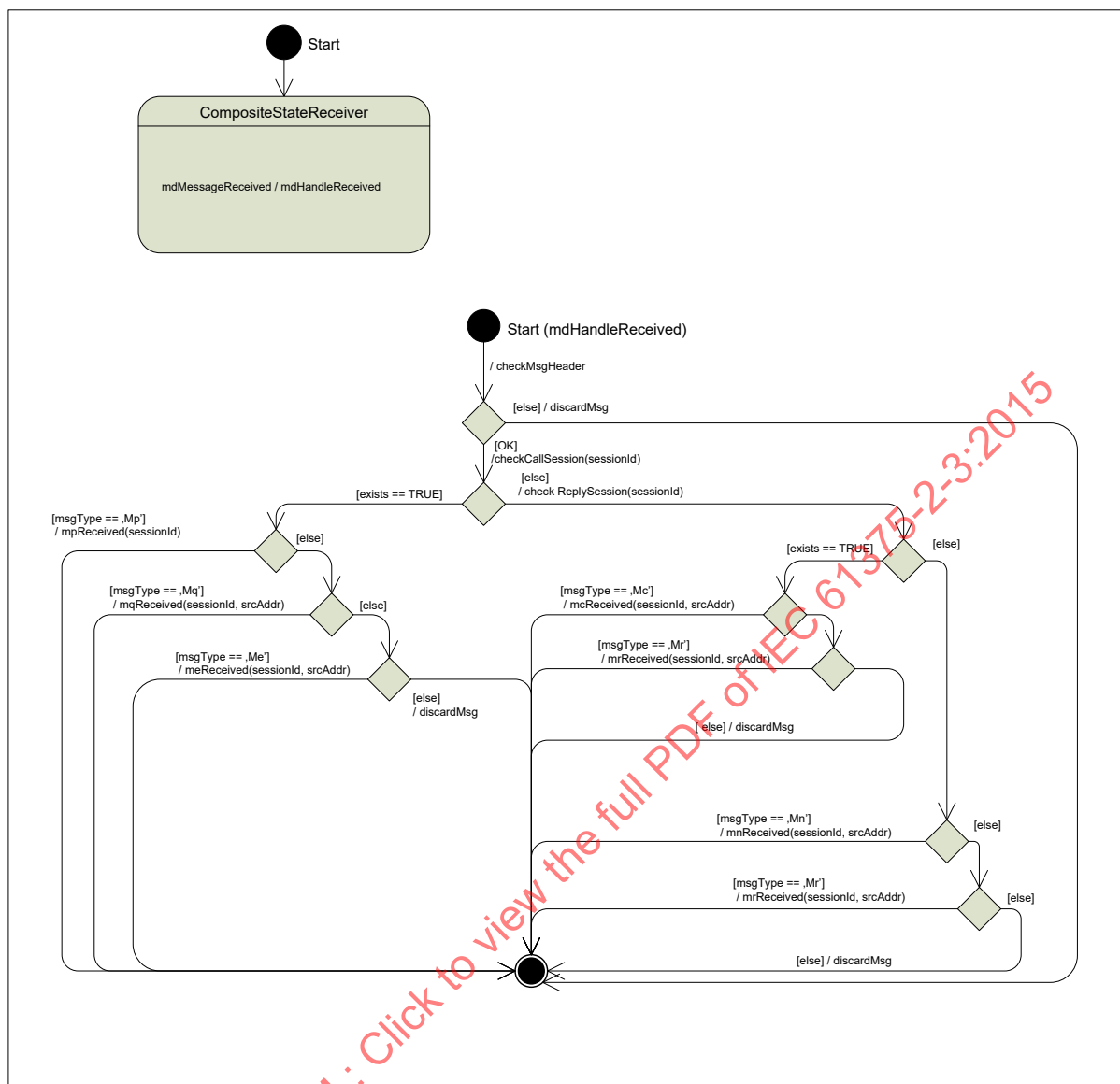
Actions	Description
	mdAbort = TRUE
getSequCnt(sessionId)	Contrôle si le message de demande a été répété par l'appelant (à savoir si le compteur de séquences du message de demande reçu auparavant est différent de la valeur réelle reçue). repeated == TRUE: compteurs de séquences différents repeated == FALSE: compteurs de séquences identiques
discardMsg	Supprime le message reçu.
restartTimer(confirmTime)	Démarre un minuteur à partir de la valeur replyTimeout définie par l'utilisateur (primitive de service MD.reply).
sendReplyMsg(param)	Envoie un message de réponse. Param = 'data': – Type de réponse: 'Mq' OU 'Mp' (défini par l'utilisateur) – Etat de réponse: défini par l'utilisateur – ComId de réponse: défini par l'utilisateur – Dataset: défini par l'utilisateur
sendReplyErr(replyStatus)	Envoie le message de réponse d'erreur Param = 'noListener' – Type de réponse: 'Me' – Etat de réponse: replyStatus – ComId de réponse: 0 – Dataset: vide

Tableau A.29 – Diagramme d'états du répondeur de MD: états

Etat	Description
CompositeStateReplier	Etat composite du répondeur.
StateWaitForApplReply	Etat d'une session de réponse ayant envoyé un message de demande reçu portant sur l'envoi d'une réponse à l'application et étant actuellement en attente de la primitive MD.reply de l'application.
StateWaitForConfirm	Etat d'une session de réponse ayant envoyé une réponse portant sur l'envoi d'une confirmation et étant actuellement en attente de cette dernière.

