PD IEC TS 62453-43:2024



BSI Standards Publication

Field device tool (FDT) interface specification

Part 43: Object model integration profile — CLI and HTML

Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

National foreword

This Published Document is the UK implementation of IEC TS 62453-43:2024.

The UK participation in its preparation was entrusted to Technical Committee GEL/65/5, Devices and integration in enterprise systems.

A list of organizations represented on this committee can be obtained on request to its committee manager.

Contractual and legal considerations

This publication has been prepared in good faith, however no representation, warranty, assurance or undertaking (express or implied) is or will be made, and no responsibility or liability is or will be accepted by BSI in relation to the adequacy, accuracy, completeness or reasonableness of this publication. All and any such responsibility and liability is expressly disclaimed to the full extent permitted by the law.

This publication is provided as is, and is to be used at the recipient's own risk.

The recipient is advised to consider seeking professional guidance with respect to its use of this publication.

This publication is not intended to constitute a contract. Users are responsible for its correct application.

This publication is not to be regarded as a British Standard.

© The British Standards Institution 2025
Published by BSI Standards Limited 2025

ISBN 978 0 539 21171 9

ICS 25.040.40; 35.100.05; 35.110

Compliance with a Published Document cannot confer immunity from legal obligations.

This Published Document was published under the authority of the Standards Policy and Strategy Committee on 31 January 2025.

Amendments/corrigenda issued since publication

Date	Text affected

Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

IEC TS 62453-43

Edition 1.0 2024-11

TECHNICAL SPECIFICATION

colour inside

Field device tool (FDT) interface specification – Part 43: Object model integration profile – CLI and HTML

INTERNATIONAL ELECTROTECHNICAL COMMISSION

ICS 25.040.40; 35.100.05; 35.110

ISBN 978-2-8327-0035-8

Warning! Make sure that you obtained this publication from an authorized distributor.

® Registered trademark of the International Electrotechnical Commission

Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

CONTENTS

FOREWORD 19
INTRODUCTION 21
1 Scope 23
2 Normative references 23
3 Terms, definitions, abbreviated terms and conventions 23
3.1 Terms and definitions 23
3.2 Abbreviated terms 30
3.3 Conventions 31
4 Implementation concept 31
4.1 General 31
4.2 Implementation of abstract FDT Object model 31
4.3 FDT Frame Application (FA) 32
4.3.1 General 32
4.3.2 System communication 33
4.4 DTM Business Logic 34
4.4.1 General 34
4.4.2 DTM, DTM Device Type, and Device Ident Info 35
4.4.3 Device data info 36
4.4.4 Process data info 36
4.4.5 Diagnostic data info 37
4.4.6 Network management info 37
4.4.7 Function info 37
4.4.8 Report info 37
4.4.9 Document reference info 37
4.5 DTM functions 38
4.5.1 DTM User Interface 38
4.5.2 Function access control 38
4.5.3 User interaction in DTM WebUIs 38
4.5.4 Command functions 39
4.5.5 Static Function 39
4.6 Communication Channel 39
4.7 DTM categories 41
4.7.1 General 41
4.7.2 Device DTM 41
4.7.3 Communication DTM (CommDTM) 41
4.7.4 Gateway DTM 42
4.7.5 Composite Device DTM 43
4.7.6 Module DTM 44
4.7.7 Block Type Manager 44
4.8 User management 46
4.8.1 General 46
4.8.2 Multi-user access 46
4.8.3 User levels 46
4.9 FDT and system topology 49
4.9.1 General 49
4.9.2 Topology management 50
4.9.3 Address management 52

AVA Documents Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

4.9.4 Configuration of fieldbus master or communication scheduler 53
4.9.5 Data exchange between Frame Applications 54
4.10 Modularity 54
4.11 FDT communication 55
4.11.1 General 55
4.11.2 Point-to-point communication 56
4.11.3 Nested communication 57
4.11.4 Dynamic changes in network 58
4.12 Identification 59
4.12.1 DTM instance identification 59
4.12.2 System GUI label 59
4.12.3 Hardware identification 59
4.13 Scanning and DTM assignment 60
4.13.1 Scanning introduction 60
4.13.2 Scanning 60
4.13.3 DTM assignment 61
4.13.4 Manufacturer-specific device identification 61
4.14 DTM data persistence and synchronization 62
4.14.1 Persistence overview 62
4.14.2 Relations of DTMDataSet 62
4.14.3 DTMDataSet structure 63
4.14.4 Types of persistent DTM data 65
4.14.5 Data synchronization 65
4.15 Device data and IO information 66
4.15.1 Exposing device data and IO information 66
4.15.2 Data access control 67
4.15.3 Routed IO information 69
4.15.4 Comparison of DTM and device data 70
4.15.5 PLC tool support 70
4.15.6 Support for multirole devices 72
4.16 Clone of DTM instances 73
4.16.1 General 73
4.16.2 Replicating a part of topology with Parent DTM and a subset of its Child DTMs 73
4.16.3 Cloning of a DTM without its children 73
4.16.4 Delayed cloning 73
4.17 Lifecycle concepts 74
4.18 Audit trail 74
4.18.1 General 74
4.18.2 Audit trail events 74
5 Technical concepts 75
5.1 General 75
5.2 Support of HTML versions 77
5.3 Support of JavaScript versions 77
5.4 Support of .NET Common Language Runtime versions 78
5.4.1 General 78
5.4.2 DTM rules 78
5.4.3 Frame Application rules 78
5.5 Support for 32-bit and 64-bit target platforms 79

AVA Documents Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

5.6 Object activation and deactivation 79
5.6.1 General 79
5.6.2 Assembly loading and object creation 79
5.6.3 Assembly dependencies 79
5.6.4 Shared assemblies 80
5.6.5 Object deactivation and unloading 81
5.7 Datatypes 81
5.7.1 General 81
5.7.2 Serialization / deserialization 82
5.7.3 Optional elements 82
5.7.4 Verify 83
5.7.5 Clone 83
5.7.6 Equals 84
5.7.7 Lists 84
5.7.8 Nullable 84
5.7.9 Enumeration 84
5.7.10 Protocol-specific datatypes 84
5.7.11 Custom datatypes 87
5.8 General object interaction 88
5.8.1 General 88
5.8.2 Decoupling of FDT Objects 88
5.8.3 Parameter interchange with .NET datatypes 90
5.8.4 Interaction patterns 90
5.8.5 Properties 90
5.8.6 Synchronous methods 90
5.8.7 Asynchronous methods 90
5.8.8 Events pattern 97
5.8.9 Exception handling 97
5.9 Threading 101
5.9.1 General 101
5.9.2 Races 101
5.9.3 Locks 102
5.9.4 Deadlocks 102
5.10 Threading rules 102
5.10.1 Implementation rules 102
5.10.2 Avoiding deadlocks 103
5.10.3 FDT Object interaction rules 103
5.11 Localization support 104
5.11.1 General 104
5.11.2 Access to localized resources and culture-dependent functions 104
5.11.3 Handling of cultures 104
5.11.4 Switching the User Interface language 105
5.12 DTM User Interface implementation 105
5.12.1 General 105
5.12.2 Private dialogs 106
5.12.3 Modal DTM WebUI 106
5.13 DTM User Interface hosting 106
5.13.1 General 106
5.13.2 Hosting DTM WebUI 107

AVA Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

5.14 Static Function implementation 110
5.15 Persistence 112
5.15.1 Overview 112
5.15.2 Data format 113
5.15.3 Adding / reading / writing / deleting of data 113
5.15.4 Searching for data 115
5.16 Comparison of DTM and device data 116
5.16.1 Comparison of datasets using IDeviceData / IInstanceData 116
5.16.2 Comparison of datasets using IComparison 117
5.17 Tracing 117
5.18 Report generation 117
5.18.1 Introduction 117
5.18.2 Report types 118
5.18.3 DTM report data format 118
5.18.4 Report data exchange 119
5.19 Security 119
5.19.1 General 119
5.19.2 Strong naming of assemblies 119
5.19.3 Identification of origin 120
5.19.4 Code access security 120
5.19.5 Validation of FDT compliance certification 120
6 FDT Objects and interfaces 122
6.1 General 122
6.2 Frame Application 123
6.2.1 General 123
6.2.2 Frame Application Business Logic 123
6.2.3 Frame Application WebUI 126
6.3 DTM Business Logic 126
6.3.1 DTM BL interfaces 126
6.3.2 State machines related to DTM BL 131
6.3.3 State machine of instance data 138
6.4 DTM User Interface 142
6.4.1 DTM WebUI 142
6.5 Communication Channel 143
6.6 Availability of interface methods 145
7 FDT datatypes 146
7.1 General 146
7.2 Datatypes – Base 146
7.3 General datatypes 147
7.4 Datatypes – DtmInfo / TypeInfo 147
7.5 Datatypes – DeviceIdentInfo 149
7.6 Datatypes for installation and deployment 154
7.6.1 Datatypes – DtmPackageManifest 154
7.6.2 Datatypes – DtmManifest 155
7.6.3 Datatypes – DtmWebUiManifest 156
7.7 Datatypes – Communication 157
7.8 Datatypes – BusCategory 163
7.9 Datatypes – Device / instance data 163
7.9.1 General 163

AVA Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

7.9.2 Datatypes used in reading and writing DeviceData 170
7.10 Datatypes for export and import 172
7.10.1 Datatypes – TopologyImportExport 172
7.10.2 Datatypes – ImportExportDataset 174
7.11 Datatypes for process data description 174
7.11.1 Datatypes – ProcessDataInfo 174
7.11.2 Datatypes – Process Image 179
7.12 Datatypes – Address information 181
7.13 Datatypes – NetworkDataInfo 184
7.14 Datatypes – DTM functions 186
7.15 Datatypes – DTM messages 188
7.16 Datatypes – CommunicationChannelInfo 189
7.17 Datatypes – HardwareIdentification and scanning 191
7.17.1 General 191
7.17.2 Datatypes – DeviceScanInfo 191
7.17.3 Example – HardwareIdentification and scanning for HART® 192
7.18 Datatypes – DTM report types 193
7.19 Information related to device modules in a monolithic DTM 194
8 Workflows 196
8.1 General 196
8.2 Instantiation, loading and release 196
8.2.1 Finding a DTM BL object 196
8.2.2 Instantiation of a new DTM BL 198
8.2.3 Configuring access rights 200
8.2.4 Loading a DTM BL 201
8.2.5 Loading a DTM with Expert user level 202
8.2.6 Release of a DTM BL 204
8.3 Persistent storage of a DTM 204
8.3.1 Saving instance data of a DTM 204
8.3.2 Copy and versioning of a DTM instance 205
8.3.3 Dataset commit failed 206
8.3.4 Export a DTM dataset to file 206
8.4 Locking and DataTransactions in multi-user environments 207
8.4.1 General 207
8.4.2 Propagation of changes 208
8.4.3 Synchronizing DTMs in multi-user environments 209
8.5 Execution of DTM Functions 211
8.5.1 General 211
8.5.2 Finding a DTM WebUI 211
8.5.3 Instantiation of a DTM WebUI 212
8.5.4 Release of a DTM WebUI 213
8.5.5 Execution of command functions 214
8.5.6 Opening of documents 215
8.5.7 Interaction between DTM WebUI and DTM Business Logic 216
8.5.8 Interaction between DTM Business Logic and DTM WebUI 218
8.5.9 Interaction between DTM WebUI and DTM Business Logic with Cancel 218
8.5.10 Retrieving information about available Static Functions 219
8.5.11 Executing a Static Function 221
8.6 DTM communication 223

AVA Documents Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

8.6.1 General 223
8.6.2 Establishing a communication connection 223
8.6.3 Cancel establishment of communication connection 224
8.6.4 Communicating with the device 225
8.6.5 Frame Application or Child DTM disconnect a device 226
8.6.6 Terminating a communication connection 227
8.6.7 DTM aborts communication connection 228
8.6.8 Communication Channel aborts communication connection 229
8.7 Nested communication 230
8.7.1 General 230
8.7.2 Communication request for a nested connection 231
8.7.3 Propagation of errors for a nested connection 232
8.8 Topology planning 233
8.8.1 General 233
8.8.2 Adding a DTM to the topology 233
8.8.3 Removing a DTM from topology 234
8.8.4 Frame Application creates topology 235
8.8.5 DTM generates sub-topology 236
8.8.6 Physical Layer and DataLinkLayer 238
8.9 Instantiation, configuration, move and release of Child DTMs 238
8.9.1 General 238
8.9.2 Instantiation and configuration of Child DTM BL 239
8.9.3 Interaction between Parent DTM and Child DTM 240
8.9.4 Interaction between Parent DTM and Child DTM using IDtmMessaging 242
8.9.5 Parent DTM moves a Child DTM 242
8.9.6 Parent DTM removes Child DTM 243
8.10 Topology scan 244
8.10.1 General 244
8.10.2 Scan of network topology 244
8.10.3 Cancel topology scan 245
8.10.4 Scan based DTM assignment 246
8.10.5 Manufacturer-specific device identification 247
8.11 Configuration of communication networks 249
8.11.1 Configuration of a fieldbus master 249
8.11.2 Integration of a passive device 250
8.12 Using IO information 250
8.12.1 Assignment of symbolic name to process data 250
8.12.2 Creation of Process Image 252
8.12.3 Validation of changes in process image while PLC is running 253
8.12.4 Changing of variable names using process image interface 254
8.13 Managing addresses 255
8.13.1 Set DTM address with user interface 255
8.13.2 Set DTM addresses without user interface 257
8.13.3 Display or modify addresses of all Child DTMs with user interface 258
8.14 Device-initiated data transfer 259
8.15 Reading and writing data 260
8.15.1 Read/write instance data 260
8.15.2 Read/write device data 261
8.16 Comparing data 261