A.7.8.3 Gestion des messages reçus (action mdHandleReceived)

Le diagramme d'états présenté à la Figure A.23 et dont les explications sont apportées du Tableau A.30 au Tableau A.33 décrit la gestion des télégrammes reçus relatifs aux données de messagerie.



IEC

Anglais	Français
Start	Lancement

Figure A.23 – Réception du télégramme de MD de la couche TRDP

Tableau A.30 – Diagramme d'états du destinataire de MD: triggers

Triggers	Description
mdMessageReceived	Un télégramme de MD a été reçu.

Tableau A.31 – Diagramme d'états du destinataire de MD: guards

Guards	Description
msgType	Type de télégramme du message tel que défini le Tableau A.18

Tableau A.32 – Diagramme d'états du destinataire de MD: actions

Actions	Description
checkCallSession	Contrôle si une session d'appelant existe pour la réponse donnée. Si c'est le cas, il contrôle si au moins un des cas énumérés dans le Tableau A.5 pour les compteurs de topographie est vrai. exists = TRUE: les deux contrôles ont été validés
checkReplySession	Contrôle si une session de réponse existe pour la demande ou la confirmation donnée exists = TRUE: contrôle positif NOTE Plusieurs sessions de réponse peuvent exister.
checkMsgHeader	Contrôle l'en-tête de télégramme de MD pour la FCS, la version et le compteur de séquences puis le supprime s'il est incorrect.
mpReceived	Déclenche la réception du télégramme de MD de type 'Mp'.
mqReceived	Déclenche la réception du télégramme de MD de type 'Mq'.
mcReceived	Déclenche la réception du télégramme de MD de type 'Mc'.
mrReceived	Déclenche la réception du télégramme de MD de type 'Mr'.
mnReceived	Déclenche la réception du télégramme de MD de type 'Mn'.
discardMsg	Supprime le message.

Tableau A.33 – Diagramme d'états du destinataire de MD: état

Etat	Description
CompositeStateReceiver	Etat composite du destinataire.

A.7.9 Gestion des connexions TCP

Ce paragraphe définit la gestion des connexions TCP pour la transmission de données de messagerie TRDP.

L'utilisation du protocole TCP pour la transmission de données exige l'utilisation d'une connexion TCP établie entre l'appelant et l'appareil du répondeur. Ainsi, la première demande à un appareil spécifique doit impliquer l'ouverture d'une connexion lorsqu'aucune n'a encore été établie auparavant.

Tant qu'une connexion TCP vers un autre appareil est ouverte dans la direction exigée, il convient de la réutiliser pour les appels suivants réalisés vers celui-ci.

L'appelant doit clôturer une connexion TCP existante (extrémité active) dans les cas suivants:

- un signal indiquant que la connexion TCP sera clôturée a été reçu;
- le protocole TRDP est éteint ou réinitialisé;
- un délai a expiré, la connexion TCP n'ayant pas été utilisée pendant un certain temps.

Le répondeur doit utiliser la connexion ouverte par l'appelant.

L'appelant doit clôturer une connexion TCP existante (extrémité passive) dans les cas suivants:

- un signal indiquant que la connexion TCP sera clôturée a été reçu;
- le protocole TRDP est éteint ou réinitialisé;
- Une autre connexion TCP a été ouverte pour le même appareil de l'appelant et l'ancienne n'a pas été utilisée depuis un certain temps.

A.8 Serveur de données de messagerie d'écho (option)

Les appareils terminaux qui prennent en charge les données de messagerie TRDP peuvent fournir une fonction d'écho de données de messagerie. Un système qui souhaite soumettre à l'essai la communication avec un appareil terminal envoie une demande d'écho (ERQ) à l'appareil terminal en utilisant la primitive de service MD.request. L'appareil terminal active une fonction d'écho qui crée une réponse d'écho (ERP), copie les données de l'ERQ vers l'ERP puis les renvoie à l'émetteur.

Il convient de mettre en œuvre les données de messagerie d'écho dans la couche TRDP.

Lorsque les données de messagerie arrivent avec le paramètre ComId d'écho (voir Tableau A.2), un message d'écho est créé et renvoyé à la source. Les données utilisateur du message renvoyé doivent correspondre à la copie des données utilisateur reçues.

IECNORM.COM : Click to view the full PDF of IEC 61375-2-3:2015

Annexe B (normative)

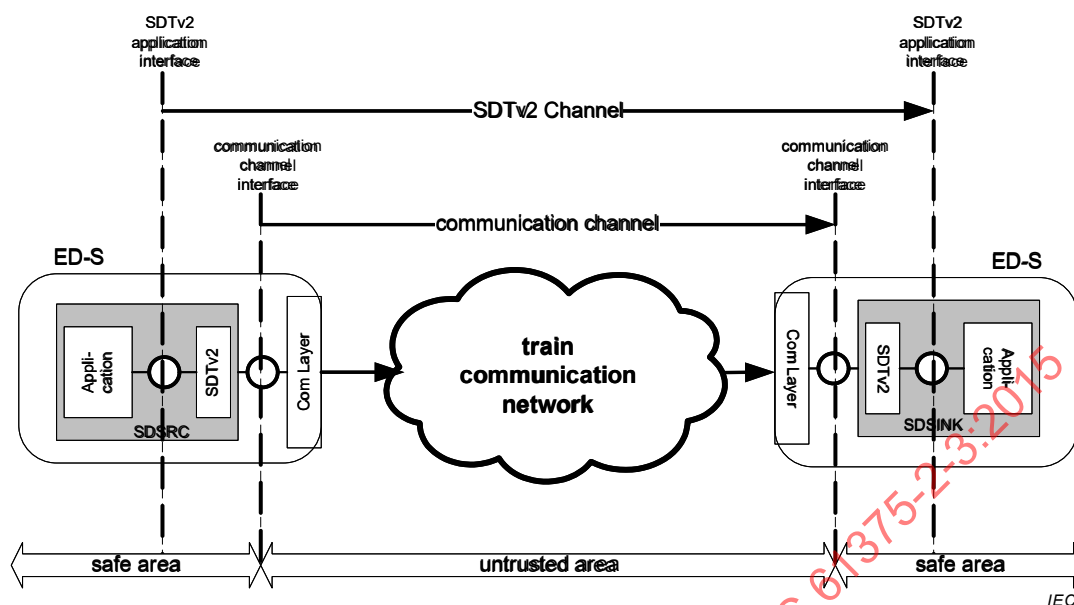
Transmission de données sécurisées (SDTv2)

B.1 Généralités

La présente annexe définit l'extension du profil de communication pour la transmission de données relatives à la sécurité (stratégiques) sur l'ETB.

B.2 Vue d'ensemble de la SDTv2 (informelle)

La SDTv2 offre un chemin de communication sécurisé entre la source de données relatives à la sécurité (stratégiques), ou SDSRC, et le(s) destinataire(s) concerné(s), ou SDSINK. Ce chemin de communication sécurisé est appelé "canal SDTv2". La Figure B.1 représente un modèle logique de canal SDTv2. Le canal SDTv2 commence et se termine sur des applications sécurisées et couvre l'ensemble du chemin, de l'application SDSRC à une ou plusieurs applications SDSINK. Il s'agit notamment du réseau embarqué de train (TCN) et des couches de communication relatives qui se trouvent sur les appareils terminaux sécurisés (ED-S) hébergeant l'application sécurisée, lesquels sont considérés comme non sûrs en termes de sécurité. Un appareil terminal sécurisé peut contenir un ou plusieurs éléments SDSRC ou SDSINK. Ces éléments peuvent être décrits comme une combinaison de l'application sécurisée et de la couche de protocole SDTv2. La couche de protocole SDTv2 offre deux interfaces: celle du canal de communication et celle de l'application SDTv2. L'interface du canal de communication est définie par la technologie de communication sous-jacente (TRDP). Il s'agit de l'interface sur laquelle les unités de données de protocole SDTv2, à savoir les paquets de données stratégiques (VDP), sont échangées avec le TCN. Spécifique au produit, l'interface de l'application SDTv2 constitue l'endroit où les données de processus relatives à la sécurité sont échangées avec l'application. La couche SDTv2 du côté de l'élément SDSRC est principalement destinée à ajouter des informations sur le protocole en vue de créer un VDP, lequel est nécessaire au transfert sécurisé de ces données sur le réseau avant leur envoi. Du côté destinataire, la couche SDTv2 valide les VDP reçus. Lorsque la validation est probante, les données stratégiques correspondantes sont présentées dans l'interface de l'application SDTv2.



Anglais	Français
SDTv2 application interface	Interface d'application SDTv2
SDTv2 channel	Canal SDTv2
communication channel interface	interface du canal de communication
communication channel	canal de communication
train communication network	réseau de communication de train
Com Layer	Couche communication
safe area	zone sécurisée
untrusted area	zone non sécurisée

Figure B.1 – Canal SDTv2

Les VDP sont configurés pour une transmission cyclique (par exemple, ils sont transportés avec la charge utile des données de processus TRDP).