AVA Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

8.16.1 Comparing device dataset and instance dataset 263
8.16.2 Comparing different instance datasets 263
8.17 Reassigning a different DtmDeviceType at a device node 264
8.17.1 General 264
8.17.2 DTM detects a change in connected device type 265
8.17.3 Search matching DtmDeviceTypes after incompatible device exchange 267
8.17.4 Reassign DtmDeviceType after incompatible device exchange 268
8.18 Copying part of FDT Topology 270
8.18.1 Cloning of a single DTM without Children 270
8.18.2 Cloning of a DTM with all its Children 271
8.19 Sequences for audit trail 271
8.19.1 General 271
8.19.2 Audit trail of parameter modifications in instance dataset 271
8.19.3 Audit trail of parameter modifications in device dataset 272
8.19.4 Audit trail of function calls 273
8.19.5 Audit trail of general notification 274
9 Installation 274
9.1 General 274
9.2 Common rules 274
9.2.1 Predefined installation paths 274
9.2.2 Predefined web-server paths 276
9.2.3 Manifest files 277
9.2.4 Paths in manifest files 277
9.2.5 DTM Installation package format 277
9.2.6 Digital signatures of package components 278
9.3 Installation of FDT core assemblies 278
9.4 Installation of communication protocols 278
9.4.1 General 278
9.4.2 Registration 279
9.4.3 Protocol manifest 279
9.4.4 NuGet package 280
9.5 Installation of DTMs 281
9.5.1 General 281
9.5.2 Registration 282
9.5.3 DTM manifest 283
9.5.4 DTM WebUI manifest 284
9.5.5 WebUI container files 284
9.6 DTM package file 285
9.6.1 Naming convention 285
9.6.2 Structure (physical model) 286
9.6.3 Relationships (logical model) 287
9.6.4 Core properties 288
9.6.5 License file 289
9.6.6 Readme file 289
9.6.7 Icon file 289
9.6.8 DTM package manifest 290
9.6.9 DTM device identification manifest 292
9.6.10 DTM package file creation rules 293
9.6.11 Countersignatures 293

AVA Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

9.6.12 Referenced NuGet packages 293
9.7 DTM deployment 294
9.8 Paths and file information 295
9.8.1 Path information provided by a DTM via IFunction 295
9.8.2 Paths and persistence 296
9.8.3 Multi-user systems 296
10 Life cycle concept 296
10.1 General 296
10.2 Technical concept 296
10.2.1 General 296
10.2.2 DtmManifest / DtmInfo 297
10.2.3 TypeInfo 298
10.2.4 Supported DataSet formats 299
10.2.5 DeviceIdentInfo 299
10.2.6 Dataset 300
10.2.7 DeviceScanInfo 300
10.3 DTM installation 301
10.3.1 General 301
10.3.2 Handling of DTM installations 302
10.4 Life cycle scenarios 302
10.4.1 Overview 302
10.4.2 Search for device type in DTM packages 304
10.4.3 Search for installed DTMs 304
10.4.4 Dataset migration for reassigned DTM 306
11 Frame Application architectures 307
11.1 General 307
11.2 Standalone application 307
11.3 Remote User Interface 307
11.4 Distributed multi-user application 308
11.5 OPC UA 309
11.6 Web services 310
Annex A (normative) Use case model 312
A.1 Use case model overview 312
A.2 Actors 312
A.3 Use cases 313
A.3.1 Use case overview 313
A.3.2 Observation use cases 314
A.3.3 Operation use cases 315
A.3.4 Maintenance use cases 317
A.3.5 Planning use cases 321
A.3.6 Main operation 324
A.3.7 OEM service 324
A.3.8 Administration 325
Annex B (normative) FDT interface definition and datatypes 326
Annex C (normative) Mapping of services to interface methods 327
C.1 General 327
C.2 DTM services 327
C.3 Presentation object services 331

AVA Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

C.4 General channel services 331
C.5 Process Channel services 332
C.6 Communication Channel Services 332
C.7 Frame Application Services 333
Annex D (normative) FDT version interoperability guide 337
D.1 Overview 337
D.2 General 337
D.3 Component interoperability 338
Annex E (normative) Definition of JavaScript APIs 339
E.1 General 339
E.2 Request / Response datatypes 339
E.3 Message datatypes 340
E.3.1 SendMessages 340
E.3.2 Request datatype 340
E.3.3 Response datatype 341
E.4 API for WebUIs 341
E.5 API for tracing 344
Annex F (informative) Implementation hints 347
F.1 IAsyncResult pattern 347
F.1.1 Overview 347
F.1.2 BeginOperationName 347
F.1.3 EndOperationName 347
F.1.4 Asynchronous Completed callback 348
F.2 Threading best practices 348
F.3 Testing DTM messages in regard to JSON serialization/deserialization 349
Annex G (informative) Comparison of IEC TR 62453-42 and IEC TS 62453-43 351
Annex H (informative) Physical layer examples 352
H.1 General 352
H.2 Interbus S 352
H.3 PROFIBUS 352
H.4 PROFINET 353
Annex I (informative) Predefined SemanticIds 354
I.1 General 354
I.2 Data 354
I.3 Images 354
I.4 Documents 355
Annex J (informative) Standard StaticFunctions 356
J.1 General 356
J.2 StaticFunction GetDeviceStatus 356
J.3 StaticFunction GetProcessValue 357
Bibliography 360

Figure 1 – Relation of IEC TS 62453-43 to the IEC 62453 series 21
Figure 2 – IEC TS 62453-43 Object model 32
Figure 3 – Frame Application 32
Figure 4 – Frame Application with integrated Communication Channel 34
Figure 5 – DTM Business Logic 34

AVA Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

Figure 6 – DTM, Device Type and Device Ident Info 35
Figure 7 – Process data info 36
Figure 8 – Communication Channel 40
Figure 9 – Device DTM 41
Figure 10 – Communication DTM 42
Figure 11 – Gateway DTM 42
Figure 12 – Composite Device DTM 43
Figure 13 – Module DTM 44
Figure 14 – Block Type Manager 45
Figure 15 – Logical topology and physical topology 50
Figure 16 – FDT and logical topology 50
Figure 17 – DTMs and physical topology 51
Figure 18 – Address setting via DTM WebUI 53
Figure 19 – Fieldbus master configuration tool as part of a DTM 54
Figure 20 – Point-to-point communication 57
Figure 21 – Nested communication 58
Figure 22 – Identification of connected devices 60
Figure 23 – Fieldbus scanning 61
Figure 24 – FDT storage and synchronization mechanism 62
Figure 25 – Relation between DTMDataSet, DTM instance, and device 63
Figure 26 – DTMDataSet structure 64
Figure 27 – Data Synchronization 65
Figure 28 – Example for same organization of FunctionInfo and DataInfo 67
Figure 29 – Routed IO information 69
Figure 30 – Process Image 70
Figure 31 – Transfer of layout information using IProcessImage 71
Figure 32 – Multirole device 72
Figure 33 – FDT .NET Assemblies 76
Figure 34 – FDT Object implementation 77
Figure 35 – Example: Assembly dependencies 80
Figure 36 – Example: Datatype definition 82
Figure 37 – Example: Data cloning 83
Figure 38 – Example: Methods without data cloning 84
Figure 39 – Protocol-specific datatypes 85
Figure 40 – Protocol manifest and type info attributes 86
Figure 41 – Example: Protocol assembly attributes 87
Figure 42 – Example: Handling of protocol-specific assemblies in Frame Application 87
Figure 43 – Example: Handling of KnownType for custom data types 88
Figure 44 – Decoupled FDT Objects in IEC TS 62453-43 89
Figure 45 – IAsyncResult pattern: blocking call 91
Figure 46 – Example: Blocking use of asynchronous interface 92
Figure 47 – IAsyncResult pattern (simplified): blocking call 92
Figure 48 – IAsyncResult pattern: non-blocking call 93

AVA Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

Figure 49 – Example: Non-blocking use of asynchronous interface 94
Figure 50 – IAsyncResult pattern (simplified depiction): non-blocking call 94
Figure 51 – IAsyncResult pattern: canceling an operation 95
Figure 52 – IAsyncResult pattern: providing progress events 96
Figure 53 – General concept for hosting DTM WebUI 107
Figure 54 – Example: Integration of a DTM WebUI into a Frame WebUI 108
Figure 55 – Example: Hosting a DTM WebUI in a WPF Frame Application 109
Figure 56 – Example: Integrating the WebData connector in a DTM WebUI 110
Figure 57 – Relation of StaticFunctionDescription to Static Function 111
Figure 58 – DTMDataset structure 112
Figure 60 – Example: Writing of DTM data in DTMDataSubset 114
Figure 61 – Example: Reading of DTM data from a DTMDataSubset 115
Figure 62 – Example: Creation of a BulkData.DTMDataSubset with descriptor 116
Figure 63 – Example: Searching for DTMDataSubsets with specific descriptor 116
Figure 64 – Skeleton of a DTM-specific report fragment 119
Figure 65 – Example: Conformity record file 121
Figure 66 – Example: checking conformity record file 122
Figure 67 – Frame Application BL interfaces 124
Figure 68 – Frame Application WebUI 126
Figure 69 – DTM Business Logic interfaces (Part 1) 127
Figure 70 – DTM Business Logic interfaces (Part 2) 128
Figure 71 – State machine of DTM BL 133
Figure 72 – Online state machine of DTM 135
Figure 73 – Modifications of data through a DTM 139
Figure 74 – ModifiedInDtm: State machine of instance data 140
Figure 75 – ModifiedInDevice: State machine related to device data 141
Figure 76 – DTM WebUI interfaces 142
Figure 77 – Example: Definition of DtmWebUi object 142
Figure 78 – Example: Access to DtmWebUi by Frame WebUI 142
Figure 79 – Communication Channel interfaces 143
Figure 80 – Connection state machine 144
Figure 81 – FdtDatatype and FdtList 146
Figure 82 – DtmInfo / TypeInfo – datatypes 148
Figure 83 – DeviceIdentInfo – datatypes 150
Figure 84 – DeviceIdentInfo – Example for HART® 151
Figure 85 – Example: DeviceIdentInfo creation 153
Figure 86 – Example: Using DeviceIdentInfo 154
Figure 87 – Example: DeviceIdentInfoTypeAttribute 154
Figure 88 – DtmPackageManifest – datatypes 155
Figure 89 – DtmManifest – datatypes 156
Figure 90 – DtmWebUiManifest – datatypes 157
Figure 91 – Communication datatypes – Connect 158
Figure 92 – Communication datatypes – Transaction 161

AVA Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

Figure 93 – Communication datatypes – Disconnect 159
Figure 94 – Communication datatypes – Subscribe 159
Figure 95 – Communication datatypes – Scanning 160
Figure 96 – Communication datatypes – Address setting 160
Figure 97 – Example: Communication – Connect for HART® 162
Figure 98 – Example: Communication – CommunicationType for HART 162
Figure 99 – BusCategory – datatypes 163
Figure 100 – Device / instance data – datatypes 164
Figure 101 – Example: Providing information on data of a HART® device 166
Figure 102 – Example: Providing information on module data of a PROFIBUS device 167
Figure 103 – Example: Providing information on data 168
Figure 104 – Example: Providing information on structured data 169
Figure 105 – EnumInfo – datatype 170
Figure 106 – Read and Write Request – datatypes 170
Figure 107 – ResponseInfo – datatype 171
Figure 108 – TopologyImportExport – datatypes 173
Figure 109 – ImportExportDataset – datatypes 174
Figure 110 – ProcessDataInfo – datatypes 175
Figure 111 – IOSignalInfo – datatypes 176
Figure 112 – Example: ProcessDataInfo for HART (UML) 177
Figure 113 – Example: ProcessDataInfo creation for HART® 178
Figure 114 – Example: Using ProcessData for HART® 179
Figure 115 – Example: IOSignalInfoType attribute 179
Figure 116 – ProcessImage – datatypes 180
Figure 117 – AddressInfo – datatypes 181
Figure 118 – Example: AddressInfo creation 182
Figure 119 – Example: Using AddressInfo 183
Figure 120 – Example: DeviceAddressTypeAttribute 183
Figure 121 – NetworkDataInfo – datatypes 184
Figure 122 – Example: NetworkDataInfo creation example 185
Figure 123 – Example: NetworkDataInfo using example 186
Figure 124 – Example: NetworkDataTypeAttribute example 186
Figure 125 – DTM Function – datatypes 187
Figure 126 – DTM Messages – datatypes 188
Figure 127 – CommunicationChannelInfo – datatypes 190
Figure 128 – Example: Channel information 191
Figure 129 – DeviceScanInfo – datatypes 192
Figure 130 – Example: HARTDeviceScanInfo – datatype 193
Figure 131 – DTM Report – datatypes 194
Figure 132 – Information related to device modules 195
Figure 133 – Finding a DTM BL object 197
Figure 134 – Instantiation of a new DTM BL 199

AVA Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

Figure 135 – Configuration of user permissions 201
Figure 136 – Loading a DTM BL 202
Figure 137 – Loading a DTM with Expert user level 203
Figure 138 – Release of a DTM BL 204
Figure 139 – Saving data of a DTM 205
Figure 140 – Dataset commit failed 206
Figure 141 – Export a DTM dataset to file 207
Figure 142 – Propagation of changes 209
Figure 143 – Synchronizing DTMs in multi-user environments 210
Figure 144 – Finding a DTM WebUI 212
Figure 145 – Instantiation of a DTM WebUI 213
Figure 146 – Release of a DTM WebUI 214
Figure 147 – Execute a command function 215
Figure 148 – Opening a document 216
Figure 149 – Interaction triggered by the DTM WebUI 217
Figure 150 – Interaction triggered between DTM BL and DTM WebUI 218
Figure 151 – Interaction triggered and canceled by the DTM WebUI 219
Figure 152 – Retrieving information about available Static Functions 220
Figure 153 – Example: Information about available Static Functions 221
Figure 154 – Executing a Static Function 222
Figure 155 – Establishing a communication connection 224
Figure 156 – DTM cancels ongoing Connect operation 225
Figure 157 – Communicating with the device 226
Figure 158 – Child DTM disconnects 227
Figure 159 – Child DTM terminates a connection 228
Figure 160 – Child DTM aborts a connection 229
Figure 161 – Communication Channel aborts a connection 230
Figure 162 – Example: Nested communication behavior 231
Figure 163 – Example: Nested communication data exchange 232
Figure 164 – Add DTM to topology 234
Figure 165 – Removing a DTM from topology 235
Figure 166 – Frame Application creates topology 236
Figure 167 – DTM generates sub-topology 237
Figure 168 – Instantiation and configuration of Child DTM BL 239
Figure 169 – Interaction between Parent DTM and Child DTM 241
Figure 170 – Interaction using IDtmMessaging 242
Figure 171 – Parent DTM moves a Child DTM 243
Figure 172 – Parent DTM removes Child DTM 244
Figure 173 – Scan of network topology 245
Figure 174 – Cancel topology scan 246
Figure 175 – Scan based DTM assignment 247
Figure 176 – Manufacturer-specific device identification 248
Figure 177 – Configuration of a fieldbus master 249