B.3 Exigences fonctionnelles de sécurité

Les exigences ci-dessous ont constitué un élément obligatoire pour l'élaboration du protocole SDTv2:

- Les communications de sécurité doivent être indépendantes des communications normalisées (ordinaires). En revanche, les appareils normalisés et les appareils de sécurité doivent pouvoir utiliser le même canal de communication.
- En mode de fonctionnement continu, les communications de sécurité doivent être conformes au niveau d'intégrité de sécurité SIL2 (voir l'IEC 61508-1).
- Elles doivent utiliser un système de communication monocanal. La redondance ne peut être utilisée qu'occasionnellement, en vue d'augmenter la disponibilité.
- La mise en œuvre du protocole de transmission sécurisé doit être limitée aux appareils de communication terminaux.
- Au regard des propriétés dynamiques des compositions de train dont le nombre de rames varie, une relation de communication 1:n entre la source et le destinataire de données sécurisées doit être prise en charge.
- La durée des transmissions doit être surveillée.

- g) En l'absence de norme de produit spécifique, les conditions d'environnement doivent être conformes aux exigences générales applicables aux trains, et notamment à l'IEC 60571.
- h) Les équipements de transmissions à l'instar des contrôleurs, circuits intégrés spécifiques (ASIC, Application Specific Integrated Circuit), tores de commutation Ethernet, câbles, coupleurs, etc. ne doivent pas être modifiés (canal noir). Les fonctions de sécurité doivent être supérieures à la couche OSI 7 (profil sans modification ni amélioration du protocole normalisé).
- i) Les communications de sécurité ne doivent pas réduire le nombre d'appareils admis.
- j) Elles doivent prendre en charge l'échange de données sur l'ETB pour les compositions de train qui changent de manière dynamique.

B.4 Mesures de sécurité

Les mesures de sécurité énumérées dans le Tableau B.1 en matière de gestion des erreurs potentielles de transmission constituent un composant significatif de la SDTv2.

Les mesures de sécurité doivent être traitées et surveillées par un même appareil de sécurité.

Tableau B.1 – Mesures adoptées face aux erreurs de communication

Erreur de communication	Mesures de sécurité						
	Compteur de séquences sécurisées	Supervision du temps de transmission	Code de sécurité	Intervalle de garde	Identificateur de la source (SID)	Surveillance de la latence	Surveillance du canal
Corruption			X				
Répétition intempestive	X						
Séquence erronée	X						
Perte		X					
Délai trop important		X				X	
Insertion	X				X		
Usurpation	X				X		
Adressage					X		
Défaillance de la redondance (sources redondantes actives)				X			
Taux d'erreur trop important sur le canal de communication		X	X				X

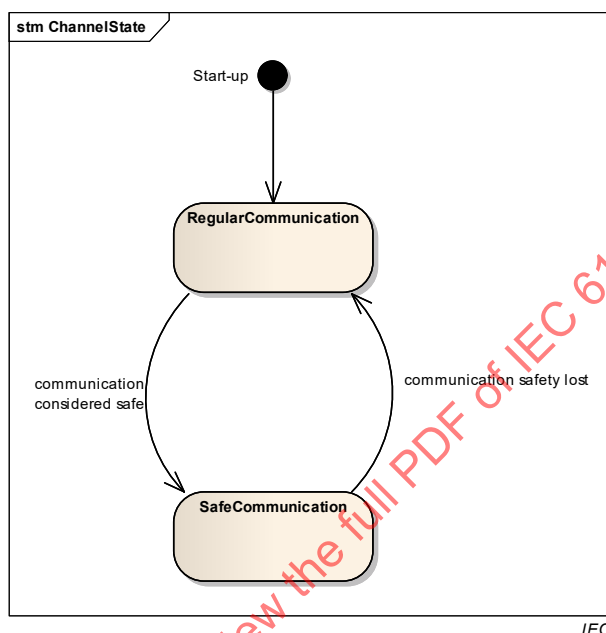
B.5 Modes opérationnels du canal SDTv2

Un canal SDTv2 peut fonctionner selon deux modes opérationnels de base (voir Figure B.2):

le mode RegularCommunication: les données de processus stratégiques transmises ne peuvent pas être considérées comme sécurisées;

le mode SafeCommunication: les données de processus stratégiques transmises peuvent être considérées comme sécurisées.

Les conditions de modification relatives à l'un ou l'autre de ces modes sont décrites dans les paragraphes suivants.



Anglais	Français
Start-up	Démarrage
Regular Communication	Communication normale
communication considered safe	communication considérée comme sécurisée
communication safety lost	perte de sécurité de la communication
Safe Communication	Communication sécurisée

Figure B.2 – Etats du canal SDTv2

B.6 Présentation des données

Toutes les données de la SDTv2 doivent être transmises au format big-endian (l'octet de poids fort d'un élément de données est prioritaire).

Toutes les structures de données utilisées avec la SDTv2 doivent être naturellement harmonisées (éléments de données enregistrés au niveau de l'adresse de décalage, laquelle correspond à un multiple de leur dimension).

Les structures de données de longueur variable (réseaux à circuit ouvert, enregistrements) ne doivent pas être utilisées.

Les éléments de données structurés des VDP (enregistrement, séquence) doivent être placés dans l'ordre dans lequel ils sont présentés.

B.7 SC-32

Le SC-32 désigne un code de redondance cyclique utilisé pour le calcul du SID et du paramètre SafetyCode. Le polynôme qui doit être utilisé est '1F4ACFB13', comme le définit l'IEC 61784-3-3. Il convient d'utiliser l'algorithme de la notation du langage de programmation C décrit à la Figure B.3 (et défini dans l'IEC 61784-3-3) pour le calcul du SC-32 (voir également Figure B.4).

```

/*****
 * GLOBAL FUNCTION DEFINITIONS
 */

/**
 * @internal
 * Calculates and returns a 32-bit CRC.
 *
 * @param buf    Input buffer
 * @param len    Length of input buffer
 * @param crc    Initial (seed) value for the CRC calculation
 *
 * @return Calculated CRC value
 */
static UNSIGNED32 sdt_crc32(const UNSIGNED8 buf[], UNSIGNED32 len, UNSIGNED32 crc)
{
    UNSIGNED32 i;

    for (i = 0; i < len; i++)
    {
        crc = crctab32[((UNSIGNED32)(crc >> 24) ^ buf[i]) & 0xff] ^ (crc << 8);
    }

    return crc;
}