AVA Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

Figure 178 – Integration of a passive device 250
Figure 179 – Assignment of process data 251
Figure 180 – Creation of process image 253
Figure 181 – Validation of changes while PLC is running 254
Figure 182 – Changing of variable names using process image interface 255
Figure 183 – Set DTM address with WebUI 256
Figure 184 – Set DTM addresses without UI 257
Figure 185 – Display or modify child addresses with UI 258
Figure 186 – Device-initiated data transfer 259
Figure 187 – Read/write instance data 260
Figure 188 – Read/write device data 262
Figure 189 – Comparing device dataset and instance dataset 263
Figure 190 – Compare instance data with persisted dataset 264
Figure 191 – DTM triggers ActiveTypeChanged event 266
Figure 192 – Find matching DtmDeviceTypes after incompatible device exchange 268
Figure 193 – Reassign a DtmDeviceType after incompatible device exchange 269
Figure 194 – Clone DTM without children 270
Figure 195 – Clone DTM with all children 271
Figure 196 – Audit trail of parameter modifications in instance dataset 272
Figure 197 – Audit trail of parameter modifications in device 273
Figure 198 – Audit trail of function calls 273
Figure 199 – Installation paths (with example DTM) 276
Figure 200 – Installed FDT and protocol assemblies 279
Figure 201 – Example: Protocol manifest 280
Figure 202 – NuGet file structure 281
Figure 203 – Search for installed DTMs 282
Figure 204 – Example: DtmManifest 283
Figure 205 – Example: DtmWebUiManifest 284
Figure 206 – Structure of a WebUI container file 285
Figure 207 – DTM package file structure 287
Figure 208 – DTM package file relationships 288
Figure 209 – Example: DtmPackageManifest 291
Figure 210 – Example: DeviceIdentManifest 292
Figure 211 – DTM deployment 294
Figure 212 – Unpacking the DTM package 295
Figure 213 – Overview DTM identification 297
Figure 214 – Identification attributes in DTM package 301
Figure 215 – Search DTM package for list of supported types 304
Figure 216 – Scan installed DTMs 305
Figure 217 – Dataset migration to a reassigned DtmDeviceType 306
Figure 218 – Client / Server Application 308
Figure 219 – Example for distributed multi-user application 309
Figure 220 – OPC UA server based on IEC TS 62453-43 310

AVA Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

Figure 221 – Web server based on IEC TS 62453-43 311
Figure A.1 – Main use case diagram 312
Figure A.2 – Observation use cases 314
Figure A.3 – Operation use cases 316
Figure A.4 – Maintenance use cases 318
Figure A.5 – Planning use cases 322
Figure F.1 – Example: Test for JSON data 349
Figure F.2 – Example: Erroneous DTM-specific message example 349
Figure F.3 – Example: Failing test for erroneous JSON data 350

Table 1 – FDT user levels 47
Table 2 – Role-dependent access rights and user interfaces for DTMs 47
Table 3 – Description of properties related to data access control 68
Table 4 – Supported HTML versions 77
Table 5 – Supported ECMAScript versions 77
Table 6 – Supported .NET Standard library versions 78
Table 7 – Frame Application BL interfaces 125
Table 8 – Frame Application WebUI interfaces 126
Table 9 – DTM Business Logic interfaces 129
Table 10 – Availability of interfaces depending of type of DTM 130
Table 11 – Definition of DTM BL state machine 134
Table 12 – Definition of online state machine 136
Table 13 – Description of instance dataset states 140
Table 14 – Description of dataset states regarding online modifications 141
Table 15 – DTM WebUI interfaces 142
Table 16 – Communication Channel interfaces 143
Table 17 – Definition of connection state machine 144
Table 18 – Availability of DTM BL methods in different states 145
Table 19 – FDT base datatypes 147
Table 20 – FDT General datatypes 147
Table 21 – DtmInfo datatype description 149
Table 22 – DeviceIdentInfo datatype description 150
Table 23 – DeviceIdentInfo – Example for HART® 152
Table 24 – DtmPackageManifest datatype description 155
Table 25 – DtmManifest datatype description 156
Table 26 – DtmWebUiManifest datatype description 157
Table 27 – Communication datatype description 161
Table 28 – BusCategory datatype description 163
Table 29 – DeviceData datatype description 165
Table 30 – Reading and Writing datatype description 170
Table 31 – Reading and Writing datatype description 172
Table 32 – TopologyImportExport datatype description 173
Table 33 – ImportExportDataset datatype description 174

AVA Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

Table 34 – ProcessDataInfo datatype description 175
Table 35 – IOSignalInfo datatype description 176
Table 36 – ProcessImage datatype description 180
Table 37 – AddressInfo datatype description 181
Table 38 – NetworkDataInfo datatype description 185
Table 39 – DTM Function datatype description 188
Table 40 – DTM Messages datatype description 189
Table 41 – CommunicationChannelInfo datatype description 190
Table 42 – DeviceScanInfo datatype description 192
Table 43 – Example: HARTDeviceScanInfo datatype description 193
Table 44 – Reporting datatype description 194
Table 45 – Predefined FDT installation paths 274
Table 46 – Predefined web-server paths 277
Table 47 – Overview of relationships 288
Table 48 – Overview of core properties 289
Table 49 – DTM identification 298
Table 50 – TypeInfo – User readable description of supported types 298
Table 51 – TypeInfo identification 299
Table 52 – DtmType – Dataset support identification 299
Table 53 – Dataset identification 300
Table 54 – DeviceScanInfo – scanned device identification 301
Table 55 – Package information 302
Table 56 – Changing DTM – overview 303
Table A.1 – Actors 313
Table A.2 – Observation use cases 314
Table A.3 – Operation use cases 316
Table A.4 – Maintenance use cases 318
Table A.5 – Planning use cases 322
Table C.1 – General services 327
Table C.2 – DTM service related to installation 327
Table C.3 – DTM service related to DTM information 328
Table C.4 – DTM services related to DTM state machine 328
Table C.5 – DTM services related to function 328
Table C.6 – DTM services related to documentation 329
Table C.7 – DTM services to access the instance data 329
Table C.8 – DTM services to access diagnosis 329
Table C.9 – DTM services to access to device data 330
Table C.10 – DTM services related to network management information 330
Table C.11 – DTM services related to online operation 330
Table C.12 – DTM services related to FDT-Channel objects 330
Table C.13 – DTM services related to import and export 331
Table C.14 – DTM services related to data synchronization 331
Table C.15 – DTM UI state control 331

AVA Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

Table C.16 – General channel service 331
Table C.17 – Channel services for IO related information 332
Table C.18 – Channel services related to communication 332
Table C.19 – Channel services related sub-topology management 333
Table C.20 – Channel services related to functions 333
Table C.21 – Channel services related to scan 333
Table C.22 – FA services related to general event 333
Table C.23 – FA services related to topology management 334
Table C.24 – FA services related to redundancy 334
Table C.25 – FA services related to storage of DTM data 334
Table C.26 – FA services related to DTM data synchronization 334
Table C.27 – FA services related to presentation 335
Table C.28 – FA services related to audit trail 335
Table C.29 – FA services related to sandboxing a DTM 335
Table C.30 – FA services related to sandboxing a Communication Channel 336
Table D.1 – Interoperability between components of different versions 338
Table J.1 – StaticFunctionDescription for GetDeviceStatus 356
Table J.2 – IStaticFunction2:BeginExecute arguments for GetDeviceStatus 357
Table J.3 – Exceptions for GetDeviceStatus 357
Table J.4 – StaticFunctionDescription for GetProcessValue 358
Table J.5 – IStaticFunction2:BeginExecute arguments for GetDeviceStatus 358
Table J.6 – Exceptions for GetProcessValue 359

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

INTERNATIONAL ELECTROTECHNICAL COMMISSION

_____

FIELD DEVICE TOOL (FDT) INTERFACE SPECIFICATION –

Part 43: Object model integration profile – CLI and HTML

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) IEC draws attention to the possibility that the implementation of this document may involve the use of (a) patent(s). IEC takes no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, IEC had not received notice of (a) patent(s), which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at https://patents.iec.ch. IEC shall not be held responsible for identifying any or all such patent rights.

IEC TS 62453-43 has been prepared by subcommittee 65E: Devices and integration in enterprise systems, of IEC technical committee 65: Industrial-process measurement, control and automation. It is a Technical Specification.

The text of this Technical Specification is based on the following documents:

Draft	Report on voting
65E/1115/DTS	65E/1146/RVDTS

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

The language used for the development of this Technical Specification is English.

Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

This document was drafted in accordance with ISO/IEC Directives, Part 2, and developed in accordance with ISO/IEC Directives, Part 1 and ISO/IEC Directives, IEC Supplement, available at www.iec.ch/members_experts/refdocs. The main document types developed by IEC are described in greater detail at www.iec.ch/standardsdev/publications.

A list of all parts in the IEC 62543 series, published under the general title *Field Device Tool (FDT) interface specification*, can be found on the IEC website.

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

- reconfirmed,
- withdrawn, or
- revised.

IMPORTANT – The "colour inside" logo on the cover page of this document 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.

Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

INTRODUCTION

This document is an interface specification for developers of FDT (Field Device Tool) components for function control and data access within a client/server architecture. The specification is a result of an analysis and design process to develop standard interfaces to facilitate the development of servers and clients by multiple vendors that need to interoperate seamlessly.

With the integration of fieldbuses into control systems, there are a few other tasks which need to be performed. In addition to fieldbus-specific and device-specific tools, there is a need to integrate these tools into higher-level system-wide planning tools or engineering tools. In particular, for use in extensive and heterogeneous control systems, the unambiguous definition of engineering interfaces that are easy to use for all those involved is of great importance.

A device-specific software component, called DTM (Device Type Manager), is supplied by the field device manufacturer with its device. The DTM is integrated into engineering tools via the FDT interfaces defined in this specification. The approach to integration, in general, is open for all kind of fieldbuses and thus meets the requirements for integrating different kinds of devices into heterogeneous control systems.

Figure 1 shows how IEC TS 62453-43 is related to the IEC 62453 series.

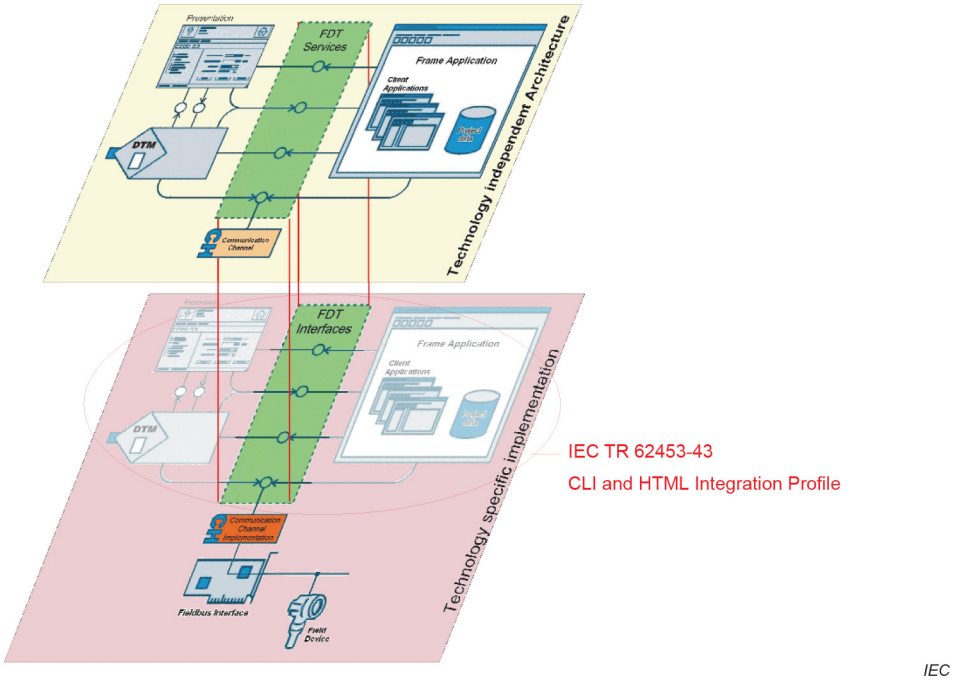


Figure 1 – Relation of IEC TS 62453-43 to the IEC 62453 series

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

The document structure is:

- Clause 3 explains the used terms, definitions and conventions
- Clause 4 introduces the general concepts of IEC TS 62453-43
- Clause 5 describes the technical concepts used to implement IEC TS 62453-43 and how FDT concepts are mapped to .NET Standard
- Clause 6 provides an overview of the FDT Objects, their interfaces and behaviour
- Clause 7 presents an overview of the IEC TS 62453-43 datatypes
- Clause 8 shows the interaction of FDT Objects at runtime
- Clause 9 explains rules related to installation and deployment of DTMs
- Clause 10 explains how FDT life cycle concepts are implemented
- Clause 11 shows examples for Frame Application architectures

Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

FIELD DEVICE TOOL (FDT) INTERFACE SPECIFICATION –

Part 43: Object model integration profile – CLI and HTML

1 Scope

This part of IEC 62453, which is a technical specification, specifies how the common FDT principles are implemented based on the CLI technology and web technologies for graphical user interfaces. The specification includes the object behaviour and object interaction via .NET Standard interfaces and JavaScript APIs. Emphasis has been placed on support of distributed Frame Application architectures.