```

Figure B.3 – Calcul du SC-32

```

/*
 * Table for 32-bit CRC calculation according to IEC 61784-3-3 (2007)
 * CRC of the string "123456789" is 0x1697d06a
 */
static const UNSIGNED32 crcTab32[256]={
    0x00000000U, 0xF4ACFB13U, 0x1DF50D35U, 0xE959F626U,
    0x3BEA1A6AU, 0xCF46E179U, 0x261F175FU, 0xD2B3EC4CU,
    0x77D434D4U, 0x8378CFC7U, 0x6A2139E1U, 0x9E8DC2F2U,
    0x4C3E2EBEU, 0xB892D5ADU, 0x51CB238BU, 0xA567D898U,
    0xEFA869A8U, 0x1B0492BBU, 0xF25D649DU, 0x06F19F8EU,
    0xD44273C2U, 0x20EE88D1U, 0xC9B77EF7U, 0x3D1B85E4U,
    0x987C5D7CU, 0x6CD0A66FU, 0x85895049U, 0x7125AB5AU,
    0xA3964716U, 0x573ABC05U, 0xBE634A23U, 0x4ACFB130U,
    0x2BFC2843U, 0xDF50D350U, 0x36092576U, 0xC2A5DE65U,
    0x10163229U, 0xE4BAC93AU, 0x0DE33F1CU, 0xF94FC40FU,
    0x5C280C97U, 0xA884E784U, 0x41DD11A2U, 0xB571EAB1U,
    0x67C206FDU, 0x936EFDEEU, 0x7A370BC8U, 0x8E9BF0DBU,
    0xC45441EBU, 0x30F8BAF8U, 0xD9A14CDEU, 0x2D0DB7CDU,
    0xFFBE5B81U, 0x0B12A092U, 0xE24B56B4U, 0x16E7ADA7U,
    0xB380753FU, 0x472C8E2CU, 0xAE75780AU, 0x5AD98319U,
    0x886A6F55U, 0x7CC69446U, 0x959F6260U, 0x61339973U,
    0x57F85086U, 0xA354AB95U, 0x4A0D5DB3U, 0xBEA1A6A0U,
    0x6C124AECU, 0x98BEB1FFU, 0x71E747D9U, 0x854BBCCAU,
    0x202C0452U, 0xD4809F41U, 0x3DD96967U, 0xC9759274U,
    0x1BC67E38U, 0xEF6A852BU, 0x0633730DU, 0xF29F881EU,
    0xB850392EU, 0x4CFC23DU, 0xA5A5341BU, 0x5109CF08U,
    0x83BA2344U, 0x7716D857U, 0x9E4F2E71U, 0x6AE3D562U,
    0xCF840DFAU, 0x3B28F6E9U, 0xD27100CFU, 0x26DDFBDU,
    0xF46E1790U, 0x00C2EC83U, 0xE99B1AA5U, 0x1D37E1B6U,
    0x7C0478C5U, 0x88A883D6U, 0x61F175F0U, 0x955D8EE3U,
    0x47EE62AFU, 0xB34299BCU, 0x5A1B6F9AU, 0xAEB79489U,
    0x0BD04C11U, 0xFF7CB702U, 0x16254124U, 0xE289BA37U,
    0x303A567BU, 0xC496AD68U, 0x2DCF5B4EU, 0xD963A05DU,
    0x93AC116DU, 0x6700EA7EU, 0x8E591C58U, 0x7AF5E74BU,
    0xA8460B07U, 0x5CEAF014U, 0xB5B30632U, 0x411FFD21U,
    0xE47825B9U, 0x10D4DEAAU, 0xF98D288CU, 0x0D21D39FU,
    0xDF923FD3U, 0x2B3EC4C0U, 0xC26732E6U, 0x36CBC9F5U,
    0xAFF0A10CU, 0x5B5C5A1FU, 0xB205AC39U, 0x46A9572AU,
    0x941ABB66U, 0x60B64075U, 0x89EFB653U, 0x7D434D40U,
    0xD82495DBU, 0x2C886ECBU, 0xC5D198EDU, 0x317D63FEU,
    0xE3CE8FB2U, 0x176274A1U, 0xFE3B8287U, 0x0A977994U,
    0x4058C8A4U, 0xB4F433B7U, 0x5DADC591U, 0xA9013E82U,
    0x7BB2D2CEU, 0x8F1E29DDU, 0x6647DFFBU, 0x92EB24E8U,
    0x378CFC70U, 0xC3200763U, 0x2A79F145U, 0xDE50A56U,
    0x0C66E61AU, 0xF8CA1D09U, 0x1193EB2FU, 0xE53F103CU,
    0x840C894FU, 0x70A0725CU, 0x99F9847AU, 0x6D557F69U,
    0xBF69325U, 0x4B4A6836U, 0xA2139E10U, 0x56BF6503U,
    0xF3D8BD9BU, 0x07744688U, 0xEE2DB0AEU, 0x1A814BBDU,
    0xC832A7F1U, 0x3C9E5CE2U, 0xD5C7AAC4U, 0x216B51D7U,
    0x6BA4E0E7U, 0x9E081BF4U, 0x7651EDD2U, 0x82FD16C1U,
    0x504EFA8DU, 0xA4E2019EU, 0x4DBBF7B8U, 0xB9170CABU,
    0x1C70D433U, 0xE8DC2F20U, 0x0185D906U, 0xF5292215U,
    0x279ACE59U, 0xD336354AU, 0x3A6FC36CU, 0xCEC3387FU,
    0xF808F18AU, 0x0CA40A99U, 0xE5FDFCBFU, 0x115107ACU,
    0xC3E2EBE0U, 0x374E10F3U, 0xDE17E6D5U, 0x2ABB1DC6U,
    0x8FDC555EU, 0x7B703E4DU, 0x9229C86BU, 0x6853378U,
    0xB436DF34U, 0x409A2427U, 0xA9C3D201U, 0x5D6F2912U,
    0x17A09822U, 0xE30C6331U, 0x0A559517U, 0xFFE96E04U,
    0x2C4A8248U, 0xD8E6795BU, 0x31BF8F7DU, 0xC513746EU,
    0x6074ACF6U, 0x94D857E5U, 0x7D81A1C3U, 0x892D5AD0U,
    0x5B9EB69CU, 0xAF324D8FU, 0x466BBBA9U, 0xB2C740BAU,
    0xD3F4D9C9U, 0x275822DAU, 0xCE01D4FCU, 0x3AAD2FEFU,
    0xE81EC3A3U, 0x1CB238B0U, 0xF5EBCE96U, 0x01473585U,
    0xA420ED1DU, 0x508C160EU, 0xB9D5E028U, 0x4D791B3BU,
    0x9FCAF777U, 0x6B660C64U, 0x823FFA42U, 0x76930151U,
    0x3C5CB061U, 0xC8F04B72U, 0x21A9BD54U, 0xD5054647U,
    0x07B6AA0BU, 0xF31A5118U, 0x1A43A73EU, 0xEEFF5C2DU,
    0x4B8884B5U, 0xBF247FA6U, 0x567D8980U, 0xA2D17293U,
    0x70629EDFU, 0x84CE65CCU, 0x6D9793E9U, 0x993B68F9U
};

```

Figure B.4 – Table du SC-32

B.8 SID

Toutes les sources des données relatives à la sécurité (SDSRC) doivent être identifiées par un identificateur source (SID). Il doit s'agir d'une valeur UNSIGNED32 calculée sous la forme d'une signature SC-32 de la structure de données suivante (comme indiqué à la Figure B.5). La valeur initiale (de base) doit être 'FFFFFFFF'H.

```

SID_STRUCT ::= RECORD
{
    SMI                UINT32          -- user defined safety message
                                   identifier. Shall be unique within
                                   a consist.
                                   value: 1..'FFFFFFFF'H (0 = reserved)

    reserved01         UINT16          -- reserved, shall be set to 0.

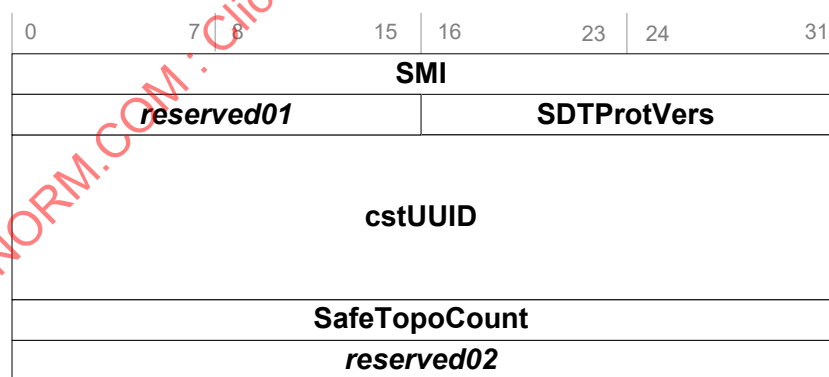
    SDTProtVers        UINT16          -- version of the SDTv2 protocol.
                                   shall be set to '0002'H

    cstUUID            ARRAY[16] OF   -- unique consist identifier
        UINT8

    SafeTopoCount      UINT32          -- safe topography counter (STC),
unique                                     identification of the actual train
                                   composition. Provided by the
                                   communication subsystem. To be set
                                   to 0 for consist network internal
                                   communication

    reserved02         UINT32          -- reserved, shall be set to 0.
}

```



IEC

Figure B.5 – Calcul du SID

Sauf spécification contraire, la valeur SafeTopoCount doit être identique à la valeur opTrnTopoCnt, conformément au 5.3.3.2.13 relatif à la communication dans l'ensemble du train via l'ETB.

Les valeurs SMI de 1 à 999 doivent être réservées pour les VDP génériques spécifiés dans la présente partie de la norme.

Les éléments SDSINK peuvent calculer les SID des VDP prévus grâce aux informations récupérées dans la TTDB. Seul le SMI doit être préconfiguré.

EXEMPLE Un élément SDSINK prévoyant un VDP avec la valeur SMI = 4861 issue du véhicule de tête recherche le paramètre CstUUID du véhicule de tête et le paramètre opTrnTopoCnt dans la TTDB puis calcule le SID.

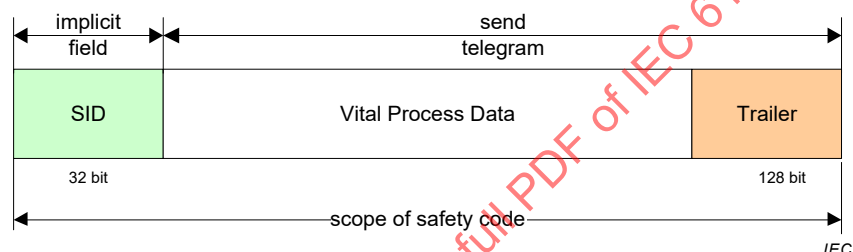
Les éléments SDSINK prévoyant des VDP issus d'un élément SDSRC redondant doivent calculer deux SID car les SMI doivent être différents (voir B.12.3.2).

B.9 Paquets de données stratégiques

Un élément SDSRC doit encapsuler les données critiques pour la sécurité dans un paquet de données stratégiques (VDP). Les VDP transférés sur l'ETB (VDP ETB) sont transmis avec les données utilisateur via un télégramme de données de processus TRDP.

Un VDP peut également être transmis avec la section utilisateur via un télégramme de données de messagerie TRDP, auquel cas les restrictions énumérées dans X doivent être prises en considération.

Le VDP ETB est défini conformément à la Figure B.6:



Anglais	Français
implicit field	zone implicite
send telegram	envoyer télégramme
Vital Process Data	Données de processus stratégiques
SID	SID
Trailer	Queue
scope of safety code	Domaine d'application du code de sécurité

Figure B.6 – VDP ETB

Le VDP ETB doit être représenté par la structure de données ci-dessous, comme indiqué à la Figure B.7:

```

ETB_VDP ::= RECORD
{
    VitalProcessData      ARRAY [VDPSize] of UINT8
                        -- user defined data set
                        -- with safe and non-safe process data
                        -- VDPsize = 0 .. 984 (padding with
                        -- trailing bytes may be necessary to
                        -- have a total
                        -- length of a multiple of 32 bits).

    reserved01            UINT32      -- reserved, shall be set to 0.

    reserved02            UINT16      -- reserved, shall be set to 0.

    UserDataVersion       VERSION    -- version of the vital process data part

    SafeSequCount         UINT32      -- safe sequence counter (SSC),

```

implemented as a counter which is incremented each time a new VDP is generated and stored to the communication channel interface

```

SafetyCode      UINT32      -- SC-32, computed over VDP starting
                           with most significant byte of
                           VitalProcessData up to the
                           SafeSequCount.
                           Seed value: SID
}