This document specifies FDT version 3.0.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 62453-1:2024, *Field Device Tool (FDT) interface specification – Part 1: Overview and guidance*

IEC 62453-2:2022, *Field Device Tool (FDT) interface specification – Part 2: Concepts and detailed description*

3 Terms, definitions, abbreviated terms and conventions

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC 62453-1, IEC 62453-2 and the following apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

- IEC Electropedia: available at https://www.electropedia.org/
- ISO Online browsing platform: available at https://www.iso.org/obp

3.1.1
action
execution of a function which may involve several calls to interface methods of different FDT Objects

3.1.2
asynchronous methods
methods that trigger execution of asynchronous operations

Note 1 to entry: See also 5.8.7.

Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

3.1.3
asynchronous operation
operation that is performed while the FDT Object (client) that has requested the operation does not wait for the result, but the client is notified when the operation is finished

3.1.4
bulk data
device-node-specific persisted data, which is stored besides DTM instance data

Note 1 to entry: Example for bulk data: accumulated historical data, used for trend analysis.

3.1.5
bulk operation
operation to perform one or more tasks at a group of device nodes

Note 1 to entry: Examples for bulk operation: up- or download for a group of devices, parameter adjustment for a group of devices or report generation for a group of devices.

3.1.6
clone DTM instance
process of creating a new device node in the FDT topology based on an existing device node

Note 1 to entry: This includes copying DTM instance (see 3.1.10) and resetting device-node-specific DTM data.

Note 2 to entry: The identification attributes of the device are changed.

3.1.7
communication channel
component representing access to a fieldbus segment or to other means of communication

3.1.8
compatibility
feature of a component (hardware or software) that enables it to be interoperable with another component, or replace another version of the component

3.1.9
compatibility attributes
attributes used to find compatible components, to replace components or to validate compatibility after a component replacement

Note 1 to entry: Compatibility attributes are required to check whether a component is compatible with another component.

Note 2 to entry: Compatibility attributes are used to define compatibility in regard to meaning b) of compatibility.

3.1.10
DataTransaction
transaction regarding the data of a DTM (persistent or device data)

3.1.11
delegate
reference type that can be used to encapsulate a method

3.1.12
device configuration
process of changing data related to device-specific characteristics/basic behavior

Note 1 to entry: An example of such characteristics may be the structure of a remote I/O or the type of measurement procedure like absolute pressure or differential pressure

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

3.1.13
device data
configuration data that resides on the device

3.1.14
device parameterization
process of changing device-specific data in order to adjust application-specific behavior

3.1.15
device node
node in the topology, which represents a device

Note 1 to entry: A DtmDeviceType is assigned to a device node, which is instantiated to operate the device in online or offline modes. See Figure 25.

3.1.16
device type check
process of checking the device type when a DTM is going online with a connected physical device

Note 1 to entry: The DTM shall reject to go online if the connected physical device type is not supported. The check shall be based on same information as in DeviceIdentInfo. See 6.3.2.3.

3.1.17
DD-Interpreter DTMs
DTMs which interpret device descriptions at runtime

3.1.18
Device Type Manager
DTM
software component containing device-specific application software, including DTM Business Logic, DTM User Interface and related objects (e.g. Communication Channel)

Note 1 to entry: Older FDT specification documents used the term "DTM" for the object DTM Business Logic as well as for the whole component consisting of DTM BL, DTM UI and channels.

3.1.19
DTM Business Logic
DTM BL
part of the DTM, which contains all the functionality to access storage and communication and which manages the instance data of a DTM

3.1.20
DTM identifier
identifier, which is used to identify a DTM (DTM BL class)

Note 1 to entry: In order to uniquely identify a DTM BL class, the property DtmInfo.Id is used.

3.1.21
DTM instance modal
modal which prevents user interaction with other windows of the DTM instance

Note 1 to entry: See modal window.

3.1.22
DTM User Interface (DTM UI)
part of the DTM, which is displayed to a human user

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

3.1.23
DtmType
type of component supported by DTM

Note 1 to entry: DtmType can be either DtmDeviceType, DtmModuleType or DtmBlockType. DtmType is described by TypeInfo.

3.1.24
DtmDeviceType
element of a DTM software supporting one or more device types

Note 1 to entry: DtmDeviceType is described by DeviceTypeInfo class.

3.1.25
DtmModuleType
element of a DTM software supporting one or more device module types

Note 1 to entry: DtmModuleType is described by class ModuleTypeInfo.

3.1.26
DtmBlockType
element of a DTM software supporting one or more block types

Note 1 to entry: DtmBlockType is described by class BlockTypeInfo.

3.1.27
FDT Object
object defined by FDT (e.g. Frame Application, DTM Business Logic, DTM User Interface, Communication Channel)

3.1.28
FDT Protocol Annex
document defining support for a communication protocol for FDT

Note 1 to entry: Examples for such documents are "PROFIBUS protocol annex" and "HART protocol annex". Within IEC 62453 the series parts with numbers 3xy define support for communication protocols.

3.1.29
FDT Application Profile Annex
document defining support for a type of application for FDT

Note 1 to entry: An example for such a document is the "PLC Tool interface" (defined for FDT1.2). Other such documents may be defined at a later time also for IEC TS 62453-43.

3.1.30
Frame Application modal
modal which prevents user interaction with windows of the Frame Application

Note 1 to entry: See modal window.

3.1.31
hardware platform
hardware on which FDT software is executed

Note 1 to entry: Different hardware platforms may be supported that are based on different architectures and display formats, for example PC and others.

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

3.1.32
identification attributes
attributes which describe the identity of a component

Note 1 to entry: These attributes are typically displayed to users or used to validate and ensure compatibility of components.

3.1.33
incompatibility
situation where a component is not interoperable or where a component cannot replace another component

3.1.34
instance data
configuration data that resides in the DTM instance

3.1.35
lifetime of DTM instance
time span of executing a DTM BL (from state 'created' till state 'released')

3.1.36
link
logical relation of a DTM to a physical device (not the communication connection)

3.1.37
modal window
prevents user interaction with all windows of the process

3.1.38
online data
configuration data that resides on the device and can be accessed by communicating with the device

Note 1 to entry: Online data may be a subset of device data (i.e. it may be that not all device data is accessible by communicating with the device).

3.1.39
operation
procedure that may involve one or more method calls between FDT Objects

3.1.40
passive device
device that does not communicate

3.1.41
pattern
standard solution to common problems in software design

3.1.42
platform
combination of hardware platform, target platform and target CLR, that defines the environment for execution of FDT software

3.1.43
project
generic term for the sum of information related to a set of devices

Note 1 to entry: The definition of project is specific for a Frame Application.

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

3.1.44
proxy object
object which functions as a representative of another object

Note 1 to entry: The proxy pattern is often used as software design pattern.

Note 2 to entry: IEC TS 62453-43 uses the proxy pattern for interaction between DTM BL and DTM UI, between DTM BLs, between DTM BL and Communication Channels, for supporting multiple CLI implementations and versions in a Frame Application and for providing backward compatibility to FDT1.2.x [1], [2] and FDT2 [3].

3.1.45
reassign
change the TypeInfo assigned to a device node, either:

a) to the same TypeInfo.Id in a different DTM, or
b) to another TypeInfo.Id within the same DTM or to another DTM

3.1.46
reassignment
process of assigning a different DtmType to a device node with assigned DtmType

Note 1 to entry: It is possible that for the previously assigned DtmType already a dataset exists. This dataset should be considered in the reassignment.

3.1.47
replacing installation
installation of a new version of a DTM which replaces a currently installed DTM version

Note 1 to entry: A Frame Application is notified about the installation, but the DtmDeviceTypes do not need to be reassigned. DTM updates (see update) and DTM upgrades (see upgrade) replace installations of older versions of the DTMs.

3.1.48
revision
identification of modification of non-FDT components, e.g. device firmware or device hardware

Note 1 to entry: Not all fieldbus specifications supported by FDT and/or device types provide a version identification which allows to derive compatibility statements.

Note 2 to entry: In contrast to a version, revisions require additional fieldbus-specific or device-type-specific knowledge to derive compatibility or interchangeability predictions.

3.1.49
SemanticInfo
identifier that provides a reference to semantics defined in a specific context

Note 1 to entry: The reference is provided by the SemanticId, the context is provided by the ApplicationDomain that accompanies the SemanticId.

Note 2 to entry: There may be several semantics provided for an information item, e.g. a parameter may be described in a fieldbus profile as well as in a device profile (e.g. for drives), that is why several SemanticInfo entries may be provided for an information item.

3.1.50
set point
target value that an automatic control system will aim to reach

Note 1 to entry: For example a boiler control system may have a temperature set point, which is the temperature the control system aims to attain.

3.1.51
Sibling DTMs
DTM instances in an FDT Project, which are classified by their relation to the same Communication Channel

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

3.1.52
surrogate process
process hosting an object on behalf of client processes

Note 1 to entry: A surrogate process can have other qualities than the client process. E.g. it can be used to load different CLI implementations (e.g. .NET Core, .NET Framework) and versions.

3.1.53
synchronous operation
operation that is performed while the object that requested the operation is waiting for the result

3.1.54
target CLR
common language runtime, which defines the environment for execution of FDT software

Note 1 to entry: An example for target CLR is the CLR 4.0.

3.1.55
target platform
native data size supported by the machine and operation system on which the FDT software is executed

3.1.56
update
process to replace a component with a later (up to date) revision (update revision) that includes error corrections

3.1.57
update revision
(minor) revision of a component that includes error corrections and small enhancements

Note 1 to entry: In comparison to an upgrade revision an update revision includes no major functional enhancements or new features. An update revision shall be backwards compatible to previous revisions of the same component.

3.1.58
upgrade
process to replace a component with a later revision that includes functional enhancements and/or new features (upgrade revision)

3.1.59
upgrade revision
revision of a component that includes functional enhancements and/or new features compared to a previous revision of the component

Note 1 to entry: An upgrade revision shall be backwards compatible to previous revisions of the same component.

3.1.60
version
instance of a software product derived by modification or correction of a preceding software product instance (see [26])

Note 1 to entry: The format of a version is: Major.Minor[.release[.build]] for more information see Annex D.

Note 2 to entry: Version is used in IEC TS 62453-43 for identification of FDT software components and for corresponding compatibility attributes.

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

3.2 Abbreviated terms

For the purpose of this document, the abbreviated terms given in IEC 62453-1, IEC 62453-2 and the following apply.

API	Application Programming Interface
BL	Business Logic
BTM	Block Type Manager
CLI	Common Language Infrastructure
CLR	Common Language Runtime
CLS	Common Language Specification
CSS	Cascading Style Sheet
DCS	Distributed Control System
DD	Device Description
DLL	Dynamic Link Library
DTM	Device Type Manager
FA	Frame Application
FDT®[1]	Field Device Tool
GUI	Graphical User Interface
GUID	Globally Unique Identifier (a UUID)
HART®[2]	Highway Addressable Remote Transducer
HTML	Hyper Text Markup Language
IO	Input / Output
MIME	Multipurpose Internet Mail Extensions
MSDN®[3]	Microsoft®[4] Developer Network
OPC	Open Packaging Convention
OPC UA	Open Platform Communications Unified Architecture
PLC	Programmable Logic Controller
RID	Runtime id
UI	User Interface
WPF	Windows Presentation Foundation
XSL	eXtensible Stylesheet Language
XSLT	XSL Transformations

---

1 FDT® is the trade name of a product supplied by FDT Group AISBL. This information is given for the convenience of users of this document and does not constitute an endorsement by IEC of the product named. Equivalent products may be used if they can be shown to lead to the same results.

2 HART® is the trade name of a product supplied by Fieldcom Group. This information is given for the convenience of users of this document and does not constitute an endorsement by IEC of the product named. Equivalent products may be used if they can be shown to lead to the same results.

3 MSDN® is the trade name of a product supplied by Microsoft Corporation (Redmond, USA). This information is given for the convenience of users of this document and does not constitute an endorsement by IEC of the product named. Equivalent products may be used if they can be shown to lead to the same results.

4 Microsoft® is a trade name registered by Microsoft Corporation (Redmond, USA). This information is given for the convenience of users of this document and does not constitute an endorsement by IEC of the product named. Equivalent products may be used if they can be shown to lead to the same results.

AVA Documents Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

3.3 Conventions

The conventions for the UML notation used in this document are defined in IEC 62453-1:2016, Annex A.

This document specifies requirements to software. Different levels of requirement may be recognized by the used wording.

Wording	Indicates
'shall', 'has to', 'have to' ,or 'Mandatory'	No exceptions allowed.
'should' or 'Recommended'	Strong recommendation. It may make sense in special exceptional cases to differ from the described behavior.
'conditional'	Function or behavior shall be provided, depending on defined conditions.
'can' or 'Optional'	Function or behavior may be provided.

Further conventions are:

Convention	Indicates
Note:	Indicates text (in small letters), which does not express requirements, but provides additional information.
<MethodName>	Angle brackets are used to indicate a reference to an asynchronous method. NOTE If looking for definition in Annex B: Such methods are implemented as pair of BeginMethodName()/EndMethodName()

Code examples provided in this document are intended for illustration of the described concepts. They should not be used as is. Developers of FDT software should consider where the developed code is applied and design the software accordingly. For exact specification of protocol-specific implementations, refer to the FDT Protocol Annex documents.

4 Implementation concept

4.1 General

This clause defines the general concepts of IEC TS 62453-43. The object model, use cases and general solution approaches are explained.

IEC TS 62453-43 is an evolution of FDT, based on IEC TR 62453-42. For a comparison of IEC TS 62453-43 with IEC TR 62453-42 refer to Annex G.

4.2 Implementation of abstract FDT Object model

Figure 2 provides an overview of how the FDT Objects (defined in IEC 62453-2) are implemented in IEC TS 62453-43 and how their relationship to each other is implemented. The FDT Objects are implemented as .NET objects. WebUIs, as a specific type of user interface, are implemented based on HTML and JavaScript. See 5.2 and 5.3 for the supported HTML and JavaScript versions respectively. See Annex C for an explanation of how the services defined in IEC 62453-2 are implemented in IEC TS 62453-43.

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

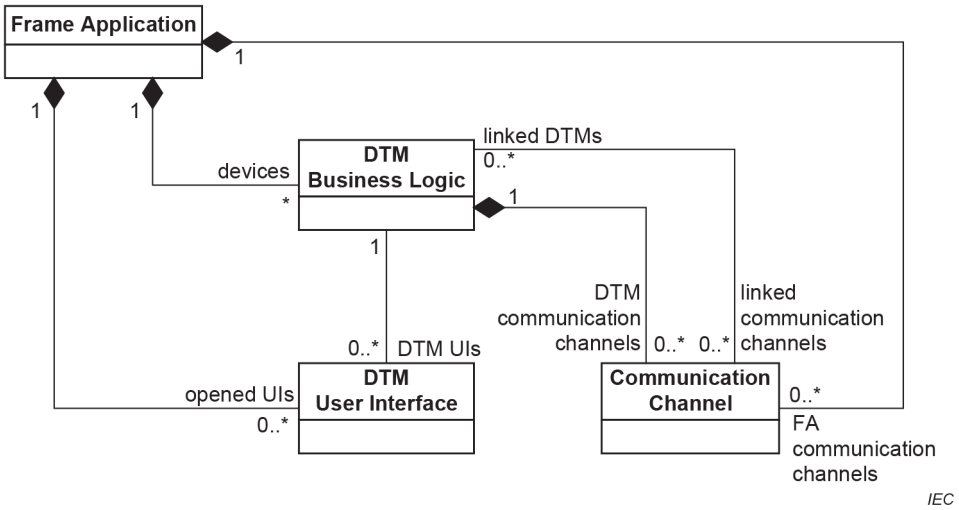


Figure 2 – IEC TS 62453-43 Object model

If the Frame Application is a distributed software system, the Frame Application is responsible to organize the instantiation of the objects (based on a vendor-specific implementation).

4.3 FDT Frame Application (FA)

4.3.1 General

Frame Applications are able to handle any type of device by integrating corresponding DTMs without the need for device or fieldbus-specific knowledge. Depending on their intended use, Frame Applications may provide different appearance and features (e.g. standalone configuration or engineering system of a DCS). Typically, the Frame Application comprises client applications focusing on specific aspects such as configuration, observation, IO planning, and using the services provided by the DTMs.

A Frame Application is the runtime environment for the DTMs and provides interfaces which enable the DTM Business Logic and the DTM User Interfaces to interact with its environment. In addition, the Frame Application manages the interaction between the DTM Business Logic and the DTM User Interface by providing a standard messaging interface (see Figure 3).

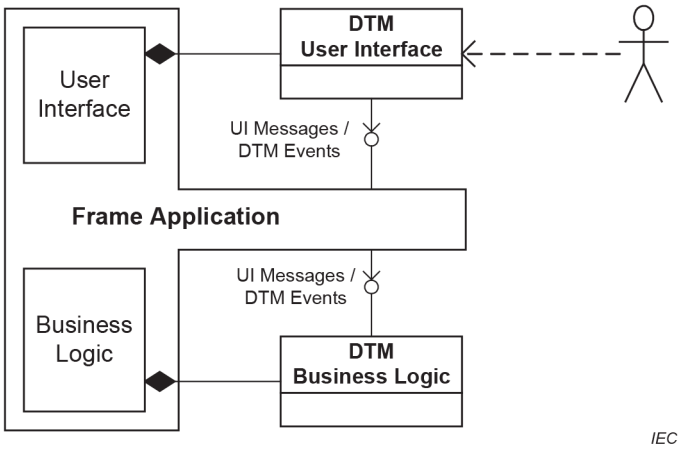


Figure 3 – Frame Application

The messaging interface is used for the transport of DTM-specific messages and events. Contents and format of the messages are proprietary and not understood by the Frame Application.

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

The Frame Application User Interface represents all functions of the frame related to user interface. The Frame Application Business Logic represents all frame functions related to business logic. Both are not specified by FDT, but are implementation-specific parts of the Frame Application (e.g. functional parts or structural parts). These two parts can be comprised in one single application or in separated applications, for example in a server and client application.

Frame Applications can have no, one or multiple Frame Application User Interfaces.

The Frame Application Business Logic part is responsible for executing the DTM Business Logic. It provides services which enable the DTM Business Logic to:

- persist data in the Frame Application persistence storage (see 4.14.1),
- communicate with associated device,
- request displaying of further user interfaces (e.g. user dialogs, additional DTM User Interface),
- browse the FDT topology and interact with other DTMs,
- inform the Frame Application regarding events (error / trace messages, progress etc.),
- interact with the DTM User Interface.

The Frame Application user interface part makes the DTM services available to the users, for example the functions and user interfaces supported by a DTM (see 4.5). It hosts the DTM User Interfaces as part of its own user interface and provides services to:

- interact with the DTM Business Logic (see 4.4)
- request displaying of further user interfaces (e.g. user dialogs, additional DTM User Interface)
- browse the FDT topology and interact with other DTMs
- inform the Frame Application regarding events (error / trace messages, progress etc.)

4.3.2 System communication

A Frame Application may have built-in communication capabilities, provided through Communication Channels. Frame Applications without built-in communication capabilities use Communication DTMs (see 4.7.3). Figure 4 shows the relation between the Frame Application and Communication Channel object.

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

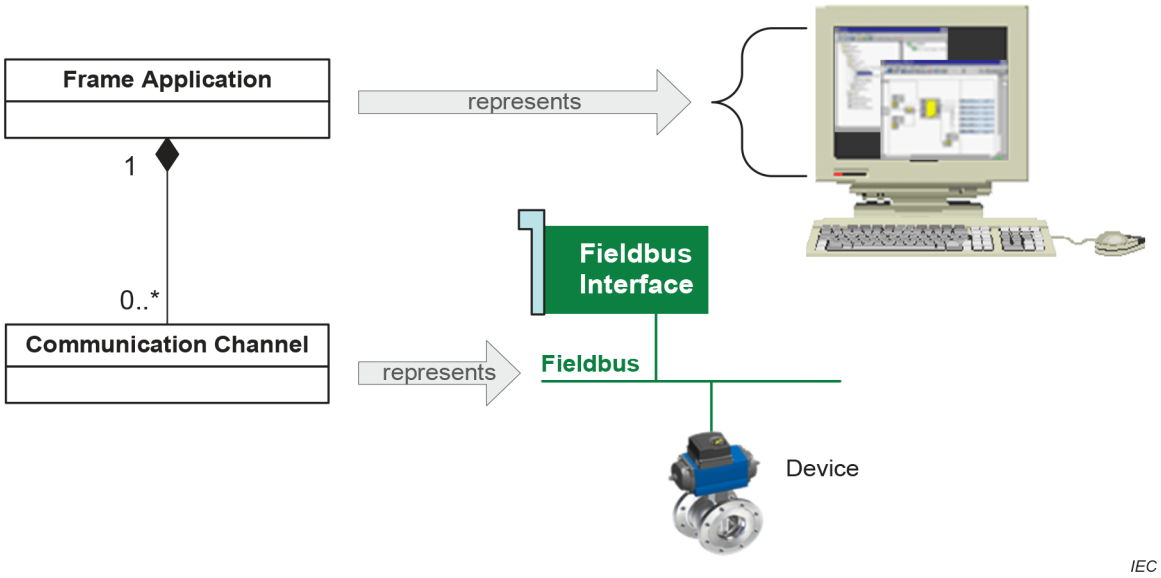


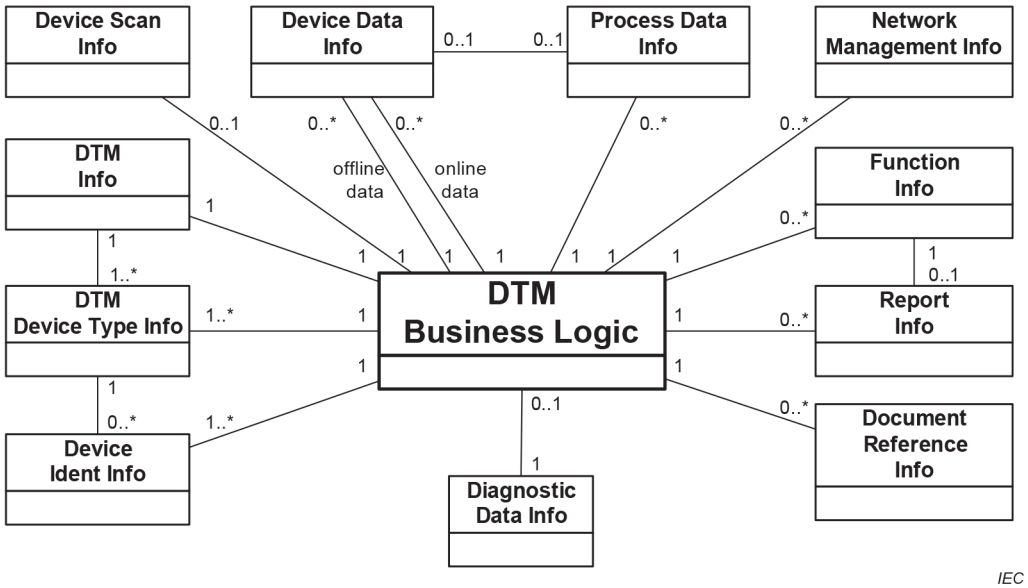
Figure 4 – Frame Application with integrated Communication Channel

4.4 DTM Business Logic

4.4.1 General

The DTM Business Logic (DTM BL) encapsulates device-specific functions and protocol-specific definitions. Thus a Frame Application is able to handle any type of device by integrating corresponding DTM Business Logic without the need for device-specific or fieldbus-specific knowledge.

The Frame Application interacts with the DTM Business Logic through defined interfaces (see Annex B). Figure 5 shows the information objects which are used by the interface definitions implemented by the DTM Business Logic.



NOTE Since this section describes the general concept of FDT, the actual implementation in a DTM may differ. (E.g. if a network protocol uses network management, it will be mandatory to provide network management information.)

Figure 5 – DTM Business Logic

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

4.4.2 DTM, DTM Device Type, and Device Ident Info

A DTM supports interfaces to provide information about itself and the devices which are supported by the DTM. This information can be used by the Frame Application to create libraries, select a DTM during creation of the topology, or for simple validations.

In order to increase performance in creation of libraries and selection of DTMs, the information shall be provided by a separate class (the so called DtmInfoBuilder class). The DtmInfoBuilder is installed together with the DTM. It implements the same interface to provide information as defined for a DTM (see IDtmInformation in Annex B). The main advantage of DtmInfoBuilder is, that it can be used without instantiating the DTM. Since this subclause describes the general concept of FDT, the actual implementation in a DTM mightdiffer. (E.g. Since the scan method is important for recognizing devices, Device DTMs shall provide Device Ident Info.)

NOTE By using the DtmInfoBuilder it is possible to adapt the available TypeInfos depending on various conditions (e.g. DD-Interpreter DTMs mightprovide TypeInfos depending on installed DD files).

Figure 6 shows the information datatypes that are provided by a DTM in order to support these Frame Application function and how this info datatypes are used to describe the supported devices (see 7.4 and 7.5 for detailed description).

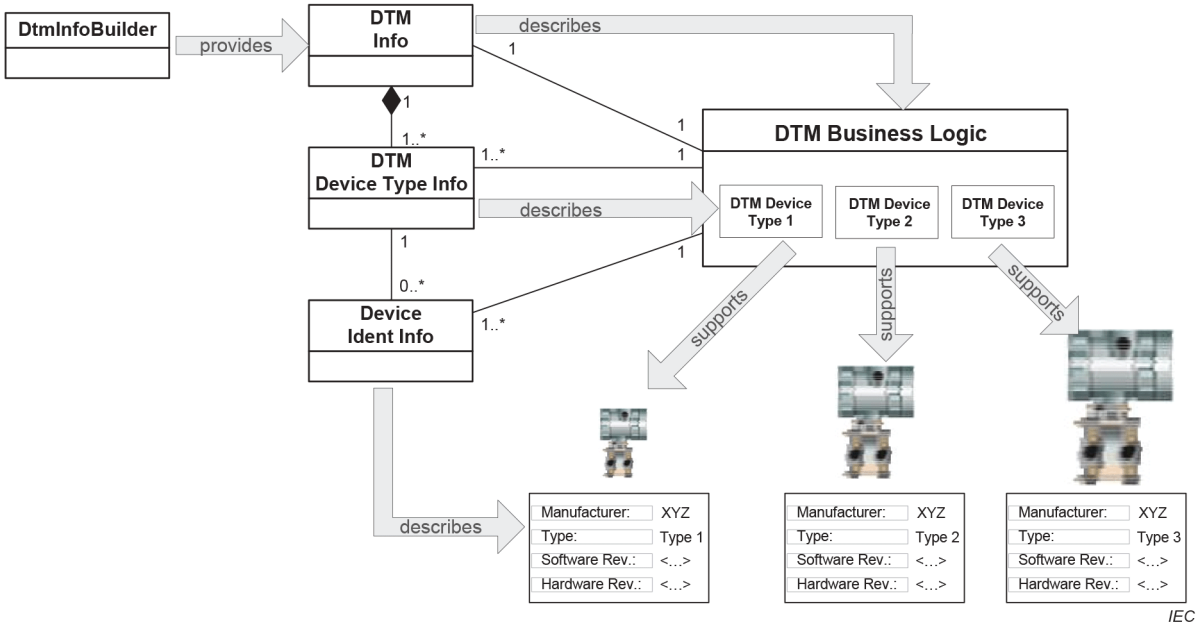


Figure 6 – DTM, Device Type and Device Ident Info

The representation for a particular physical device type within the DTM is called DTM Device Type. A DTM may provide one or more DTM Device Types. The concrete design and implementation of the DTM Device Types is not in scope of FDT.

Information about physical device types, which can be handled by the DTM Device Types is returned by the service IDtmInformation.GetDeviceIdentInfo (see definition of IDtmInformation in Annex B). Such information is for example manufacturer, type, hardware and embedded software version of the device. The DTM may even return regular expressions for some specific device identification elements to signal that the DTM Device Type can be used for all devices for which the expression matches (e.g. the term '.*' for the hardware version may signal that the DTM Device Type supports all hardware versions).

The information returned by service IDtmInformation.GetDeviceIdentInfo is fieldbus-specific and therefore defined by the document describing the protocol profile integration in IEC TS 62453-43. However, FDT defines the means to transform the information into a protocol-independent format to enable Frame Applications without protocol-specific knowledge to use it.

AVA Documents Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

4.4.3 Device data info

A DTM supports interfaces to read and write device parameters stored in the DTM instance data (offline data) and directly in the connected device (online data). This data is represented by DataInfo (see 4.15.1 for detailed description).

A DTM shall expose a defined set of device parameters which are publicly available (see 4.15 for detailed description). Parameters are provided in a bus neutral structure allowing their use without knowledge of the fieldbus protocol.

4.4.4 Process data info

Process data provided by devices (e.g. IO signals) are integrated into the functional planning of the control system. The process data related information for the integration of the device into the control system like datatype, signal direction, engineering units, and ranges is provided by the DTM Business Logic for each DTM Device Type (Figure 7), but may also depend on the device instance configuration.

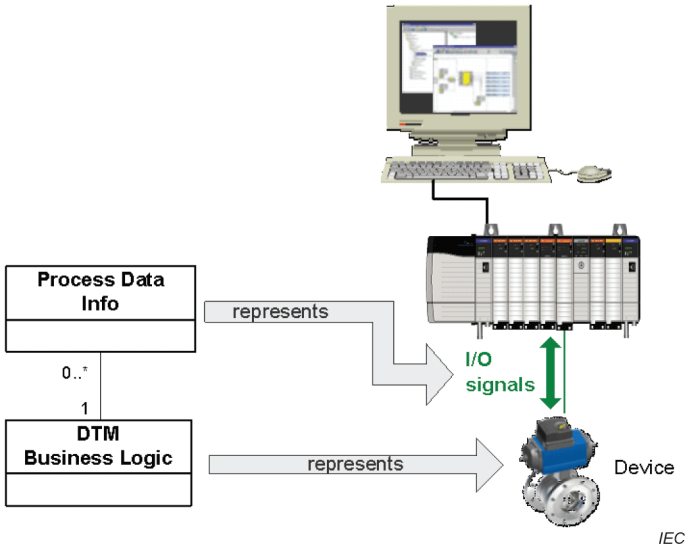


Figure 7 – Process data info

The process values provided by a device, both the number and type, may depend on the configuration of the DTM. Thus the number of available Process Data Info objects may also depend on the device configuration. A Frame Application is able to inform the DTM that further configuration changes shall be prohibited because the process data is already integrated into functional planning of the control system. In this case the DTM shall not allow any changes which will affect the definition of the available process values.

The process information is protocol specific. Each FDT Protocol Annex defines a derived class defining which information shall be contained. However, the properties in the base class Process Data Info provide common information (e.g. IO signal name, tag etc.) in a protocol-independent format to enable Frame Applications without protocol-specific knowledge to integrate the process variables into the system (see 7.11.1 for detailed description).

If the process data value is also available as device data Info, then the corresponding element is referenced by the Process Data Info element (see IOSignalRefs in 7.11.1).

A Process Data Info element may have a relation to a Communication Channel. This is a typical case for a Gateway DTM for a remote IO (see 4.7.4). The relation is represented within the Process Data Info element (see IOSignalRef in 7.11.1).

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

NOTE The Process Data Info replaces Process Channel as defined in FDT1.2.x.

4.4.5 Diagnostic data info

A DTM provides a service to retrieve the status of the related device. The status is encoded in a protocol independent way, according to [8][5], [6].

4.4.6 Network management info

The DTM supports an interface to read and write network management information which can for example be used for address management and bus master configuration (see 8.11.1).

The Network Management Info is protocol specific. It may contain device bus-address, tag and additional protocol-specific configuration settings. Each FDT Protocol Annex defines a derived class defining the protocol-specific record. However, the base class Network Management Info provides common information (e.g. bus-address, tag) in a protocol-independent format to enable Frame Applications without protocol-specific knowledge to use it (see 7.13 for detailed description).

4.4.7 Function info

Each DTM may provide a set of functions comprised by

– DTM User Interfaces,

– DTM functions without DTM User Interface (CommandFunctions) and

– references for external documents.

The Function Info object provides the information about the functions such as name, status (enabled / disabled), etc. (see 7.14 for more detailed description).

A function should provide printable information. A Frame Application may call the documentation interface (see IEC 62453-2:2022, 7.2.7.1) of the DTM to retrieve printable information. This interface returns a corresponding Report Info object which holds the printable information. The Report Info object may indicate the relation to the function.

4.4.8 Report info

If a DTM provides instance data and/or online data, then the DTM shall implement a device-type-specific reporting of the provided data for documentation and archiving purposes. The DTM BL may implement different types of reports that each cover a distinct subset of the instance or online data of a device.

The Report Info exposes a list of report types supported by a DTM (see 7.18). The list may be grouped. The list is static over the lifetime of a DTM instance, there are no dependencies on the current application context.

4.4.9 Document reference info

A DTM may provide references to external documents, which are displayed by the Frame Application. These references may be provided as part of the TypeInfo (as static information) or as part of the function Info (may change dynamically). The Frame Application may provide an own viewer for the documents or rely on the condition that external software is installed for the document type. It is recommended, that DTMs use common document formats (recommended formats are PDF and HTML).

---

5 Numbers in square brackets refer to the Bibliography.

Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

Document references may be provided as local path or as URL. Frame Applications may apply restrictions to displaying documents from URLs for security reasons.

4.5 DTM functions

4.5.1 DTM User Interface

A DTM User Interface (DTM UI) shall be a web-based user interface (DTM WebUI).

The contents and layout of the DTM WebUIs is device specific, but shall follow the DTM Style Guide [4].

4.5.2 Function access control

The Frame Application can restrict the invocation of functions provided by a DTM and restrict DTM transition to certain states (e.g. restrict transition to state Online by not calling the method EnableCommunication2()).

The Frame Application will get the list of the functions provided by the DTM in IFunction.FunctionInfo property. If possible the DTM should expose all functions available in the DTM in all states (see DTM states in 6.3.2). The FunctionItem.Enabled property will indicate if the functions are not applicable for the current DTM state. It is mandatory for a DTM to expose all functions that are available in the current configuration (e.g. the current user level). A DTM may change the list of functions, if the configuration changes.

If the DTM instance was terminated and loaded again, then a DTM shall expose the same set of functions for a device node. If the DTM is updated the list of the functions may change, the Frame Application may detect an update of a DTM (see 10.4) and shall check if the list of functions changed.

The number and type of listed functions may change, as well as the status of the functions in regard to availability and visibility may change.

The DTM shall indicate any changes by IFunction.FunctionsChanged event. The Frame Application shall update the list of functions accordingly. Frame Application should hide the functions with Hidden property set to TRUE.

When a user interface for the DTM is invoked, the DTM shall not allow switching the context from within the DTM WebUI.

For Example:
If a Diagnostic Function is invoked by the Frame Application, the DTM will present the diagnostic information in the user interface. The user should not be allowed to invoke the configuration screen from within the DTM WebUI for the diagnostic function without the permission from the frame. The user shall be able to invoke the configuration screen from the functions exposed to the Frame Application if the permissions in the Frame Application allow it.

If the MainOperation Function is invoked by the Frame Application, the DTM will present all available functions within the user interface. The user may invoke the Configuration screen from within the DTM WebUI as well as the Diagnostic function or any other integrated function.

4.5.3 User interaction in DTM WebUIs

If a user changes values in DTM WebUIs, those changes are applied after confirmation to the respective DTM or device (see [4]).

Status information for a DTM WebUI (data source, summary parameter modification state, UI operation mode, activation of service mode) can be displayed (see [4]).

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

NOTE This behavior is modelled according to the general user experience in web-based GUIs. It is based on the expectation that a web user interface might lose connection to its data source at any time.

4.5.4 Command functions

Command functions are used to execute actions (commands) either on the DTM BL. Command functions in context of the DTM BL shall not have a GUI.

A Command function may have parameters. The information about the parameters, which is provided by the DTM BL, may include default values of the parameters. The actual parameter values are passed when the Command function is executed.

One example for command functions is: Reset device.

4.5.5 Static Function

Static Function is a concept that allows processing information from a device without executing the DTM.

For observation and monitoring of a plant often it is necessary to retrieve not much data from many devices. In such use cases the effort to start and manage many DTMs may be too high. On the other hand in such use cases it may be not sufficient to read a single item from the device. Sometimes the data read from the device needs to be processed or to be combined with additional information in order to calculate the correct information.

A Static Function is part of a DTM package, the DTM provides information regarding the Static Function. A Static Function has no access and no relation to the data of a DTM.

StaticFunctions are executed using a StaticFunction Provider object. The Frame Application is responsible for creating the Static Function Provider objects. There shall be only one Static Function Provider object per device and it is initialized with the necessary communication data. On one StaticFunction Provider object the execution of a Static Function may be started multiple times in parallel. It is the responsibility of the StaticFunction Provider to process the functions sequentially in the order they were started.

It is recommended that a DTM provides Static Functions to support well defined use cases as defined by FDT protocol project groups and Profile Working groups.

Examples of Static Functions are:

– GetDeviceStatus
  Provides the current device status in FDT format (see Fdt.Dtm.DeviceStatus)
– GetProcessValue
  Provides the current process value of the device (input or output) in normalized format
– GetOperationTime
  Provides the operation time of a device in .NET Time format.

See Annex J for definitions of standard Static Functions, which provide support for common use cases.

4.6 Communication Channel

A Communication Channel provides access to the fieldbus and is communication technology dependent. It represents the entry point to a fieldbus network, a vendor-specific backplane bus or point-to-point connection. A Communication Channel may also represent logical communication to blocks within a device. A Communication Channel may be provided by a Frame Application (see 4.3.2) or by a DTM Business Logic (see 4.7.3).

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

A Communication Channel provides interfaces that are independent from the communication hardware and the fieldbus protocol. The interfaces provide methods to

- scan the fieldbus for available devices,
- connect to a device,
- send communication messages and
- disconnect from a device.

The interface methods take base arguments which are extended for protocol-specific communication by FDT Protocol Annex specifications (see Figure 8).

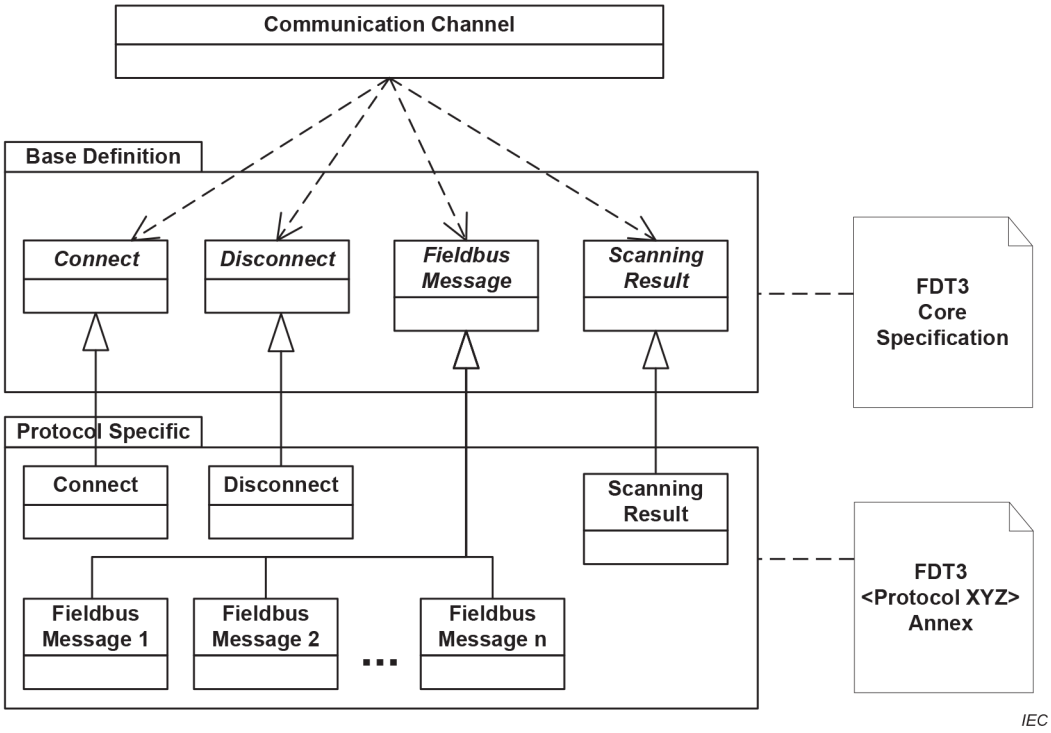


Figure 8 – Communication Channel

FDT also allows manufacturers to define support for proprietary bus protocols, which are not part of the standard; for example for a manufacturer-specific fieldbus or point-to-point communication

The communication services provided by the Communication Channel interface may be used by the Frame Application or a DTM to exchange data with a connected device or to initiate a function (e.g. identification request, device reset, broadcast, etc.). The general communication concept is further described in IEC 62453-2.

A Communication Channel shall be able to support multiple connections at the same time if the communication protocol supports this. This means supporting several DTMs creating connections to different devices as well as several DTMs creating connections to the same device. Depending on the supported protocol it may also be possible that one DTM creates several connections to one device.

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

4.7 DTM categories

4.7.1 General

DTMs represent measurement and control devices, modules, gateways or blocks as well as communication interfaces. The DTMs representing these devices only differ in the availability of provided data, functions, and user interfaces. The different DTM categories reflect the physical entities, which are represented by a DTM. The defined DTM categories cannot describe rules for all existing physical devices, but are intended to provide a general guideline for common categories of physical devices.

4.7.2 Device DTM

A Device DTM contains the application software for a standard field device, e.g. a pressure transmitter, a valve, or a drive. Such a DTM for example supports functions to configure and parameterize the device (see Figure 9).

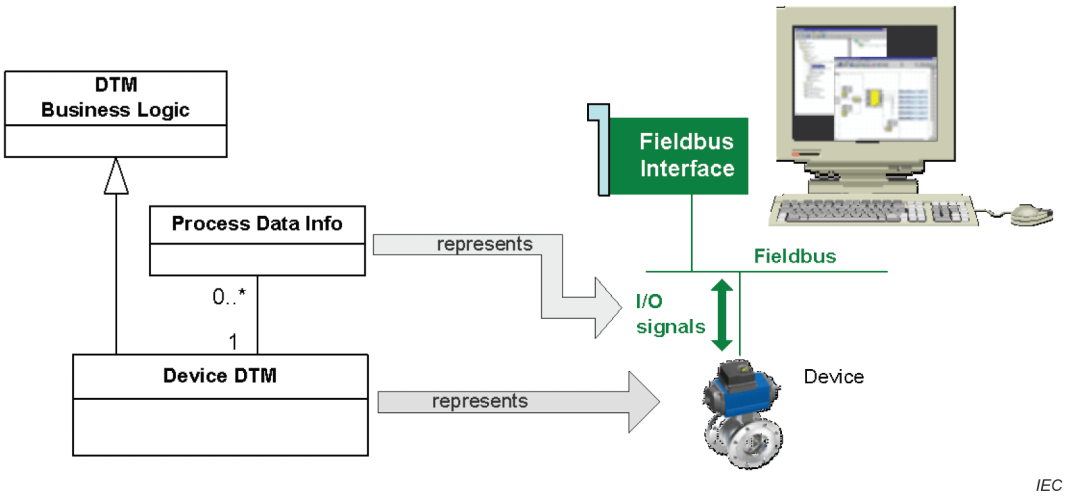


Figure 9 – Device DTM

The Device DTM shall provide ProcessDataInfo objects (see 7.11.1), if the device provides process data.

4.7.3 Communication DTM (CommDTM)

A Communication DTM is a special category of DTM containing the application software for specific communication hardware. For example, it may support configuration settings like baud rate, start and stop bits, etc.

A Communication DTM shall provide Communication Channels (see 4.6) which provide the services to access the fieldbus (see Figure 10). The number of provided channels may depend on the configuration of the Communication DTM (i.e. it might occur that a DTM temporary provides no channel, for instance during reconfiguration).

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

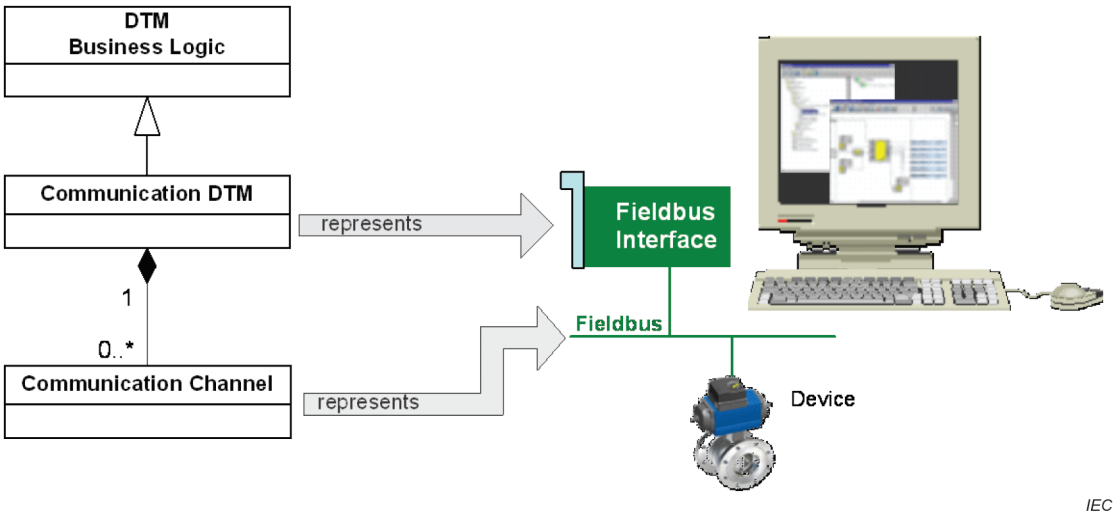


Figure 10 – Communication DTM

The advantage of Communication Channels provided by a Communication DTM compared to the Communication Channels provided by a Frame Application is, that they may be utilized by different Frame Applications. Each Frame Application thus may extend the number of protocols that the system is able to access. Both ways of providing Communication Channels may coexist in a Frame Application.

4.7.4 Gateway DTM

A Gateway DTM is a special category of DTM representing a device that connects different fieldbus segments, for example PROFIBUS DP with HART® (see Figure 11).

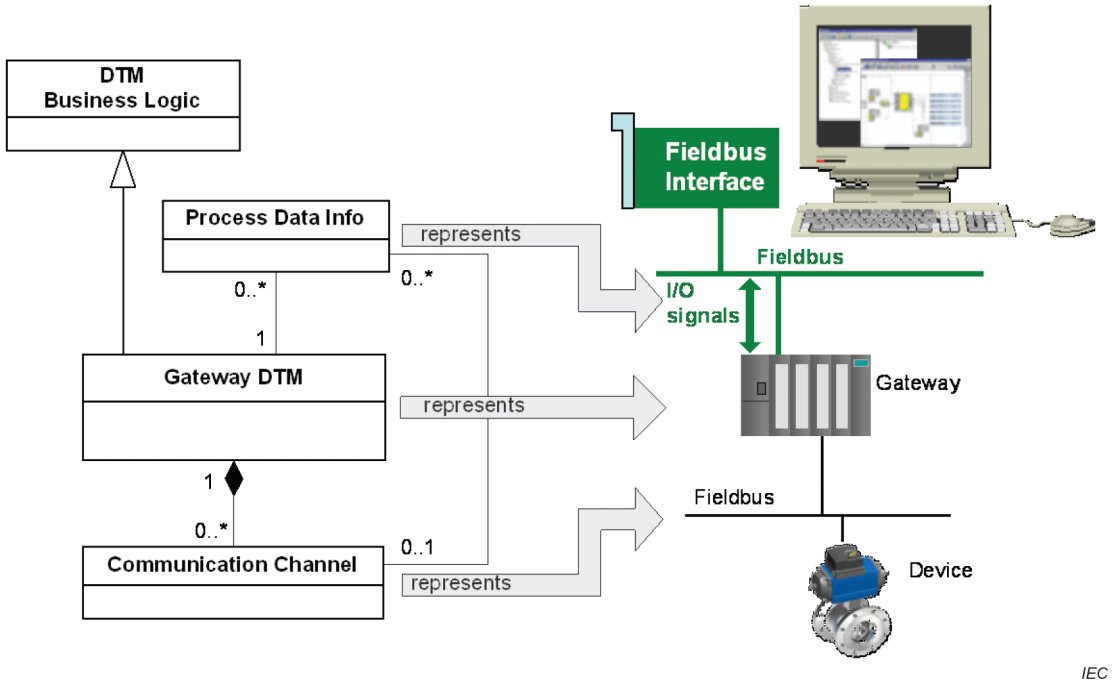


Figure 11 – Gateway DTM

Such a DTM can be seen as a combination of a Communication DTM and a Device DTM. It shall provide Communication Channels (see 4.6) to provide access to the linked fieldbus and ProcessDataInfo objects (see 7.11.1) for the fieldbus where it's connected to, if it is required to make the gateway I/O signals or process values accessible to the host system.

AVA Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

There may be a relation between the IO signals provided by a gateway device and the communication interface of that device (e.g. the gateway provides IO signals that are received from a device connected to the communication interface). In such a case the relation is represented as relation between Process Data Info and Communication Channel.

A Child DTM is connected to the Gateway DTM via the Communication Channel of the Gateway DTM. The Frame Application is responsible to create and manage the link between the DTMs. The Device DTM is able to communicate with its device via the services provided by the Communication Channel. This concept is called "nested communication" and is further described in 4.11.3.

4.7.5 Composite Device DTM

Some devices are composed from modules and sub modules. Hardware modules are plugged into physical slots of the composite device (modular device). For example, an I/O module can be plugged into a Remote I/O device. The communication between the composite device and the modules is provided usually by a backplane bus. The function of the backplane bus is defined by the manufacturer and often is based on private protocol definitions. Such a composite device may be represented by a hierarchy of DTMs as shown in Figure 12.

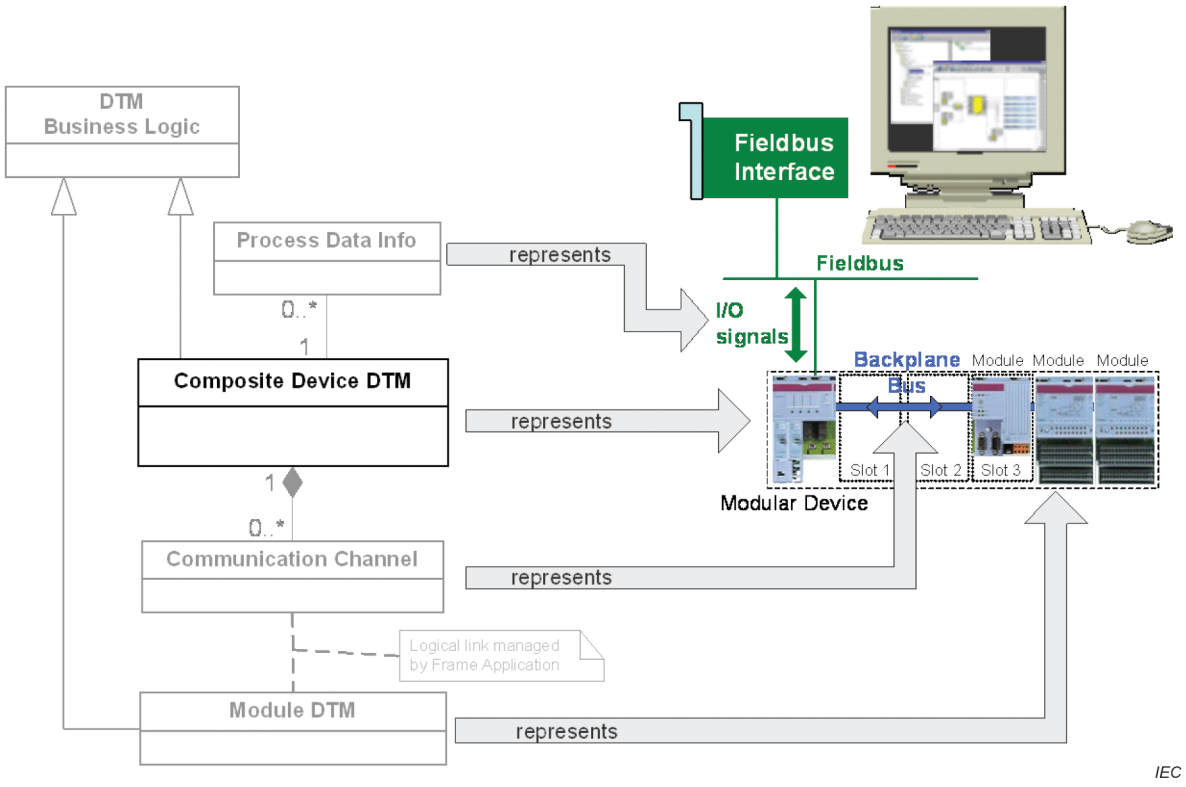


Figure 12 – Composite Device DTM

A Composite Device DTM is the root element for the complete device representation. This DTM shall provide Communication Channels (see 4.6) which represent the backplane bus. Depending on the software design of the Composite Device DTM one channel may be provided for representation of the whole backplane bus or multiple channels may be provided for representation of the slots where the modules are plugged in.

If the composite device provides process data, then the Composite Device DTM shall provide ProcessDataInfo objects (see 7.11.1). If the composite device forwards process data that is generated in the device modules, then the ProcessDataInfo provided by a Composite Device DTM may be composed from ProcessDataInfo that is provided by the Module DTMs.

AVA Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

4.7.6 Module DTM

A Module DTM is a special category of DTM containing the application software for a device hardware module (see Figure 13).

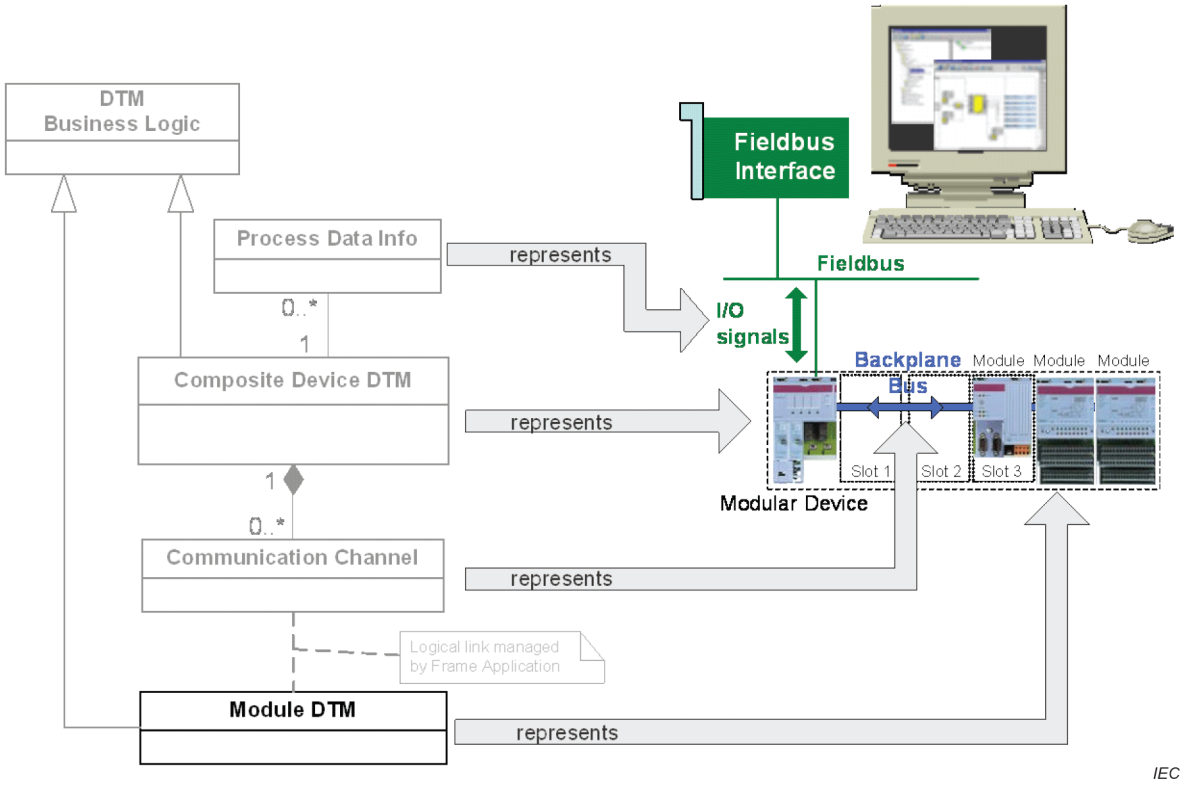


Figure 13 – Module DTM

The device modules are represented by Module DTMs. The Composite Device DTM and Module DTMs are generally shipped together and supplied by the vendor of the modular device. The relation between Composite Device DTM and related Module DTMs (and their respective DtmTypes) is defined by a specific ProtocolId for the backplane bus protocol.

A Module DTM is connected to the Composite Device DTM via the Communication Channel of the Composite Device DTM. The Frame Application is responsible to create and manage the link between the DTMs. The Module DTM is able to communicate with its module via the services provided by the Communication Channel. This means that the concept 'Nested Communication' (see 4.11.3) is also applied for Composite Device DTMs and the related Module DTMs.

If the device modules provide process data, then the respective Module DTMs should provide ProcessDataInfo objects.

If the device modules provide access to subordinate communication (e.g. in a modular gateway), then the respective Module DTMs should provide Communication Channels.

4.7.7 Block Type Manager

A Block Type Manager (BTM) is a special category of DTM containing the application software for accessible software objects inside a device. Often these software objects are defined as blocks, such as resource blocks, transducer blocks, and function blocks (see Figure 14).

AVA Documents
Document Sharing Hub

Full document details: PD IEC TS 62453-43:2024

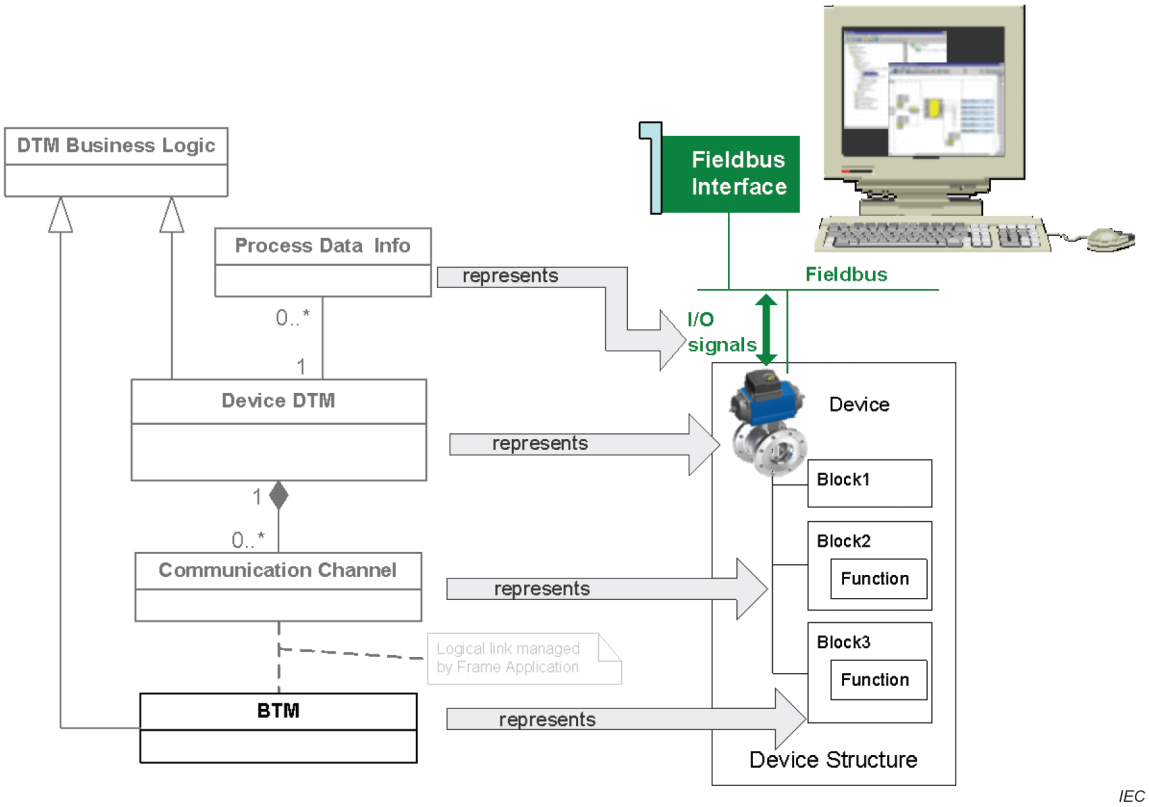


Figure 14 – Block Type Manager

A Device DTM is the root element for complete device representation. This Device DTM shall provide Communication Channels (see 4.6) which provide services to access the block information within the device.

A device may host standard blocks, vendor-specific blocks, and device-specific blocks. All types of blocks are represented by BTMs. The blocks are more flexible than device modules (see 4.7.6), for instance blocks may be moved from one device to another device (standard blocks may be even be moved between devices of different type). The Device DTM and the BTMs for device-specific blocks in general are supplied by the device vendor. BTMs for standard blocks may be supplied separately and may be used with all DTMs for devices that are able to host the respective blocks.

A BTM is connected to the hosting Device DTM via its Communication Channel, indicating that the respective blocks are hosted in the physical device. The BTM is able to interact with its block in the device via the services provided by the Communication Channel of the Device DTM. The Communication Channel represents an internal communication on the device and is used mainly to manage the link between BTM and block. The protocols supported by the CommunicationChannel are used to manage which BTMs may connect to a DTM (standard protocols for standard blocks, specific protocols for specific blocks). Additional validation rules may be applied by the Communication Channel. The Frame Application is responsible to create and manage the link between the Device DTM and the BTMs.

The BTM concept is independent from fieldbus protocol. However, a protocol may require modeling of device structures as blocks. Thus, whether and how the BTM concept is used to model device structures is defined by the FDT Protocol Annex.

If the blocks provide process data, the respective BTMs may provide ProcessDataInfo objects.

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

4.8 User management

4.8.1 General

FDT does not define a standard system for user management. The user management is part of product-specific definitions and may be implemented differently for different Frame Applications. Still it is necessary to define a common handling for access permissions, access rules and how components from different vendors communicate information regarding access permissions.

4.8.2 Multi-user access

Some Frame Applications provide multi-user capability. Such a system provides access for multiple users at the same time and may be distributed over several computers. This specification considers the distributed environment as one Frame Application. The Frame Application and the DTM are equally responsible to provide the multi-user access and to ensure consistency of data.

If, within one Frame Application, multiple users access the same device or the same device dataset, the Frame Application shall start a separate DTM instance for each user. All these DTM instances shall have same DTM type, shall be instantiated for the same DtmDeviceType and for the same physical device. Each DTM instance manages a separate instance dataset. These instance datasets are synchronized by means of the persistent dataset (see 4.14.5).

NOTE 1 An example architecture for multi-user scenarios may be found in 11.4.

NOTE 2 These rules also apply to Communication DTMs. If a Communication DTM provides access to a resource, which supports only limited access (e.g. a COM port), the Communication DTM might have to provide special measures in order to support multi-user access.

4.8.3 User levels

4.8.3.1 General

DTMs may be integrated in different Frame Applications, which may have varying requirements to restrict visibility and accessibility of device and persistent data, for example for plant safety reasons or to present a customized view to the user. The grade of restriction varies with the types of users supported by a system. Examples for users requiring data access restrictions are:

- a user assigned to observe a plant shall not have access to calibration-specific device parameters and consequently shall not see related DTM functionality,
- a device commissioning specialist needs to have access to calibration data and functions,
- a user assigned to operate a plant shall be able to change (write) set point values and be offered appropriate functionality while a user assigned to observe a plant is not allowed to execute such changes.

4.8.3.2 Access Control Concept

FDT uses a role based access control concept. A Frame Application initializes a DTM Business Logic with the FDT-specific user level (see definition of IDtm3.Init() in Annex B). The user level is immutable over the lifetime of the Business Logic/User Interface instances.

In terms of access control, every actor in an FDT system may have one of the following three user levels at the interface of a DTM (Table 1):

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

Table 1 – FDT user levels

User level name	Description
Observer	This user level stands for an actor that observes the current process only.
Expert	This user level stands for an actor who has to execute specific use cases, e.g. operation use cases (operation expert) or device maintenance use cases (maintenance expert). This user level allows the Frame Application to configure access and privileges.
Engineer	This user level stands for an actor that has to do the plant planning, device configuration/parameterization and plant maintenance.

The user levels allow a stepwise extension of permissions. The Observer typically has a minimum permission set, the Expert has an intermediate permission set (which is configured by the Frame Application) and the Engineer has a full permission set.

NOTE For an explanation of the fundamental user levels and use cases that were considered for the design of the FDT specification see Annex A. A Frame Application might support only a subset of these use cases or additional use cases not defined in the annex.

According to the role set by the Frame Application, the DTM Business Logic and User Interface shall control access to device and persistent data (see definition of interfaces IInstanceData and IDeviceData in Annex B) as well as adapt its user interface appearance. This includes to hide some data or display it as read-only, but also to partially disable DTM-specific functionality (see definition of IFunction.FunctionInfo in Annex B), if it requires data access rights that are not associated with the specified user level.

It is mandatory for a DTM to implement a safe and read-only usage for the "Observer" user level. It is also mandatory for a DTM to implement unlimited usage for the "Engineer" user level. It is optional for a DTM to implement configurable custom usage for "Expert" user level. If "Expert" level is not implemented by a DTM but is set by the Frame Application, the DTM shall use the behavior for "Observer" user level.

Table 2 gives an overview about the user interfaces and functions that are expected to be available for the individual user levels. The data access rights should be defined to allow for the execution of these use cases.

The assignment of roles to individual users is Frame Application specific. The Frame Application may implement an own user management sub-system or use fixed user levels.

Table 2 – Role-dependent access rights and user interfaces for DTMs

<table>
<tr><th rowspan="2">Use case</th><th rowspan="2">Sub cases</th><th colspan="3">User level</th></tr>
<tr><th>Engineer(M)</th><th>Expert(O)</th><th>Observer(M)</th></tr>
<tr><td rowspan="3">System planning</td><td>Network management</td><td rowspan="9">If a DTM implements these use case, it shall expose all related commands and user interfaces.</td><td rowspan="9">If the DTM implements these use cases and supports the user role "Expert", it shall allow Frame Application to configure access to the exposed data, commands and user interfaces</td><td>-*)</td></tr>
<tr><td>Busmaster configuration</td><td>-</td></tr>
<tr><td>Channel assignment</td><td>-</td></tr>
<tr><td rowspan="2">System generation</td><td>Network management</td><td>O {r}</td></tr>
<tr><td>DTM matching</td><td>O {r}</td></tr>
<tr><td>Device configuration</td><td>-</td><td>-</td></tr>
<tr><td>Simulation (force)</td><td>-</td><td>-</td></tr>
<tr><td rowspan="2">Offline operation</td><td>Offline parameterization</td><td>-</td></tr>
<tr><td>Persistent data comparison</td><td>-</td></tr>
</table>

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024

Use case	Sub cases	User level		
		Engineer(M)	Expert(O)	Observer(M)
Online operation	Online functions (reset + other functionality that requires online device connection)	If a DTM implements these use cases, it shall expose all related commands and user interfaces.	If the DTM implements these use cases and supports the user role "Expert", it shall allow Frame Application to configure access to the exposed data, commands and user interfaces	-
	Online parameterization			-
	Calibration			-
	Device/Persistent data comparison			-
	Adjust SetValues			O {r}
	Upload			-
	Download			-
Bulk operation	Upload			-
	Download			-
Online view	Network scan			O {r}
	Online status			M {r}
	Online trend			M {r}
	Device identification			M {r}
	Online view parameter set			O {r}
Report generation	-			M {r}
Device-specific operations	Device-vendor-specific (or extended) DTM functions after DTM/Device-specific OEM Service login			-

M Mandatory (if a DTM implements this use case, it shall expose all related commands and user interfaces in the specified user level)

O Optional (a DTM may expose the related commands and user interfaces in the specified user level)

r User level shall have read access to all data related to the use case

- Use case not supported in this user level (DTM shall not expose any related commands/user interfaces)

*) A DTM shall allow all user levels to set the device address in the DTM with the method SetAddressInfo().

4.8.3.3 Frame Application configured access control

It is very difficult and even impossible for the DTM vendor to provide correct access control settings for all occasions. The data, which can be accessed, and the functions, which can be used, are changed by the user; depending on where the DTM is used or what is the operational phase of the plant. Here are some examples:

a) The same user may have different permissions for the same device controlled by the same DTM when the device is connected in the plant or in the instrument shop. In the instrument shop, the user may have all equipment to calibrate the instrument and the corresponding privileges should be granted. Little or no changes may be allowed when the instrument is connected later to the actual running control system.

b) The same user may have full control when the plant is being engineered, but the changes to the device may be significantly restricted, when the plant is in running state.

c) The same group of users may have different permissions for different instruments – some of the maintenance personnel may be trained to work with transmitters, other may be specialized in valve maintenance.

d) In a small application one person may be responsible for the entire application and he may have unlimited access to all device maintenance procedures, but in a big application, often the access is controlled according to the individual experience of the technical staff.

Documents
Document Sharing Hub
Full document details: PD IEC TS 62453-43:2024