```

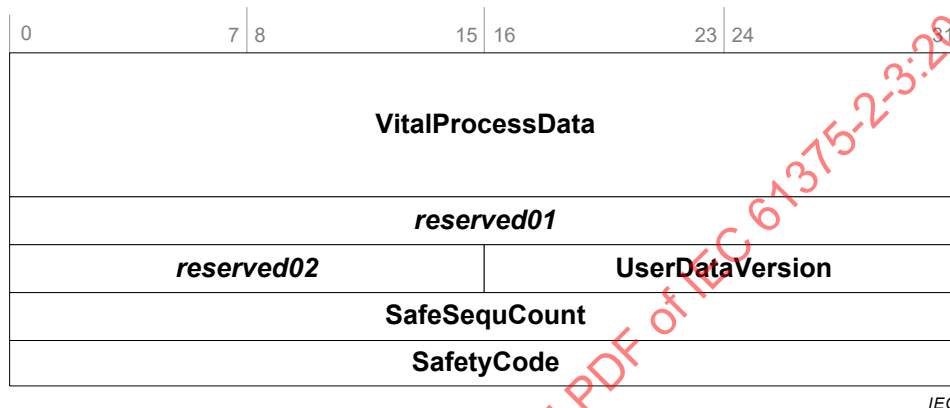


Figure B.7 – Format du VDP ETB

B.10 Exclusivité

L'appareil terminal qui exécute le protocole SDTv2 ne doit pas mettre en œuvre un autre protocole de communication utilisant le même SC-32 ou le même codage que le SID.

B.11 Paramètres temporels de configuration

Le protocole SDTv2 exige un ensemble de paramètres temporels (de configuration) qui sont mentionnés ci-après. Les sections suivantes proposent une spécification plus détaillée de ces paramètres.

T_{tx_period}	Période d'envoi du VDP, telle que définie pour l'élément SDSRC;
T_{rx_period}	Période de réception du VDP (échantillonnage), telle que définie pour l'élément SDSINK;
T_{rx_safe}	Durée maximale de tolérance de l'élément SDSINK de l'absence de nouvelles données stratégiques (ou actualisées);
T_{guard}	Durée pendant laquelle l'élément SDSINK détecte la présence non souhaitée de plusieurs éléments SDSRC actifs en cas d'élément SDSRC redondant.

B.12 Source de données sécurisées (SDSRC)

B.12.1 Généralités

Le présent article définit les exigences de protocole relatives aux sources de données sécurisées. Une source de données sécurisées doit produire des VDP: ceux-ci sont générés puis transmis à la couche communication à des fins de transmission.

B.12.2 Préparation des données sécurisées (application)

Les exigences suivantes définissent certaines conditions d'application. En général, l'application est chargée de fournir les données de processus stratégiques qui doivent être envoyées avec la SDTV2. Deux classes de données d'entrée sont identifiées:

- a) **données continues:** elles sont caractérisées par leur valeur qui varie de façon plus ou moins continue dans le temps (signal de vitesse par exemple). Seuls certains échantillons de ces données doivent être transmis. Une fois échantillonnées, les valeurs des données restent inchangées jusqu'à l'échantillon suivant (principe d'échantillonnage-blocage). La période pendant laquelle la valeur des données échantillonnées reste inchangée est ainsi appelée "durée du signal échantillon" (T_{sig});
- b) **données discrètes:** elles sont caractérisées par leur valeur qui varie en fonction des événements (signal d'ouverture ou de fermeture des portes par exemple). Les différentes valeurs de ces données doivent être transmises sous peine de perdre les informations relatives à la sécurité. La période minimale pendant laquelle la valeur du signal reste inchangée est également appelée "durée du signal échantillon" (T_{sig}). A la différence des signaux d'entrée continus pour lesquels T_{sig} correspond précisément à une période d'échantillonnage, T_{sig} peut correspondre à un multiple de ladite période avec les signaux discrets.

De ce fait, T_{sig} désigne la période pendant laquelle l'échantillon de la valeur des données reste inchangé dans l'interface de l'application SDTV2 de l'élément SDSRC.

Lorsque le VDP contient plusieurs éléments de données discrètes (une situation potentiellement classique), la durée du signal échantillon peut être différente d'un élément à l'autre, celle-ci constituant une propriété inhérente aux éléments de données source. La valeur de T_{sig_min} correspond à la durée du signal échantillon la plus petite relevée pour tous les éléments de données d'un VDP.

L'application doit garantir que chaque modification de la valeur d'un élément de données discrètes (le signal) est échantillonnée et que chaque échantillon reste inchangé dans l'interface de l'application SDTV2 pendant une période T_{sig} .

Le taux d'échantillonnage de l'application pour l'élément de données d'entrée doit être supérieur au taux de modification (fréquence) correspondant.

NOTE La valeur de T_{sig} peut varier d'un élément de données d'entrée à l'autre.

B.12.3 Envoi de données sécurisées

B.12.3.1 Protocole

Les données relatives à la sécurité doivent être envoyées avec le paramètre VitalProcessData intégré au paquet de données stratégiques (VDP).

Le producteur (SDSRC) du VDP doit générer régulièrement le VDP selon une durée de cycle T_{tx_period} .

NOTE 1 La valeur T_{tx_period} sera définie par l'application (dans le cadre de l'IEC 61375-2-4 par exemple).

Le choix de la valeur T_{tx_period} doit respecter la condition suivante: