

DIN EN 62453-2

ICS 35.240.50; 35.160

Einsprüche bis 2015-01-28
Vorgesehen als Ersatz für
DIN EN 62453-2:2010-02**Entwurf****Field Device Tool (FDT)-Schnittstellenspezifikation –
Teil 2: Konzept und grundlegende Beschreibung
(IEC 65E/334/CDV:2013);
Englische Fassung FprEN 62453-2:2013**Field Device Tool (FDT) Interface Specification –
Part 2: Concepts and detailed Description
(IEC 65E/334/CDV:2013);
English version FprEN 62453-2:2013Spécification des interfaces des outils des dispositifs de terrain (FDT) –
Partie 2: Concepts et description détaillée
(CEI 65E/334/CDV:2013);
Version anglaise FprEN 62453-2:2013**Anwendungswarnvermerk**

Dieser Norm-Entwurf mit Erscheinungsdatum 2014-11-28 wird der Öffentlichkeit zur Prüfung und
Stellungnahme vorgelegt.

Weil die beabsichtigte Norm von der vorliegenden Fassung abweichen kann, ist die Anwendung dieses
Entwurfes besonders zu vereinbaren.

Stellungnahmen werden erbeten

- vorzugsweise online im Norm-Entwurfs-Portal des DIN unter www.entwuerfe.din.de bzw. für Norm-Entwürfe der DKE auch im Norm-Entwurfs-Portal der DKE unter www.entwuerfe.normenbibliothek.de, sofern dort wiedergegeben;
- oder als Datei per E-Mail an dke@vde.com möglichst in Form einer Tabelle. Die Vorlage dieser Tabelle kann im Internet unter www.din.de/stellungnahme oder für Stellungnahmen zu Norm-Entwürfen der DKE unter www.dke.de/stellungnahme abgerufen werden;
- oder in Papierform an die DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik im DIN und VDE, Stresemannallee 15, 60596 Frankfurt am Main.

Die Empfänger dieses Norm-Entwurfs werden gebeten, mit ihren Kommentaren jegliche relevanten
Patentrechte, die sie kennen, mitzuteilen und unterstützende Dokumentationen zur Verfügung zu stellen.

Gesamtumfang 166 Seiten

DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik im DIN und VDE

Nationales Vorwort

Die englische Fassung des europäischen Dokuments FprEN 62453-2:2013 „Field Device Tool (FDT)-Schnittstellenspezifikation – Teil 2: Konzept und grundlegende Beschreibung“ (Entwurf in der Umfrage) ist unverändert in diesen Norm-Entwurf übernommen worden.

Die Internationale Elektrotechnische Kommission (IEC) und das Europäische Komitee für Elektrotechnische Normung (CENELEC) haben vereinbart, dass ein auf IEC-Ebene erarbeiteter Entwurf für eine Internationale Norm zeitgleich (parallel) bei IEC und CENELEC zur Umfrage (CDV-Stadium) und Abstimmung als FDIS (en: Final Draft International Standard) bzw. Schluss-Entwurf für eine Europäische Norm gestellt wird, um eine Beschleunigung und Straffung der Normungsarbeit zu erreichen. Dem entsprechend ist das internationale Dokument IEC 65E/334/CDV:2013 „Field Device Tool (FDT) Interface Specification – Part 2: Concepts and detailed Description“ unverändert in den Entwurf FprEN 62453-2:2013 übernommen worden.

Das internationale Dokument wurde vom SC 65E „Devices and integration in enterprise systems“ der Internationalen Elektrotechnischen Kommission (IEC) erarbeitet.

Bei der Abstimmung zu dem Europäischen Schluss-Entwurf bei CENELEC und dem Internationalen Schluss-Entwurf bei IEC [Final Draft International Standard (FDIS)] sind jeweils nur „JA/NEIN“-Entscheidungen möglich, wobei „NEIN“-Entscheidungen fundiert begründet werden müssen. Dokumente, die bei CENELEC als Europäische Norm angenommen und ratifiziert werden, sind unverändert als Deutsche Normen zu übernehmen.

Für dieses Dokument ist das nationale Arbeitsgremium K 956 „Feldbus“ der DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik im DIN und VDE (www.dke.de) zuständig.

Für den Fall einer undatierten Verweisung im normativen Text (Verweisung auf eine Norm ohne Angabe des Ausgabedatums und ohne Hinweis auf eine Abschnittsnummer, eine Tabelle, ein Bild usw.) bezieht sich die Verweisung auf die jeweils neueste gültige Ausgabe der in Bezug genommenen Norm.

Für den Fall einer datierten Verweisung im normativen Text bezieht sich die Verweisung immer auf die in Bezug genommene Ausgabe der Norm.

Der Zusammenhang der zitierten Normen mit den entsprechenden Deutschen Normen ergibt sich, soweit ein Zusammenhang besteht, grundsätzlich über die Nummer der entsprechenden IEC-Publikation. Beispiel: IEC 60068 ist als EN 60068 als Europäische Norm durch CENELEC übernommen und als DIN EN 60068 ins Deutsche Normenwerk aufgenommen.

Das Präsidium des DIN hat mit Präsidialbeschluss 1/2004 festgelegt, dass DIN-Normen, deren Inhalt sich auf internationale Arbeitsergebnisse der Informationsverarbeitung gründet, unter bestimmten Bedingungen allein in englischer Sprache veröffentlicht werden dürfen. Diese Bedingungen sind für den vorliegenden Normentwurf erfüllt.

Da sich die Benutzer des vorliegenden Normentwurfs der englischen Sprache als Fachsprache bedienen, wird die englische Fassung der E DIN EN 62453-2 veröffentlicht. Zu deren Abschnitt 3, der die Begriffe festlegt, wurde eine Übersetzung angefertigt und als informativer Nationaler Anhang NA des vorliegenden Normentwurfs hinzugefügt. Für viele der verwendeten Begriffe existieren keine gebräuchlichen deutschen Benennungen, da sich die deutschen Anwender in der Regel ebenfalls der englischen Benennungen bedienen. Dieser Normentwurf steht nicht in unmittelbarem Zusammenhang mit Rechtsvorschriften und ist nicht als Sicherheitsnorm anzusehen.

Es ist vorgesehen, auch bei der entsprechenden zukünftigen Deutschen Norm auf die deutsche Sprachfassung zu verzichten und diese in der englischsprachigen Fassung zu veröffentlichen.

Das Original-Dokument enthält Bilder in Farbe, die in der Papierversion in einer Graustufen-Darstellung wiedergegeben werden. Elektronische Versionen dieses Dokuments enthalten die Bilder in der originalen Farbdarstellung.

Es wird auf die Möglichkeit hingewiesen, dass einige Elemente dieses Dokuments Patentrechte berühren können. ISO [und/oder] IEC sind nicht dafür verantwortlich, einige oder alle diesbezüglichen Patentrechte zu identifizieren.

Änderungen

Gegenüber DIN EN 62453-2:2010-02 wurden folgende Änderungen vorgenommen:

- a) Klärung der Kategorien von DTM (z. B. neue Kategorie Composite Device DTM);
- b) Klärung der Befehl-Funktionen;
- c) neues Konzept: statische Funktionen;
- d) Klärung für Kommunikationskanäle und Kommunikation;
- e) Klärung der Identifikation;
- f) Klärung für das Scannen und der DTM-Vergabe;
- g) Neues Konzept: SPS-Tool Unterstützung.

Nationaler Anhang NA (informativ)

3.1 Begriffe

Für die Anwendung dieses Dokuments gelten die Begriffe nach IEC 62453-1 und die folgenden Begriffe.

3.2

FDT Version

(en: **FDT version**)

implementierte Version, festgelegt von der für diese Technik zuständigen Organisation

ANMERKUNG 1 zum Begriff: Die FDT Version ist in IEC/TR 62453-41 oder in IEC/TR 62453-42 festgelegt.

3.3

monolithischer DTM

(en: **monolithic DTM**)

ist ein einziger DTM, welcher das gesamte Gerät mit allen seinen Modulen darstellt

ANMERKUNG 1 zum Begriff: Es gibt auch andere Konzepte, die Module eines Gerätes in dieser Spezifikation darstellen, zum Beispiel Module der DTM und BTM.

– Leerseite –

1	CONTENTS	
2	FOREWORD	2
3	INTRODUCTION	4
4	1 Scope	5
5	2 Normative references	5
6	3 Terms, definitions, symbols, abbreviated terms and conventions	5
7	3.1 Terms and definitions	5
8	3.4 Symbols and abbreviated terms	6
9	3.5 Conventions	6
10	3.5.1 State availability statement	6
11	3.5.2 Data type names and references to data types	6
12	4 Fundamentals	6
13	4.1 General	6
14	4.2 Abstract FDT model	6
15	4.2.1 FDT model overview	6
16	4.2.2 Frame Application (FA)	10
17	4.2.3 Device Type Manager (DTM)	11
18	4.2.4 Channel object	18
19	4.3 Modularity	20
20	4.4 Bus categories	21
21	4.5 Identification	21
22	4.5.1 DTM instance identification	21
23	4.6 System and FDT topology	22
24	4.7 FDT Communication	24
25	4.7.1 General	24
26	4.7.2 Handling of communication requests	25
27	4.7.3 Handling of communication errors	25
28	4.7.4 Handling of loss of connection	25
29	4.7.5 Point-to-point communication	25
30	4.7.6 Nested communication	26
31	4.8 DTM, DTM Device Type and Hardware Identification Information	28
32	4.8.1 DTM and DTM Device Type	28
33	4.8.2 Supported hardware identification	28
34	4.8.3 Connected Hardware Identification	29
35	4.9 DTM data persistence and synchronization	29
36	4.10 DTM device parameter access	30
37	4.11 DTM state machine	31
38	4.11.1 DTM states	31
39	4.11.2 'Communication allowed' sub-states	32
40	4.12 Basic operation phases	33
41	4.12.1 Roles and access rights	33
42	4.12.2 Operation phases	33
43	4.13 FDT version interoperability	34
44	4.13.1 Version interoperability overview	34
45	4.13.2 DTM and device versions	35
46	4.13.3 Persistence	35
47	4.13.4 Nested communication	35

48	5	FDT session model and use cases	36
49	5.1	Session model overview.....	36
50	5.2	Actors	37
51	5.3	Use cases	39
52	5.3.1	Use case overview.....	39
53	5.3.2	Observation	39
54	5.3.3	Operation	39
55	5.3.4	Maintenance	43
56	5.3.5	Planning	47
57	5.3.6	OEM service	50
58	5.3.7	Administration.....	51
59	6	General concepts	52
60	6.1	Address management	52
61	6.2	Scanning and DTM assignment.....	52
62	6.2.1	Scanning introduction	52
63	6.2.2	Scanning	53
64	6.2.3	DTM assignment.....	53
65	6.2.4	Manufacturer specific device identification	53
66	6.2.5	Scan for communication hardware	54
67	6.3	Configuration of fieldbus master or communication scheduler	54
68	6.4	PLC tool support	55
69	6.4.1	General	55
70	6.4.2	Process Image Modifications while PLC is running	56
71	6.5	Slave redundancy	57
72	6.5.1	Redundancy overview.....	57
73	6.5.2	Redundancy support in Frame Application	58
74	6.5.3	Parent component for redundant fieldbus.....	58
75	6.5.4	Redundancy support in Device DTM	58
76	6.5.5	Scan and redundant slaves.....	59
77	7	FDT service specification.....	59
78	7.1	Service specification overview	59
79	7.2	DTM services	60
80	7.2.1	General services.....	60
81	7.2.2	DTM services related to installation	62
82	7.2.3	DTM services related to DTM/device information	62
83	7.2.4	DTM services related to the DTM state machine	64
84	7.2.5	DTM services related to functions	67
85	7.2.6	DTM services related to channel objects.....	70
86	7.2.7	DTM services related to documentation	70
87	7.2.8	DTM services to access the instance data	70
88	7.2.9	DTM services to evaluate the instance data	72
89	7.2.10	DTM services to access the device data	72
90	7.2.11	DTM services related to network management information	74
91	7.2.12	DTM services related to online operation	75
92	7.2.13	DTM services related to data synchronization	76
93	7.2.14	DTM services related to import and export.....	78
94	7.3	Presentation object services	79
95	7.4	Channel object service.....	79
96	7.4.1	Channel object service introduction	79

97	7.4.2	Service ReadChannelInformation.....	79
98	7.4.3	Service WriteChannelInformation.....	80
99	7.5	Process Channel object services	80
100	7.5.1	Services for I/O related information	80
101	7.6	Communication Channel object services	81
102	7.6.1	Services related to communication	81
103	7.6.2	Services related to sub-topology management.....	85
104	7.6.3	Services related to GUI and functions.....	87
105	7.6.4	Services related to scan	88
106	7.7	Frame Application services	88
107	7.7.1	General state availability	88
108	7.7.2	FA services related to general events	88
109	7.7.3	FA services related to topology management.....	90
110	7.7.4	FA services related to redundancy.....	92
111	7.7.5	FA services related to storage of DTM data	93
112	7.7.6	FA services related to DTM data synchronization.....	94
113	7.7.7	FA services related to process image validation	95
114	7.7.8	FA services related to presentation.....	96
115	7.7.9	FA Services related to audit trail.....	97
116	8	FDT dynamic behavior.....	97
117	8.1	Generate FDT topology.....	97
118	8.1.1	FDT topology generation triggered by the Frame Application	97
119	8.1.2	FDT topology generation triggered by the DTM.....	98
120	8.2	Address setting	99
121	8.2.1	Address setting introduction.....	99
122	8.2.2	Set or modify device address – with user interface	99
123	8.2.3	Set or modify device address – without user interface	99
124	8.2.4	Display or modify all child device addresses with user	
125		interface	100
126	8.3	Communication	101
127	8.3.1	Communication overview	101
128	8.3.2	Point-to-point communication.....	101
129	8.3.3	Nested communication	101
130	8.3.4	Device initiated data transfer	102
131	8.4	Scanning and DTM assignment.....	103
132	8.5	Multi-user scenarios.....	104
133	8.5.1	General	104
134	8.5.2	Synchronized and non-synchronized locking mechanism for	
135		DTMs.....	106
136	8.5.3	Additional rules.....	108
137	8.6	Notification of changes.....	108
138	8.7	DTM instance data state machines	108
139	8.7.1	Instance data set introduction	108
140	8.7.2	Modifications state machine.....	109
141	8.7.3	Persistence state machine	110
142	8.7.4	Modification in device	110
143	8.7.5	Storage life cycle	111
144	8.8	Parent component handling redundant slave.....	112
145	8.9	DTM upgrade	113

146	8.9.1	General rules.....	113
147	8.9.2	Saving data from a DTM to be upgraded.....	114
148	8.9.3	Loading data in the replacement DTM.....	114
149	Annex A (normative)	FDT data types definition	116
150	A.1	General.....	116
151	A.2	Basic data types	116
152	A.3	General data types.....	117
153	A.4	User information data types	133
154	A.5	DTM information data type	134
155	A.6	BTM data types.....	135
156	A.7	Device and Scan identification data types	136
157	A.8	Function data types	140
158	A.9	AuditTrail data types	143
159	A.10	Documentation data types.....	144
160	A.11	DeviceList data type	145
161	A.12	Network management data types	147
162	A.13	Instance data types.....	148
163	A.14	DeviceStatus data types	152
164	A.15	OnlineCompare data types.....	153
165	A.16	UserInterface data types	153
166	A.17	Fieldbus specific data types	155
167	Figure 1 – Part 2 of the IEC 62453 series		4
168	Figure 2 – Abstract FDT model		7
169	Figure 3 – Frame Application with integrated Communication Channel		10
170	Figure 4 – Device Type Manager (DTM).....		11
171	Figure 5 – Communication DTM.....		12
172	Figure 6 – Device DTM		12
173	Figure 7 – Gateway DTM		13
174	Figure 8 – Composite Device DTM.....		14
175	Figure 9 – Module DTM		15
176	Figure 10 – Block Type Manager (BTM).....		16
177	Figure 11 – Presentation object		17
178	Figure 12 – Channel object		18
179	Figure 13 – Communication Channel		19
180	Figure 14 – Combined Process / Communication Channel		20
181	Figure 15 – Identification of connected devices.....		22
182	Figure 16 – FDT topology for a simple system topology		23
183	Figure 17 – FDT topology for a complex system topology		24
184	Figure 18 – Point-to-point communication		26
185	Figure 19 – Nested communication		27
186	Figure 20 – DTM, DTM Device Type and Device Identification Information.....		28
187	Figure 21 – Connected Hardware Identification.....		29
188	Figure 22 – FDT storage and synchronization mechanisms.....		30
189	Figure 23 – DTM state machine		31
190	Figure 24 – Substates of communication allowed		32
191	Figure 25 – Main Use Case Diagram.....		37

192	Figure 26 – Observation Use Cases.....	39
193	Figure 27 – Operation Use Cases	40
194	Figure 28 – Maintenance use cases.....	43
195	Figure 29 – Planning use cases	48
196	Figure 30 – OEM service	50
197	Figure 31 – Administrator use cases	51
198	Figure 32 – Address setting via DTM presentation object.....	52
199	Figure 33 – Fieldbus scanning	53
200	Figure 34 – Fieldbus master configuration tool as part of a DTM.....	55
201	Figure 35 – Process Image	56
202	Figure 36 – Transfer of layout information using ProcessImage services	56
203	Figure 37 – Redundancy scenarios	57
204	Figure 38 – FDT topology generation triggered by the Frame Applications.....	98
205	Figure 39 – FDT topology generation triggered by a DTM	98
206	Figure 40 – Set or modify device address – with user interface.....	99
207	Figure 41 – Set or modify device address – with user interface.....	100
208	Figure 42 – Set or modify all device addresses – with user interface.....	100
209	Figure 43 – Point-to-point communication	101
210	Figure 44 – Nested communication	102
211	Figure 45 – Device initiated data transfer.....	103
212	Figure 46 – Scanning and DTM assignment.....	104
213	Figure 47 – Multi-user system	105
214	Figure 48 – General synchronized locking mechanism	106
215	Figure 49 – General non-synchronized locking mechanism	107
216	Figure 50 – Parameterization in case of synchronized locking mechanism	107
217	Figure 51 – Modifications state machine of instance data	109
218	Figure 52 – Persistence state machine of instance data.....	110
219	Figure 53 – Management of redundant topology	113
220	Figure 54 – Associating data to a dataSetId.....	114
221	Figure 55 – Loading data for a supported dataSetId.....	115
222		
223		
224	Table 1 – Description of FDT objects	7
225	Table 2 – Description of associations between FDT objects	8
226	Table 3 – Transitions of DTM states.....	32
227	Table 4 – Transitions of DTM ‘communication allowed’ sub states	32
228	Table 5 – Operation phases	34
229	Table 6 – Actors	38
230	Table 7 – Operation Use Cases	40
231	Table 8 – Maintenance use cases.....	44
232	Table 9 – Planning use cases	48
233	Table 10 – Administrator use cases	51
234	Table 11 – Arguments for service PrivateDialogEnabled	90

235	Table 12 – Arguments for service SetLanguage	61
236	Table 13 – Arguments for service SetSystemGuiLabel	62
237	Table 14 – Arguments for service GetTypeInformation (for DTM)	62
238	Table 15 – Arguments for service GetTypeInformation (for BTM)	63
239	Table 16 – Arguments for service GetIdentificationInformation (for DTM)	63
240	Table 17 – Arguments for service GetIdentificationInformation (for BTM)	63
241	Table 18 – Arguments for service Hardware information (for DTM)	64
242	Table 19 – Arguments for service GetActiveTypeInfo	64
243	Table 20 – Arguments for service GetActiveTypeInfo (for BTM)	64
244	Table 21 – Arguments for service Initialize (for DTM)	64
245	Table 22 – Arguments for service Initialize (for BTM)	65
246	Table 23 – Arguments for service SetLinkedCommunicationChannel	65
247	Table 24 – Arguments for service EnableCommunication	66
248	Table 25 – Arguments for service ReleaseLinkedCommunicationChannel	66
249	Table 26 – Arguments for service ClearInstanceData	66
250	Table 27 – Arguments for service Terminate	67
251	Table 28 – Arguments for service GetFunctions	67
252	Table 29 – Arguments for service InvokeFunctions	68
253	Table 30 – Arguments for service GetGuiInformation	68
254	Table 31 – Arguments for service OpenPresentation	69
255	Table 32 – Arguments for service ClosePresentation	69
256	Table 33 – Arguments for service GetChannels	70
257	Table 34 – Arguments for service GetDocumentation	70
258	Table 35 – Arguments for service InstanceDataInformation	70
259	Table 36 – Arguments for service InstanceDataRead	71
260	Table 37 – Arguments for service InstanceDataWrite	71
261	Table 38 – Arguments for service Verify	72
262	Table 39 – Arguments for service CompareDataValueSets	72
263	Table 40 – Arguments for service DeviceDataInformation	72
264	Table 41 – Arguments for service DeviceDataRead	73
265	Table 42 – Arguments for service DeviceDataWrite	73
266	Table 43 – Arguments for service NetworkManagementInfoRead	74
267	Table 44 – Arguments for service NetworkManagementInfoWrite	74
268	Table 45 – Arguments for service DeviceStatus (for DTM)	75
269	Table 46 – Arguments for service CompareInstanceDataWithDeviceData (for DTM)	75
270	Table 47 – Arguments for service WriteDataToDevice (for DTM)	76
271	Table 48 – Arguments for service ReadDataFromDevice (for DTM)	76
272	Table 49 – Arguments for service OnLockInstanceData	76
273	Table 50 – Arguments for service OnUnlockInstanceData	77
274	Table 51 – Arguments for service OnInstanceDataChanged	77
275	Table 52 – Arguments for service OnInstanceChildDataChanged	77
276	Table 53 – Arguments for service Export	78
277	Table 54 – Arguments for service Import	78

278	Table 55 – Arguments for service ReadChannelInformation	80
279	Table 56 – Arguments for service WriteChannelInformation	80
280	Table 57 – Arguments for service ReadChannelData	80
281	Table 58 – Arguments for service WriteChannelData	81
282	Table 59 – Arguments for service GetSupportedProtocols	81
283	Table 60 – Arguments for service Connect.....	82
284	Table 61 – Arguments for service Disconnect	82
285	Table 62 – Arguments for service AbortRequest	83
286	Table 63 – Arguments for service AbortIndication	83
287	Table 64 – Arguments for service Transaction	83
288	Table 65 – Arguments for service SequenceDefine	84
289	Table 66 – Arguments for service SequenceStart.....	84
290	Table 67 – Arguments for service ValidateAddChild	85
291	Table 68 – Arguments for service ChildAdded.....	85
292	Table 69 – Arguments for service ValidateRemoveChild	86
293	Table 70 – Arguments for service ChildRemoved	86
294	Table 71 – Arguments for service SetChildrenAddresses	86
295	Table 72 – Arguments for service GetChannelFunctions	87
296	Table 73 – Arguments for service GetGuiInformation	87
297	Table 74 – Arguments for service Scan.....	88
298	Table 75 – Arguments for service OnErrorMessage	88
299	Table 76 – Arguments for service OnProgress	89
300	Table 77 – Arguments for service OnOnlineStatusChanged	89
301	Table 78 – Arguments for service OnFunctionsChanged	89
302	Table 79 – Arguments for service GetDtmInfoList	90
303	Table 80 – Arguments for service CreateChild (DTM)	90
304	Table 81 – Arguments for service CreateChild (BTM).....	90
305	Table 82 – Arguments for service DeleteChild	91
306	Table 83 – Arguments for service MoveChild	91
307	Table 84 – Arguments for service GetParentNodes	91
308	Table 85 – Arguments for service GetChildNodes	92
309	Table 86 – Arguments for service GetDtm.....	92
310	Table 87 – Arguments for service ReleaseDtm.....	92
311	Table 88 – Arguments for service OnAddedRedundantChild	93
312	Table 89 – Arguments for service OnRemovedRedundantChild.....	93
313	Table 90 – Arguments for service SaveInstanceData	93
314	Table 91 – Arguments for service LoadInstanceData	94
315	Table 92 – Arguments for service GetPrivateDtmStorageInformation	94
316	Table 93 – Arguments for service LockInstanceData	94
317	Table 94 – Arguments for service UnlockInstanceData	95
318	Table 95 – Arguments for service OnInstanceDataChanged.....	95
319	Table 96 – Arguments for service ValidateProcessImage	95
320	Table 97 – Arguments for service OpenPresentationRequest.....	96

321	Table 98 – Arguments for service ClosePresentationRequest	96
322	Table 99 – Arguments for service UserDialog	96
323	Table 100 – Arguments for service RecordAuditTrailEvent.....	97
324	Table 101 – Modifications state machine of instance data.....	109
325	Table 102 – Persistence state machine of instance data.....	110
326	Table 103 – Example life cycle of a DTM	111
327	Table A.1 – Basic data types	117
328	Table A.2 – Simple general data types.....	117
329	Table A.3 – Definition of classificationId enumeration values	124
330	Table A.4 – General structured data types	125
331	Table A.5 – Simple user information data types	134
332	Table A.6 – Structured user information data type	134
333	Table A.7 – Structured DTM information data type	134
334	Table A.8 – Simple BTM data types	135
335	Table A.9 – Structured BTM data types.....	135
336	Table A.10 – Simple device identification data types.....	137
337	Table A.11 – Structured device identification data types	138
338	Table A.12 – Simple function data types	140
339	Table A.13 – Structured function data types.....	141
340	Table A.14 – Simple auditTrail data types	143
341	Table A.15 – Structured auditTrail data types	143
342	Table A.16 – Simple documentation data types.....	144
343	Table A.17 – Structured documentation data types	144
344	Table A.18 – Simple deviceList data type.....	146
345	Table A.19 – Structured deviceList data type	146
346	Table A.20 – Simple network management data types	147
347	Table A.21 – Structured network management data types.....	147
348	Table A.22 – Simple instance data types	148
349	Table A.23 – Structured instance data types.....	150
350	Table A.24 – Simple device status data types	152
351	Table A.25 – Structured device status data types.....	153
352	Table A.26 – Simple online compare data types.....	153
353	Table A.27 – Structured online compare data types	153
354	Table A.28 – Simple user interface data types	154
355	Table A.29 – Structured user interface data types.....	154
356	Table A.30 – Fieldbus data types.....	155
358		

INTERNATIONAL ELECTROTECHNICAL COMMISSION

FIELD DEVICE TOOL (FDT) INTERFACE SPECIFICATION –

Part 2: Concepts and detailed description

FOREWORD

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

International Standard IEC 62453-2 has been prepared by subcommittee 65E: Devices and integration in enterprise systems, of IEC technical committee 65: Industrial-process measurement, control and automation.

This second edition cancels and replaces the first edition published in 2009. This edition constitutes a technical revision.

The main changes with respect to the previous edition are listed below.

- Clarification for categories of DTMs (e.g. new category 'Composite Device DTM')
- Clarification: Command functions
- New concept: Static function
- Clarification for Communication Channel and communication
- Clarifications for identification
- Clarifications for scanning and DTM assignment

414 • New concept: PLC tool support.

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

FDIS	Report on voting
XX/XX/FDIS	XX/XX/RVD

416

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

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

420 A list of all parts of the IEC 62453 series, under the general title *Field Device Tool (FDT)*
421 *interface specification*, can be found on the IEC website.

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

- 425 • reconfirmed,
- 426 • withdrawn,
- 427 • replaced by a revised edition, or
- 428 • amended.

429

430 The National Committees are requested to note that for this publication the stability date
431 is 2018.

432 THIS TEXT IS INCLUDED FOR THE INFORMATION OF THE NATIONAL COMMITTEES AND WILL BE
433 DELETED AT THE PUBLICATION STAGE.

434

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

439

440

441

442

INTRODUCTION

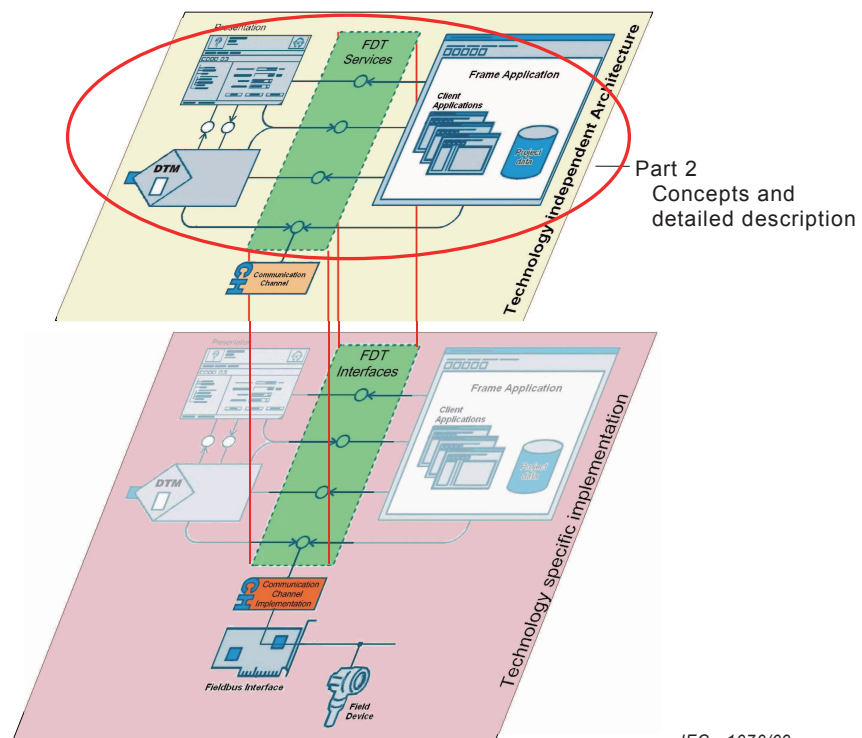
443 This part of IEC 62453 is an interface specification for developers of FDT (Field Device Tool)
 444 components for function control and data access within a client/server architecture. The
 445 specification is a result of an analysis and design process to develop standard interfaces to
 446 facilitate the development of servers and clients by multiple vendors that need to interoperate
 447 seamlessly.

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

454 A device-specific software component created according to this standard is called Device
 455 Type Manager (DTM). It integrates all device-specific data, functions and business rules into
 456 the system via the FDT services defined herein.

457 The FDT/DTM approach is open for all kind of fieldbusses and enables integration variety of
 458 devices into heterogeneous systems.

459 Figure 1 shows how IEC 62453-2 is aligned in the structure of the IEC 62453 series.



IEC 1070/09

460

461

Figure 1 – Part 2 of the IEC 62453 series

462

FIELD DEVICE TOOL (FDT) INTERFACE SPECIFICATION –**Part 2: Concepts and detailed description****1 Scope**

This part of IEC 62453 explains the common principles of the field device tool concept. These principles can be used in various industrial applications such as engineering systems, configuration programs and monitoring and diagnostic applications.

This standard specifies the general objects, general object behavior and general object interactions that provide the base of FDT.

2 Normative references

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

IEC 61131 (all parts), *Programmable controllers*

IEC/TR 62390, *Common automation device – Profile guideline*

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

IEC 62453-3xy (all parts), *Field Device Tool (FDT) interface specification – Part 3xy: Communication profile integration*

IEC/TR 62453-41:-1, *Field Device Tool (FDT) interface specification – Part 41: Object model integration profile – Common object model*

ISO/IEC 19501:2005, *Information technology – Open Distributed Processing – Unified Modeling Language (UML) Version 1.4.2*

3 Terms, definitions, symbols, abbreviated terms and conventions**3.1 Terms and definitions**

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

3.2**FDT version**

implementation version defined by the related technology specific organization

Note 1 to entry: The FDT version is specified in IEC/TR 62453-41 or in IEC/TR 62453-42.

¹ To be published concurrently with this standard.

498 **3.3**
499 **monolithic DTM**
500 is one single DTM that represents the complete device with all its modules

501 Note 1 to entry: There are also other concepts representing modules of a device in this specification, for example
502 Module DTM and BTM

503 **3.4 Symbols and abbreviated terms**

504 For the purposes of this document, the symbols and abbreviations given in IEC 62453-1 and
505 the following apply.

DD	Device description
OODMS	Object Oriented Database Management System
RDBMS	Relational Database Management System

506

507 **3.5 Conventions**

508 **3.5.1 State availability statement**

509 The state availability statement uses [] and [X] for stating the availability of a service in a
510 specific state. The expressions have the following meaning:

511 [] '<name of state>'	service is not available in this state;
512 [X] '<name of state>'	service is available in this state.

513 **3.5.2 Data type names and references to data types**

514 The conventions for naming and referencing data types are explained in subclause A.1

515 **4 Fundamentals**

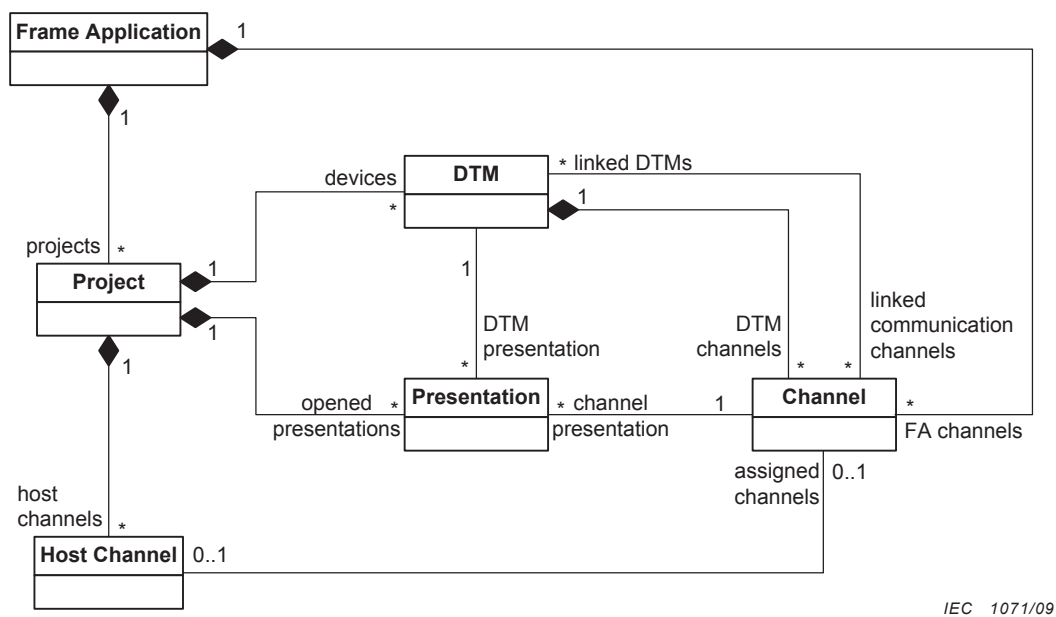
516 **4.1 General**

517 This clause introduces the FDT model and covers topics which are specific to the
518 requirements of field device integration.

519 **4.2 Abstract FDT model**

520 **4.2.1 FDT model overview**

521 Figure 2 provides an overview of the objects defined in FDT and their relationships to each
522 other



IEC 1071/09

Figure 2 – Abstract FDT model

Table 1 contains brief descriptions of all objects. More detail description of objects and related services are provided in subsequent chapters.

The FDT objects may be executed all on one platform or in a distributed system.

For data exchange and for the interaction between objects, the data types as defined in Annex A are used. The concrete implementation of these data types is defined in the IEC 62453-4z documents describing the mapping to specific implementation technologies.

Table 1 – Description of FDT objects

Object	Description
Frame Application	The Frame Application is a logical object to represent an environment like an engineering system or a standalone tool (see 4.2.2). It controls the lifetime of DTM-instances within the project. A Frame Application may handle several projects
Project	The Project belongs to the Frame Application. The Project is a logical object describing the management of device-instances. It controls the lifetime and dataset of device-instances within a Frame Application. FDT does not define any services for the Project, since it is a pure Frame Application internal object

Object	Description
Host Channel	A Host Channel belongs to the Frame Application. The Host Channel is a logical object representing a part of a function of a host system, such as DCS or PLC. It is required when additional information for the processing of device I/O data is needed, e.g. if DTM or BTM data refers to more than one I/O value. For example one Octet often is used to group the state values of eight digital I/O points. In these applications, the Host Channel object provides an association between contents of the channel data and individual process I/O points. To make the cyclic I/O data available within a Frame Application, there shall be an association between the I/O function blocks and the process values of a device. The inputs and outputs of I/O function blocks are represented by Host Channels, the process value is represented by a Channel. The association between Host Channel and Channel is called channel assignment. FDT does not define any services for the Host Channel, since it is a pure Frame Application internal object
DTM	A DTM is a software component containing the device specific application software. It encapsulates all device-specific data, functions and business rules. A DTM is generally not stand-alone executable. It needs a Frame Application (see 4.2.2) that acts as the runtime environment. A DTM is typically developed by the device manufacturer and shipped together with the devices. It depends on the software design of the DTM, whether it can be used for one specific device type or a group of different device types or a device type family. Each device installed on a plant is represented by a DTM object. Therefore one or several DTM objects are needed to handle the different devices. The DTMs are installed in the system, so that the system can be dynamically extended by installing new DTMs for new devices. A DTM may represent different types of devices, e.g. field devices, communication devices or gateway devices. (see 4.2.3)
Presentation	Presentation objects (see 4.2.3.3) represent a (visual) user interface. The Presentation object belongs to the DTM, to the BTM or to the Channel.
Channel	Channel-Object could behave in three ways: As a 'Communication-Channel', a 'Process-Channel' and a combination of both (see 4.2.4).

532

533 All the associations shown in Figure 2 above are explained in Table 2 below.

534

Table 2 – Description of associations between FDT objects

Association	Description
assigned channels	To make the cyclic I/O data available within a Frame Application there has to be an association between the I/O function blocks and the process values of a device. The I/O function blocks are represented by a Host Channel, the process value by a channel. The Frame Application (project) is responsible for handling this association. <u>Use cases:</u> - channel assignment <u>Services:</u> Informational services - DTM service: GetChannels - channel service: ReadChannelInformation Management services - channel service: WriteChannelInformation Depending services - proprietary DTM services for channel management

Association	Description
channel presentation	The instances of presentation objects are associated to the instance of their DTM business object by this relationship. <u>Use cases:</u> - all channel use cases with DTM specific GUI <u>Services:</u> Informational services (see also 7.3) - channel service: GetChannelFuntions - channel service: GetGuiInformation Depending services - frame service: OpenPresentationRequest - frame service: ClosePresentationRequest
devices	A project holds the list of DTM instances by the association devices. Releasing the project results in a release of all associated DTM instances. A DTM instance cannot be part of more than one project. <u>Use cases:</u> - system generation - system planning <u>Services:</u> Informational services - frame service: GetChildNodes Management services - services related to sub-topology management see 7.6.2 - ValidateAddChild - ChildAdded - ValidateRemoveChild - ChildRemoved - FA services related to topology management
DTM channels	A DTM can have a list of channels, which are associated by DTM channels. A channel is part of a DTM. <u>Services:</u> Informational services - frame service: GetChannels Management services - There are no services to manipulate this association. The lifetime of channel instances is controlled by a DTM
DTM presentation	The presentation object instances are associated with their DTM instance by this relationship. <u>Use cases:</u> - All DTM use cases with DTM specific GUI <u>Services:</u> - open presentation - close presentation
FA channels	The Frame Application (project) may have a list of Communication Channels. There are no services to manipulate this association. See 4.2.2
Host Channels	The FA maps the Process Channels of DTMs to internal I/O management of the projects
linked communication channels	The topology of DTMs is shown with this association. Within the topology tree a DTM object is the child node of a channel. The association shows the connection from a device to a channel. The Frame Application (project) is responsible for handling this association. <u>Services:</u> - linked channels can be explored by GetParentNodes - SetLinkedCommunicationChannel - ReleaseLinkedCommunicationChannel

Association	Description
linked DTMs	The topology of field devices is shown with this association. Within the topology tree a channel object is the parent node for a device. The association shows the connection from a channel to a device. The Frame Application (project) is responsible for handling this association. <u>Services:</u> - linked DTMs can be explored by GetChildNodes - SetLinkedCommunicationChannel - ReleaseLinkedCommunicationChannel
opened presentations	Open presentation objects are linked by this association to projects. The Frame Application (project) is responsible for handling this association
projects	A Frame Application can create and manage multiple projects, which are connected by the "projects" association. The Frame Application (project) is responsible for handling this association

535

536 4.2.2 Frame Application (FA)

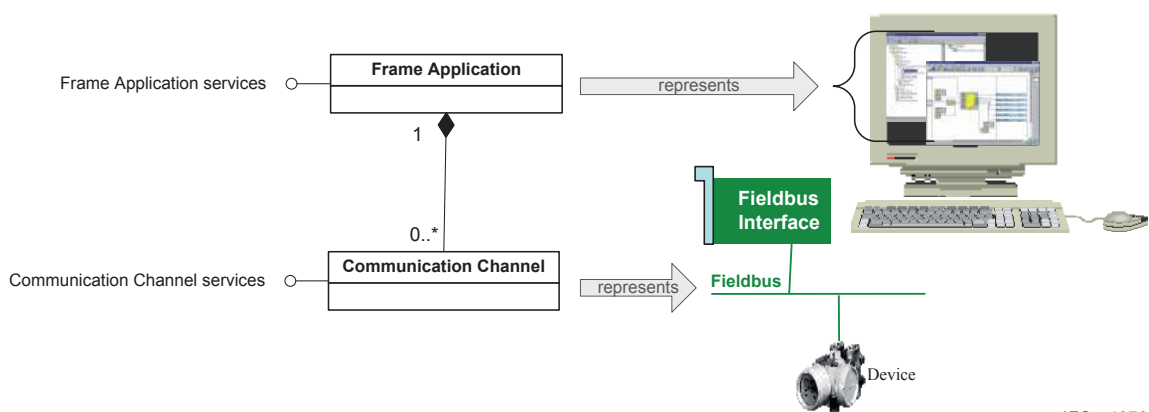
537 4.2.2.1 General

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

544 The Frame Application is the runtime environment for the DTMs. It provides the services
 545 described in 7.7 which may be used by the hosted DTMs.

546 4.2.2.2 System communication

547 If the Frame Application has built-in communication capabilities, then it is able to provide
 548 Communication Channels (see 4.2.4.3) which provide the services to access the fieldbus. In
 549 this case it has full control over communication and is able perform low-level actions such as
 550 monitoring or filtering. Frame Applications without build-in communication capabilities are
 551 using Communication DTMs (see 4.2.3.2.2). Figure 3 shows the relation between the Frame
 552 Application and Communication Channel object.



553

IEC 1072/09

554 **Figure 3 – Frame Application with integrated Communication Channel**

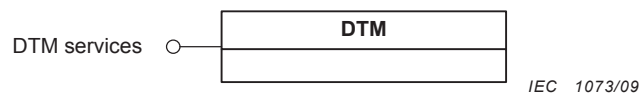
555 The Frame Application's channel objects represent the gateway from the FDT-specific to the
 556 Frame Application-specific communication. On the Frame Application's side the
 557 Communication Channel may be a PC I/O board or engineering topology with processing units
 558 and proprietary bus systems.

559 4.2.3 Device Type Manager (DTM)

560 4.2.3.1 DTM Business Logic Overview

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

565 The DTM BL provides the services described in 7.2 (see Figure 4). From a Frame
566 Application's point of view, all management and task-related interactions with the device
567 functionality are available via these services.



568
569 **Figure 4 – Device Type Manager (DTM)**

570 A DTM may represent the device structure with data type `net:DtmDeviceInstanceTopology`.
571 This information may be retrieved by service `GetActiveTypeInfo` (see 7.2.3.4). The structure of
572 the device may be described by a list of `net:Module`. Each module may have a number of
573 properties, including configuration data, Process Channels and Communication Channels.

574 A DTM exposes a list of supported device types with the data type `fdt:DtmDeviceTypes` (see
575 4.8.1).

576 4.2.3.2 Categories of DTM

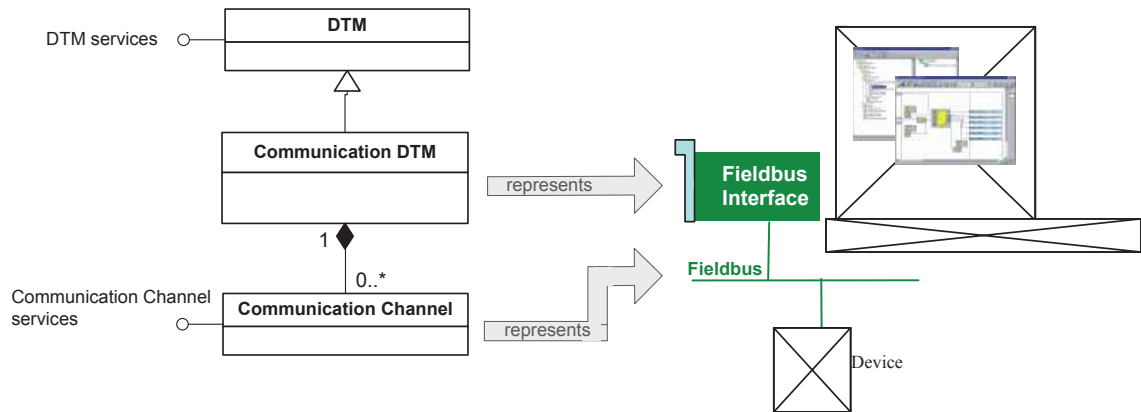
577 4.2.3.2.1 General

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

584 4.2.3.2.2 Communication DTM

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

588 A Communication DTM shall provide Communication Channels (see 4.2.4.3) which provide
589 the services to access the fieldbus (see Figure 5). The number of provided channels may
590 depend on the configuration of the Communication DTM (i.e. it might occur that a DTM
591 temporary provides no channel, for instance during reconfiguration).



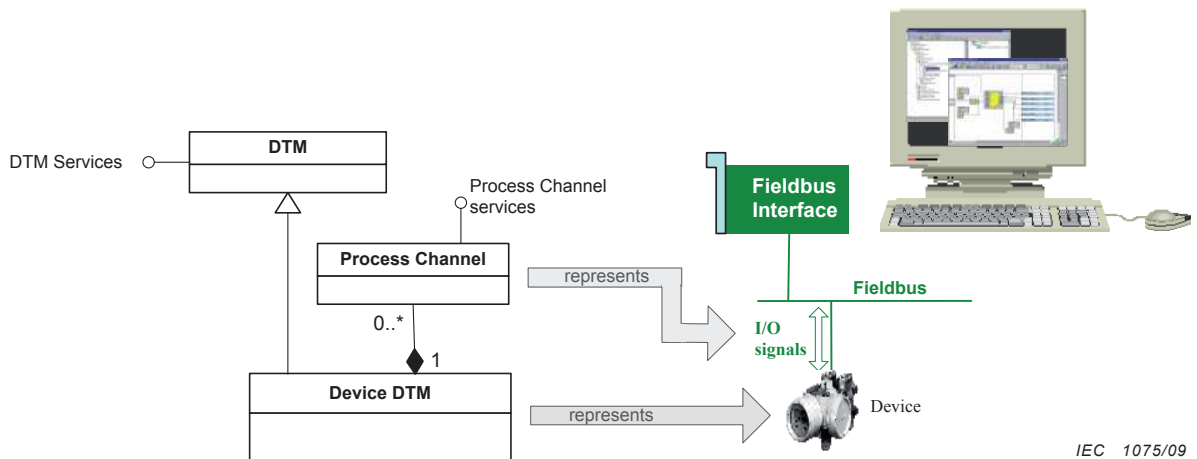
IEC 1074/09

Figure 5 – Communication DTM

The advantage of Communication Channels provided by a DTM compared to the channels provided by a Frame Application (see 4.2.2) is that they may be used by different Frame Applications. Each Frame Application thus may extend the number of protocols the system is able to access. Both ways of providing Communication Channels may coexist in a Frame Application.

4.2.3.2.3 Device DTM

A Device DTM is a special category of DTM containing the application software for a typical field device, for example, a pressure transmitter or a valve. Such a DTM for example supports functions to configure and parameterize the device (see Figure 6).



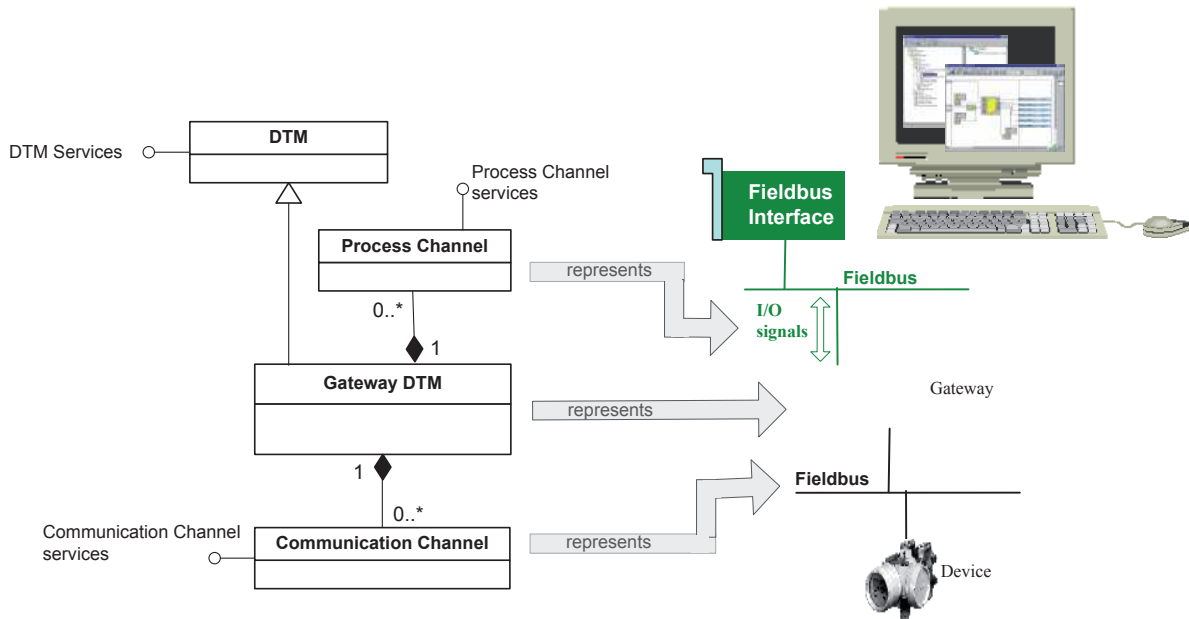
IEC 1075/09

Figure 6 – Device DTM

If it is required to make the device's I/O signals or process values accessible to the host system, then the Device DTM shall provide Process Channel objects (see 4.2.4.2).

4.2.3.2.4 Gateway DTM

A Gateway DTM is a special category of DTM containing the application software for a device that connects different fieldbus segments (see Figure 7).



IEC 1076/09

Figure 7 – 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 4.2.4.3) to provide access to the linked fieldbus and Process Channels (see 4.2.4.2) for the fieldbus where it is connected to, if it is required to make the gateway I/O signals or process values accessible to the host system.

There may be a relation between the I/O signals provided by a gateway device and the communication interface of that device (e.g. the gateway provides I/O signals that are received from a device connected to the communication interface). In such a case the relation is represented as relation between Process Channels 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 section 4.7.

4.2.3.2.5 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 8.

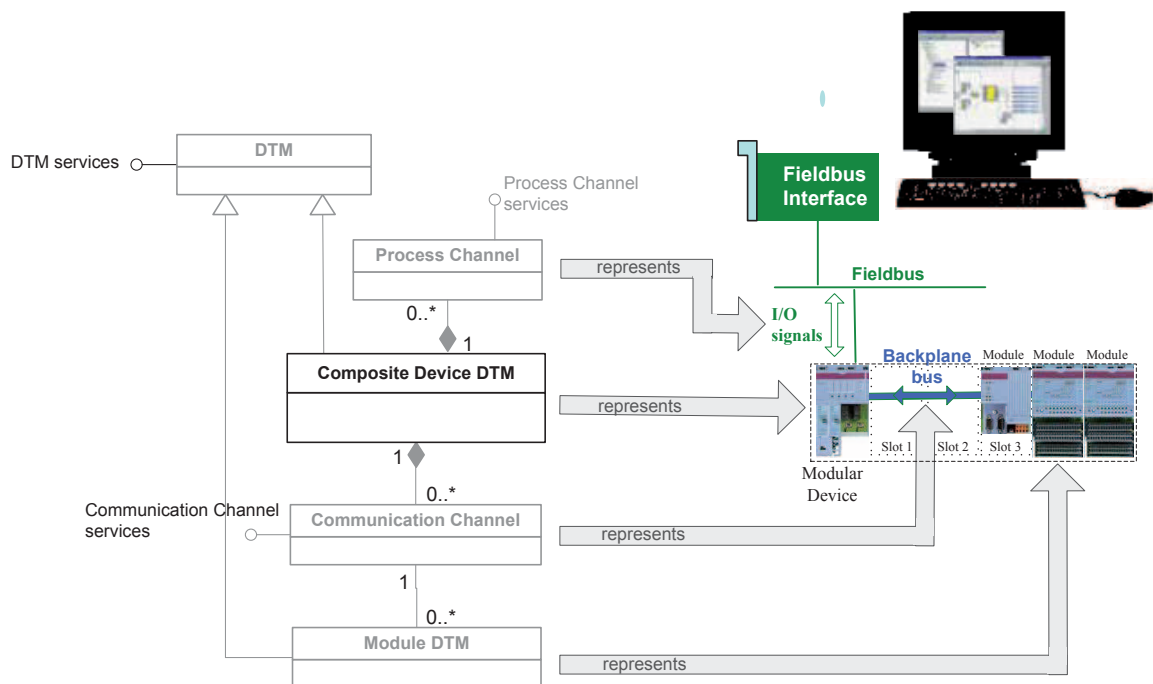


Figure 8 – Composite Device DTM

A Composite Device DTM is the root element for the complete device representation. This DTM shall provide Communication Channels (see 4.2.4.3) 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 Process Channels (see 4.2.4.2). If the composite device forwards process data that is generated in the device modules, then the Process Channels provided by a Composite Device DTM may be composed from Process Channels that are provided by the Module DTMs.

4.2.3.2.6 Module DTM

A Module DTM is a special category of DTM containing the application software for a device hardware or software module (see Figure 9).

Some devices are subdivided into modules and sub-modules. A module is either a hardware module or a software module. Hardware modules are plugged into physical slots of the device. For example, an I/O module can be plugged into a Remote I/O device. Such a modular device may be represented by several DTMs as shown in the following Figure 9.

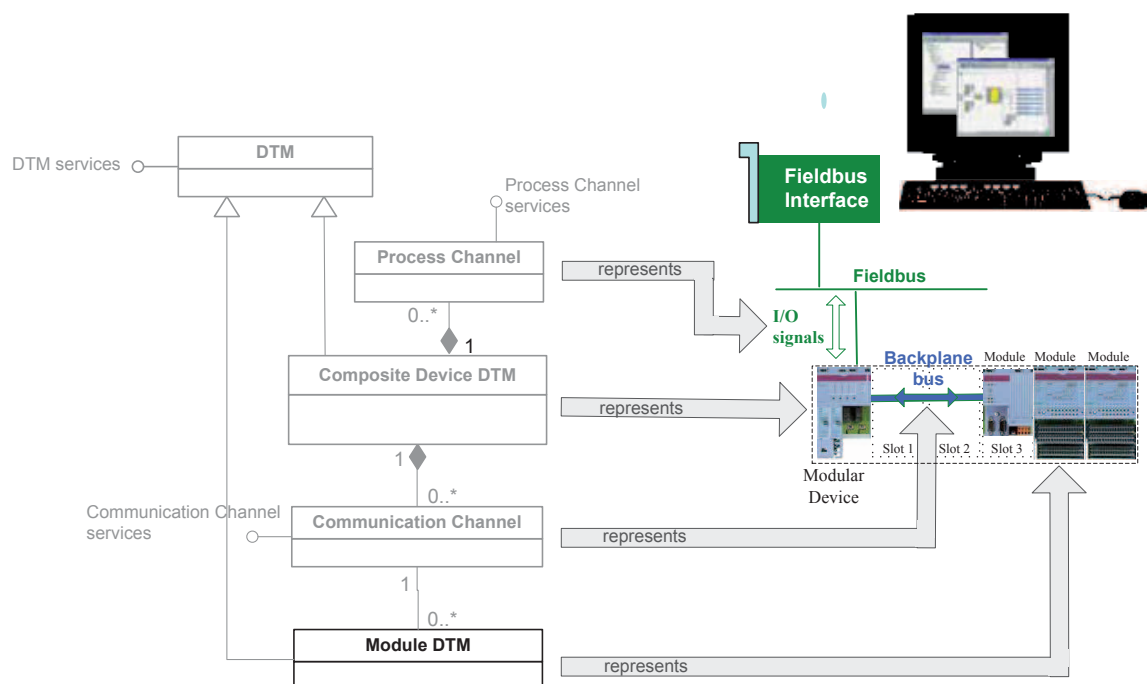


Figure 9 – Module DTM

The 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.

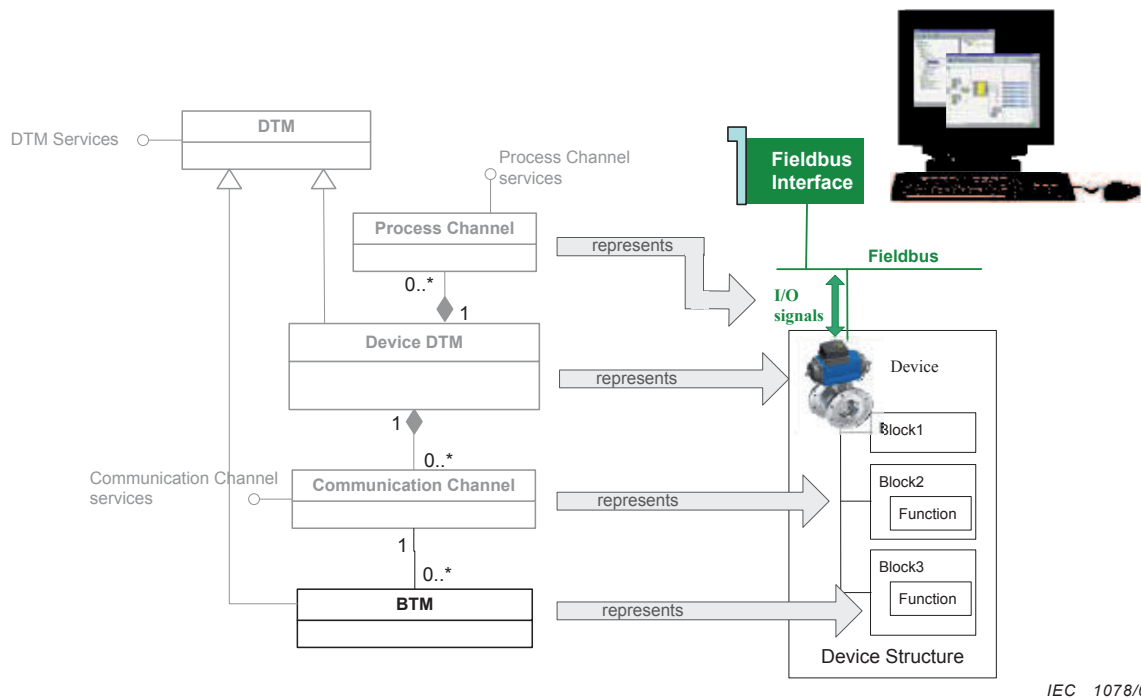
A Module DTM is connected to the Composite Device DTM via the Communication Channel of the Composite Device DTM. The Frame Application is responsible to create and manage the link between the DTMs. The Module DTM is able to communicate with its module via the services provided by the Communication Channel. This means that the concept 'Nested Communication' (see 4.7) 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 Process Channels.

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

4.2.3.2.7 Block Type Manager (BTM)

A BTM is a special category of DTM containing the application software for accessible objects inside a device. Often these software objects are defined as blocks, such as resource blocks, transducer blocks, and function blocks (see Figure 10).



IEC 1078/09

Figure 10 – Block Type Manager (BTM)

A Device DTM is the root element for complete device representation. This Device DTM shall provide Communication Channels (see 4.2.4.3) 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 software blocks are more flexible than device modules (see 4.2.3.2.6), for instance blocks may be moved between devices (standard blocks may be even be moved between devices of different type). The Device DTM and the BTMs for device specific blocks are generally shipped together and 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.

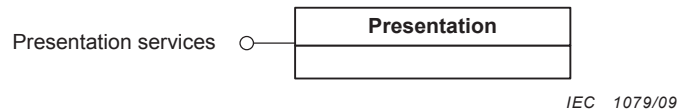
A BTM is connected to the hosting Device DTM via its Communication Channel, indicating that the respective blocks are hosted in the physical device. The BTM is able to interact with its block via the services provided by the Communication Channel of the Device DTM. The Communication Channel represents 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 IEC 62453-3xy documents describing the protocol profile integration in FDT.

If the blocks provide process data, the respective BTMs may provide Process Channels.

696 **4.2.3.3 DTM Functions**697 **4.2.3.3.1 Presentation object**

698 Each of the device specific objects (DTM, BTM and Communication Channel) may be
 699 packaged together with Presentation objects which support the services defined in 7.3 (see
 700 Figure 11)

702 **Figure 11 – Presentation object**

703 These presentation objects are implementation specific objects that provide a graphical user
 704 interface for the business object.

705 The presentation object may be a type of object, that allows integration into the GUI of the FA
 706 or it may be an object that runs outside of the GUI of the Frame Application (e.g. a standalone
 707 executable).

708 **4.2.3.3.2 Command functions**

709 Command functions are used to execute actions (commands) without user interface on the
 710 DTM. Such functions are executed with the service InvokeFunction (see 7.2.5.2).

711 One example for command functions is: Reset device.

712 **4.2.3.3.3 Static Function**

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

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

720 Such use cases typically are implemented in Frame Applications that are 'fieldbus specific',
 721 i.e. such Frame Applications know how to address a device and how to read data from a
 722 device. That is why the Static Function concept provides no means for addressing a device. –
 723 Frame Applications that are 'fieldbus-independent' should use the DTM in order to implement
 724 these use cases. The Static Function concept is intended for these use cases.

725 A Static Function is part of a DTM package, the DTM provides information regarding the Static
 726 Function, but the function may be executed independently from execution of the respective
 727 DTM. A Static Function is a stateless function, i.e. it is not allowed to manage stateful
 728 information or to influence the device state by a Static Function. A Static Function has no
 729 access and no relation to the data of a DTM. A Static Function may be executed multiple
 730 times in parallel in order to process data from one device and from multiple devices.

731 NOTE 1 StaticFunctions are executed using a StaticFunction Provider object. This object can be instantiated once
 732 or multiple times. On one StaticFunction Provider object the execution of a Static Function can be started multiple
 733 times in parallel.

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

NOTE 2 It is not expected that StaticFunctions provide functionality specific for a Frame Application. Instead Profile Working groups (e.g. a working group defining a profile for support of MES functions or a working group defining a DTM profile for support of predictive maintenance) should define standard functions that are used in many applications.

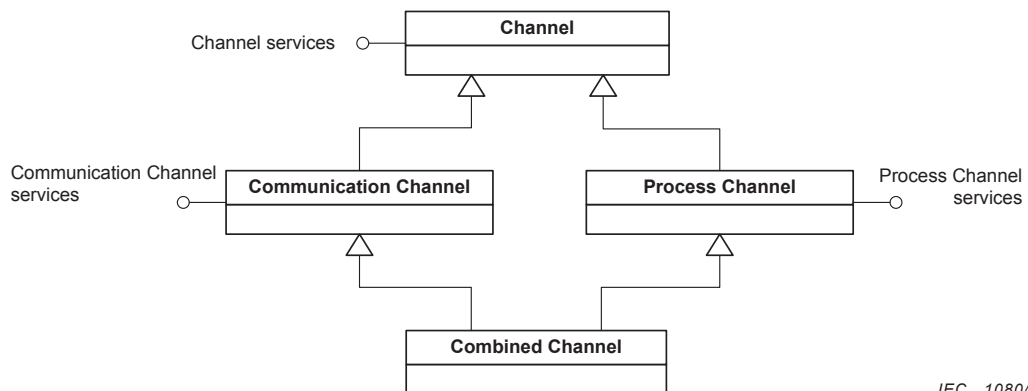
Examples of Static Functions are:

- 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 standard format.

4.2.4 Channel object

4.2.4.1 Channel overview

There are three different types of channels, the Communication Channels the Process Channels and combinations of them as shown in the Figure 12.



IEC 1080/09

Figure 12 – Channel object

4.2.4.2 Process Channel

Input and output signals provided by devices are created and integrated into the function planning of the control system. If the device provides I/O signals the associated DTM shall offer information about I/O signals of its device. This information includes addressing, signal and data type and is conveyed by Process Channels. The information does not include the current I/O signal values.

A Process Channel supports the services defined in 7.5. These services for example provide access to I/O signal information like addressing, signal and data type, but not to the current I/O signal value itself.

The I/O signal information provided by the Process Channel is protocol specific. Which information shall be provided by Process Channels is defined by the IEC 62453-3xy document describing the protocol profile integration in FDT.

The number and type of I/O signals may depend on the configuration of the device. Thus the number of Process Channels and provided information may also dependent on the device configuration made by the user. A Frame Application is able to inform the DTM by setting the ProtectedByChannelAssignent parameter via service WriteChannelData (see 7.5.1.2) that further changes shall be prohibited, because channel is already integrated into functional planning of the control system (Host Channel is assigned). In this case, DTM shall not accept any changes related to this channel.

4.2.4.3 Communication Channel

A Communication Channel provides access to the fieldbus and is communication technology dependent. It represents the entry point to a fieldbus, a vendor specific backplane bus or point-to-point connection. A Communication Channel may also represent logical communication with blocks within a device. A Communication Channel may be provided by a Frame Application (see 4.2.2) or by a DTM (see 4.2.3).

A Communication Channel provides communication services that are independent from the communication hardware and the fieldbus protocol. In general, the protocol specific services are one to one mapped to the services provided by the channel. The interfaces provide methods to

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

The services take base arguments which are extended for protocol-specific communication by IEC 62453-3xy specifications (see Figure 13).

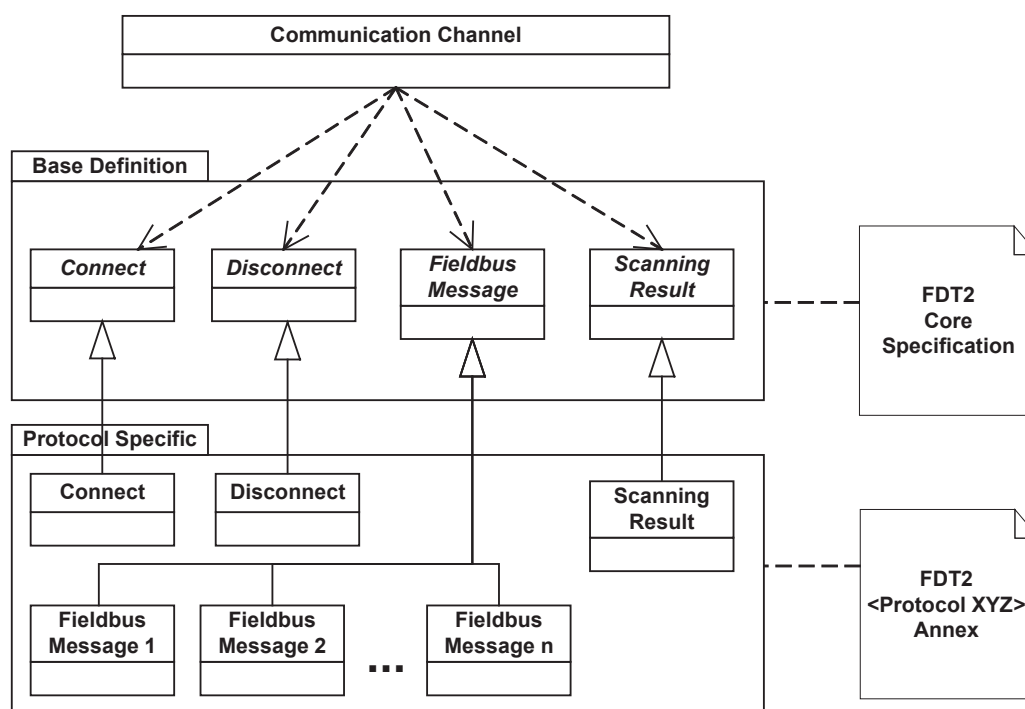


Figure 13 – Communication Channel

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 services provided by Communication Channels may be used by the Frame Application or by DTMs to exchange data with a connected device or to initiate a function (e.g. identification request, device reset, broad-cast, etc.). The general communication concept is further described in 4.7.

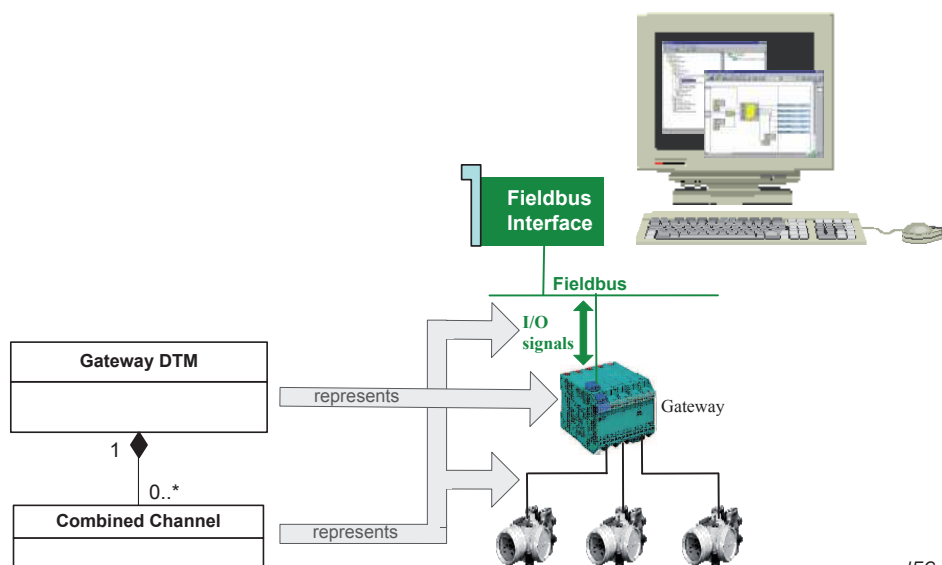
A Communication Channel shall be able to handle several connections at the same time if the communication protocol supports this. This means that a Communication Channel shall be able to support several DTMs creating connections to different devices as well as several

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. The Communication Channel shall guarantee that a link to a device is established as a peer-to-peer connection. The services that shall be provided by a Communication Channel are defined in 7.6. The pure communication related services (see 7.6.1) just define the frame for interaction with a connected device. The data (parameters) which shall be passed or returned by the service are defined by the IEC 62453-3xy documents describing the protocol profile integration in FDT.

In case of vendor specific protocols (e.g. backplane bus or point-to-point) the pure communication related services (see 7.6.1) may be replaced by vendor specific services. Following this approach only DTMs supplied by this vendor are able to communicate with each other. Therefore a vendor specific bus category (see 4.3) shall be defined. For vendor-independent fieldbus protocols an IEC 62453-3xy document describing the protocol profile integration in FDT shall be provided and standard communication services shall be used.

4.2.4.4 Combined Process and Communication Channel

A combined Process and Communication Channel typically belongs to a Gateway DTM (see 4.2.3.2.4). It represents the I/O signal that is exposed via the fieldbus where the gateway device is connected to which corresponds to the cyclic communication on the linked fieldbus. At the same time such a channel represents the entry point to the linked fieldbus (or point-to-point connection) as shown in Figure 14.



IEC 1081/09

Figure 14 – Combined Process / Communication Channel

In other words, such a channel shall support the Process Channel services (see 7.5) for the fieldbus where the gateway is connected to and the Communication Channel Services (see 7.6) for accessing the devices connected to the linked fieldbus (or point-to-point connection).

4.3 Modularity

Different field bus protocols are using different device models. FDT supports following different approaches to describing the structure of device:

- monolithic DTM with DtmDeviceInstanceTopology,
- Module DTM, and
- BTM.

831 4.4 Bus categories

832 A DTM exposes information about encapsulated protocol specific definitions. The information
833 returned by service GetTypeInformation (see 7.2.3.1) contains so called bus categories which
834 are Universally Unique Identifiers for a fieldbus protocol (or a point-to-point communication).

835 The FDT concept distinguishes between supported and required bus categories:

- 836 • supported bus categories refer to the protocol specific communication services provided
837 by a DTM (corresponding Communication Channels) to access a fieldbus;
- 838 • required bus categories refer to the protocol specific communication services required by
839 a DTM to exchange data with its device.

840 4.5 Identification**841 4.5.1 DTM instance identification****842 4.5.1.1 System Tag**

843 An FDT Frame Application shall assign a unique identifier for each DTM instance. This unique
844 identifier is referred to as "System Tag". The System Tag is used by DTMs

- 845 – For navigation in the FDT topology
- 846 – For the management of Child DTMs in the FDT topology (e.g. address setting)

847 4.5.1.2 System GUI label

848 The DtmSystemGuiLabel is a human-readable identifier, used to identify a DTM instance and
849 the related DTM GUI in the display to the user. The Frame Application and the DTM should
850 use the DtmSystemGuiLabel in order to identify dialogs and windows belonging to the DTM
851 instance.

852 For some communication protocols the DtmSystemGuiLabel may reflect an identifier of the
853 device (e.g. the device tag).

854 4.5.1.3 Hardware identification

855 A DTM supports the HardwareInformation service (see 7.2.3.3) that enables to read device
856 information online from the connected device (see Figure 15).

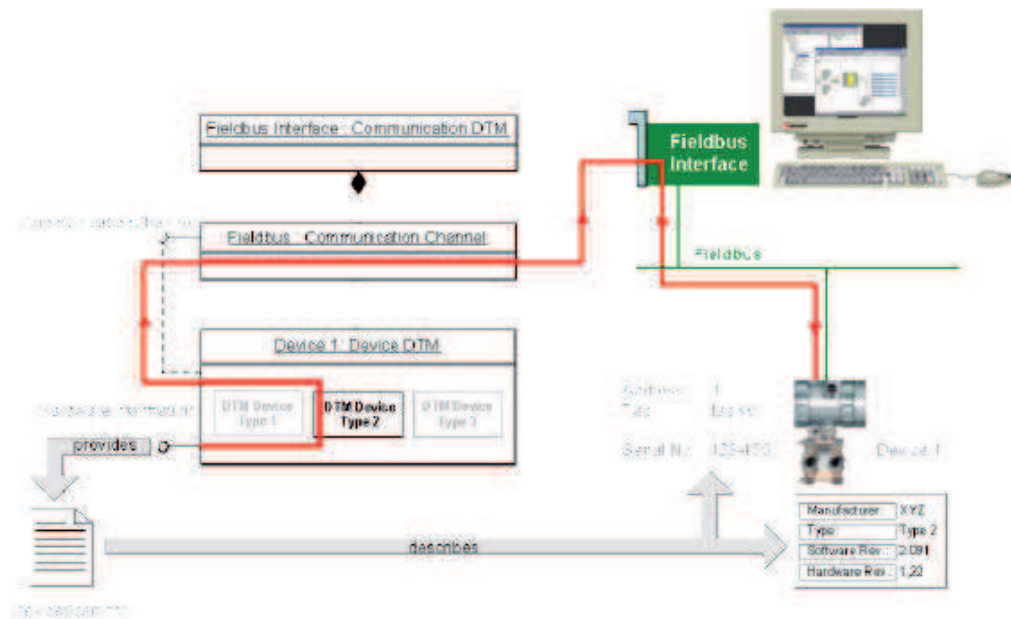


Figure 15 – Identification of connected devices

The service returns device type related information like manufacturer, type, hardware and embedded software version as well as device instance related information like fieldbus address, tag and serial number. This information is also fieldbus specific like the information returned by the service GetIdentificationInformation. Therefore the concrete protocol-specific format and transformation to the protocol-independent format which can be used by Frame Application without fieldbus specific knowledge is defined by the IEC 62453-3xy document describing the protocol profile integration in FDT.

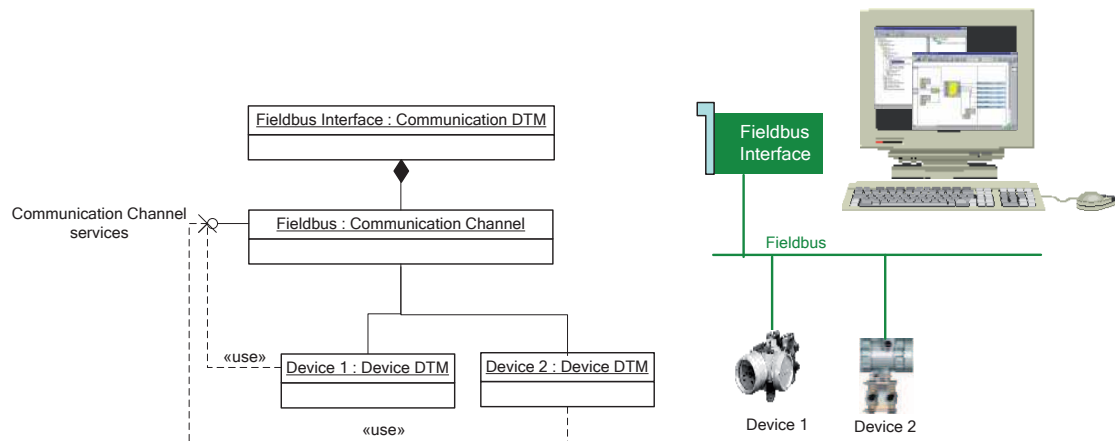
4.6 System and FDT topology

The Frame Application is responsible for managing the system topology. That means the Frame Application shall organize the routing for accessing a device in the plan. A DTM shall not need to manage any information about the system topology.

System topology management is Frame Application specific. Some Frame Applications may require user interactions; others may support automatic operations such as topology importing or fieldbus scanning (see 6.2).

A DTM exposes all required information (see 7.2.3) which enables the Frame Application (and the user) to choose the appropriated DTM for a device, for example name, vendor, version of supported device types and corresponding identification properties.

The Frame Application initializes and links the DTMs according to the system topology. The linking of DTMs is called FDT topology. Figure 16 shows the FDT topology for a simple system topology with one fieldbus and two connected devices.



IEC 1082/09

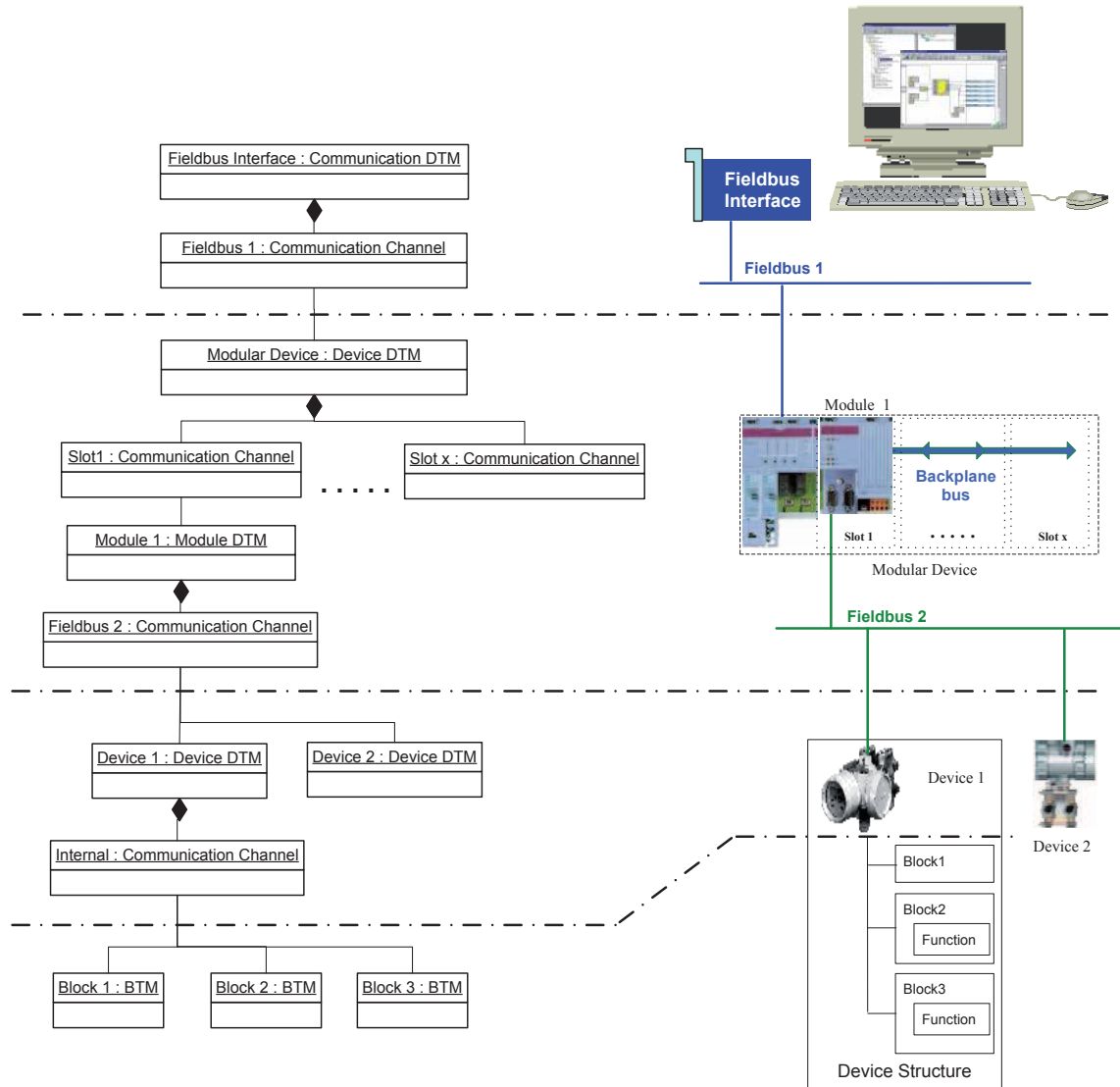
Figure 16 – FDT topology for a simple system topology

A Communication Channel is always used as the linking element between DTMs. As shown in Figure 16, the two Device DTMs are linked to the Communication Channel which provides access to the fieldbus.

The link between a Communication Channel and a DTM is created by the Frame Application. However, final decision whether a DTM shall be linked or not has to be made by the Communication Channel. The Frame Application has to call the service `ValidateAddChild` (see 7.6.2.1) before link is created. The Communication Channel shall at least check whether required bus categories of the DTM to be linked fits to its own supported bus categories. If this is not the case, then linking shall be rejected. In addition, the Communication Channel may perform additional checks, for example if the number of linked DTMs exceeds a limit. The sequence diagrams in 8.1 describe this sequence in more detail.

Neither the Communication Channel (or corresponding DTM) nor the linked DTM shall need to manage topology information. The Frame Application supports requesting topology information by service `GetParentNodes` (see 7.7.3.5) and service `GetChildNodes` (see 7.7.3.6).

The FDT topology management for a more complex system topology with gateways, modular devices and devices with blocks works exactly the same way.



IEC 1083/09

Figure 17 – FDT topology for a complex system topology

Starting from the root, all DTMs are linked via Communication Channels according to the physical system topology.

The root element DTM for a device which is represented by multiple DTMs (e.g. Modular Device and Device 1 DTMs in Figure 17) may support automatic creation of the FDT sub-topology that corresponds to the complete device structure. The Frame Application supports the service CreateChild (see 7.7.3.2), DeleteChild (see 7.7.3.3) and MoveChild (see 7.7.3.4) to enable a DTM to alter the FDT topology.

4.7 FDT Communication

4.7.1 General

In FDT the communication is based on passing communication requests from the consumer of communication to the communication provider, which is represented by a Communication Channel. Passing and receiving communication requests is based on asynchronous services (see 8.3).

In order to submit communication requests the consumer of communication first has to establish a connection to a physical device. This allows the management of active connections by the Communication Channel.

916 4.7.2 Handling of communication requests

917 The Communication Channel receives one or multiple transaction requests from the consumer
918 of communication and executes these requests. In general the Communication Channel is
919 expected to process the transaction requests in the order they are provided. The results of the
920 transaction requests are passed back to the client of Communication Channel.

921 The relation between communication requests and communication responses may be
922 managed by an id, this is implementation specific. For example each transaction request can
923 be identified by an ID, the same ID is provided in the transaction response.

924 4.7.3 Handling of communication errors

925 The communication client shall evaluate each of the received transaction results in order to
926 recognise possible communication errors.

927 If a communication client receives errors for requests that originated at that client, then the
928 client shall inform the user. In a scenario with nested communication this means, the Gateway
929 DTMs shall not inform the user about communication errors for requests that originally came
930 from a Child DTM. The Child DTM has the responsibility for informing the user.

931 4.7.4 Handling of loss of connection

932 If the connection to a device is lost irreparably, the Communication Channel shall abort the
933 connection and send an Abort notification to the respective communication client (see
934 7.6.1.5). The specific triggers for sending the Abort notification (e.g. device does not respond)
935 will be defined specifically for each communication protocol.

936 If a Gateway DTM receives an Abort notification, then it shall send Abort notifications to all
937 connected communication clients.

938 The origin of the Abort notification shall not provide a user message to inform the user about
939 loss of connection. The communication client receiving the Abort notification shall inform the
940 user about the loss of connection. E.g. if the communication client has open user interfaces,
941 then the user shall be informed about the loss of connection by updating the connection
942 status in the user interface. If the communication client is composed of several DTMs which
943 may have open connections at the same time (e.g. a Composite Device DTM with several
944 Module DTMs), then the communication client should avoid providing several user messages,
945 but should inform the user about loss of connection with one message.

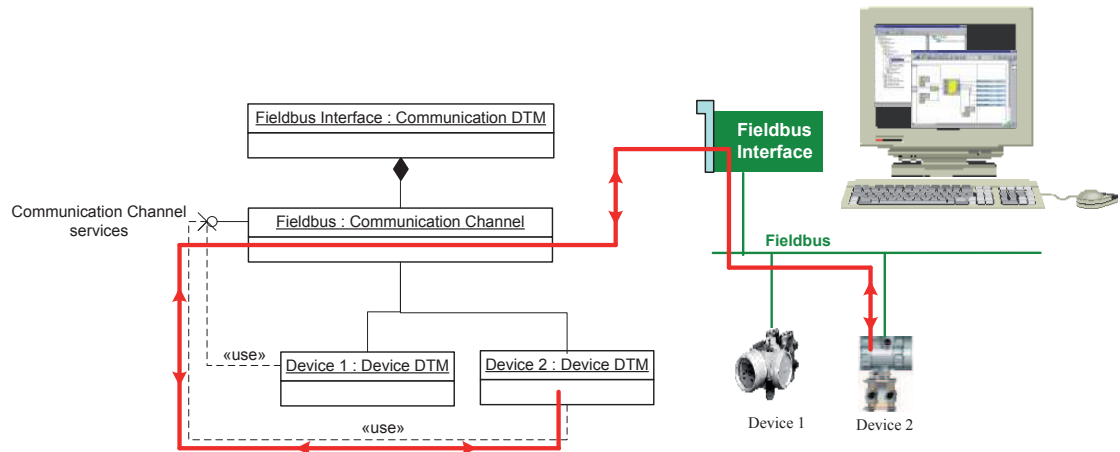
946 After sending an Abort notification the Communication Channel shall not send any further
947 transaction responses to the communication client. The communication client should ignore
948 any transaction response received after receiving an Abort notification.

949 4.7.5 Point-to-point communication

950 The Frame Application has to provide in each case a peer-to-peer connection between a DTM
951 and a device.

952 It is under the control of the Frame Application to enable a DTM to communicate with its
953 device. The Frame Application has to pass the Communication Channel to use the DTM by
954 calling the service SetLinkedCommunicationChannel (see 7.2.4.2) and allow the
955 Communication by calling the service EnabledCommunication with TRUE (see 7.2.4.3).

956 In order to access the device, the DTM uses the Communication Channel services (see 7.6.1)
957 as shown in Figure 18.



IEC 1084/09

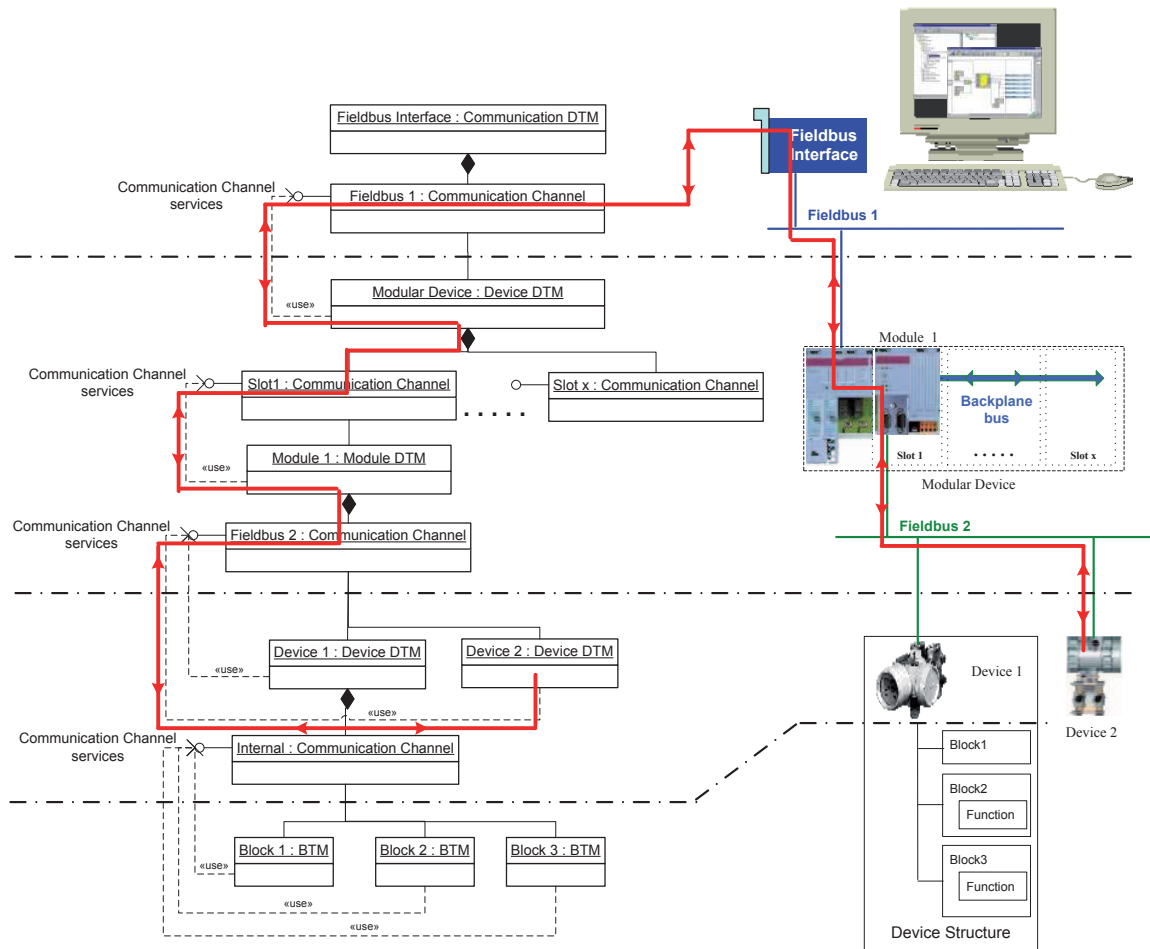
Figure 18 – Point-to-point communication

A DTM has to call the Communication Channel service Connect (see 7.6.1.2) to establish a communication link to the device. After the link is established, it is able to communicate with its device by calling the service Transaction (see 7.6.1.6). The diagram in 8.3.2 describes this sequence in more detail.

DTMs shall consider the limited availability of fieldbus resources when connecting to the device. This means a DTM should create only as many connections as needed and connections shall be terminated if they are not used (e.g. the online function is finished).

4.7.6 Nested communication

A DTM shall not need to know anything about the system topology in order to access the corresponding physical device. Thus the same DTM is able to communicate with devices attached to different fieldbus interfaces of the same protocol. Nested communication is used if the devices are connected to a sub-system, for example if a device which is connected to a channel of a Remote I/O (see Figure 19).



IEC 1085/09

Figure 19 – Nested communication

Starting from the root the Frame Application shall manage the link between Communication Channels and respective DTMs which should be enabled for communication with its device. Like in the point-to-point communication, all DTMs simply use the Communication Channel services to communicate with their devices without the awareness of the nested communication (e.g. device 2 DTM in Figure 19). The diagram in 8.3.3 describes this sequence in more detail.

The communication in FDT follows the concept of nested communication:

- each Communication Channel wraps the communication message from the linked DTM below, without knowing the contents;
- the linked DTM below the Communication Channel does not have any knowledge about the system topology;
- the DTM shall support only communication protocols that are supported by the respective device:
 - communication / routing through any system topology is possible without limitations.

4.8 DTM, DTM Device Type and Hardware Identification Information

4.8.1 DTM and DTM Device Type

A DTM supports two services to request information about itself and the supported device types (see Figure 20). Usually this information is used by the Frame Application to create libraries, select a DTM during creation of FDT topology, or for simple validations.

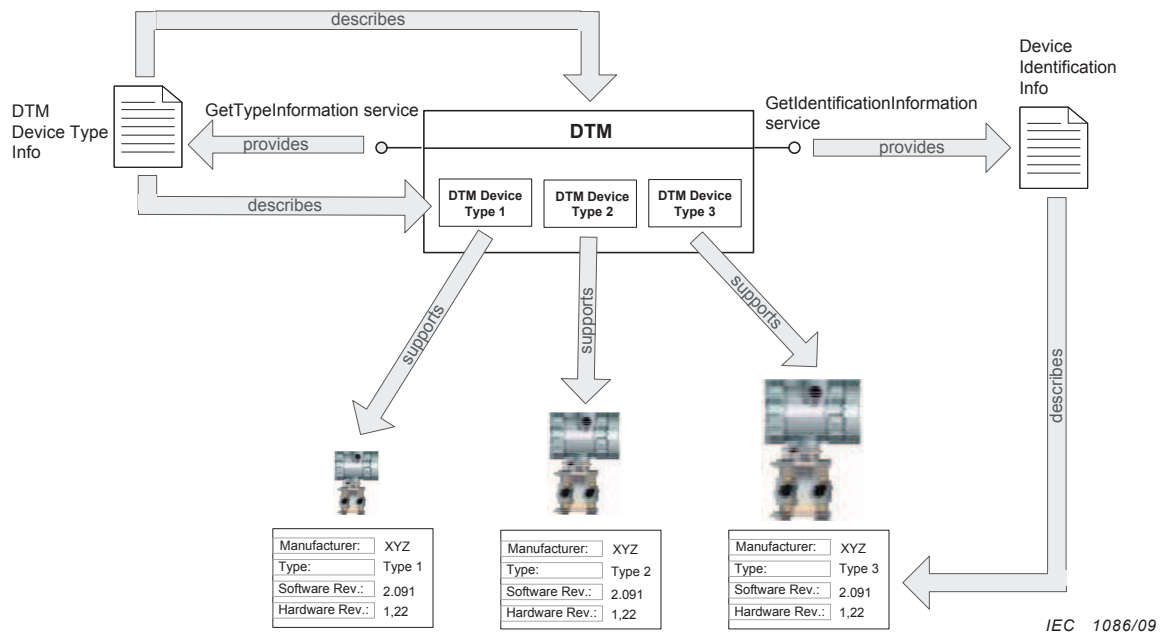


Figure 20 – DTM, DTM Device Type and Device Identification Information

A DTM is a software component containing device specific application software. The service `GetTypeInformation` (see 7.2.3.1) returns general information about this piece of software such as name, version, and vendor of the software, the FDT version (see also 4.13) as well as a list of supported device types.

The representation for a particular device type within the DTM is called DTM Device Type. A DTM may contain one or more DTM Device Types. The concrete design and implementation of the DTM Device Types is not within the scope of the FDT; but service `GetTypeInformation` also returns information about these pieces of software like name, version, vendor, supported protocols etc.

When a Frame Application adds a DTM to the FDT topology then it selects one of the supported DTM Device Types (see service `Initialize`, 7.2.4.1). The DTM shall persist the selection within its instance data, in order to restore it when a DTM representing same device is initiated next time. The DTM shall provide information about the selected DTM Device Type by the service `GetActiveTypeInfo` (see 7.2.3.4). The service shall always return the current DTM Device Type. If an update of the DTM occurred, the new DTM shall return the updated DTM Device Type information.

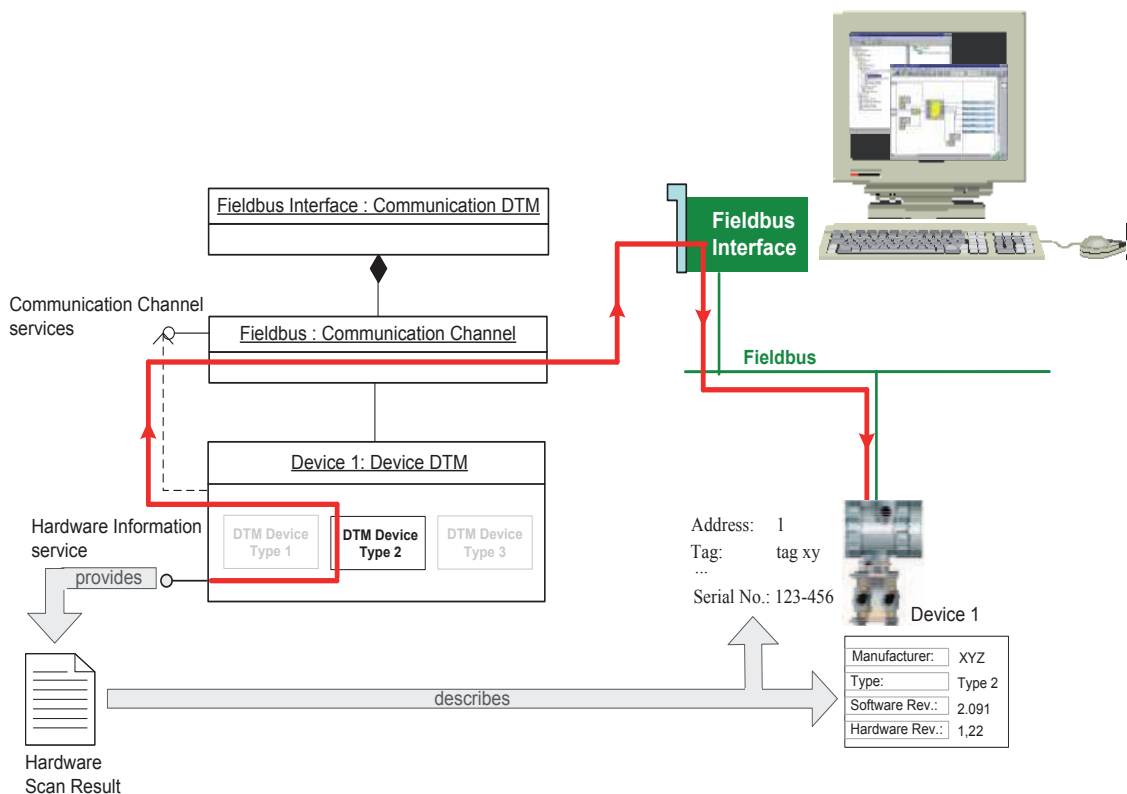
4.8.2 Supported hardware identification

Information about physical device types, which can be handled by the DTM Device Types is returned by the service `GetIdentificationInformation` (see 7.2.3.2). For example manufacturer, type, hardware and embedded software version of the device. The DTM may even return wildcards for some specific device identification elements to signal that the DTM Device Type can be used for all devices for which the wildcard matches (e.g. the character asterisk '*' for the hardware version may signal that the DTM Device Type supports all hardware versions).

1020 The information returned by service GetIdentificationInformation is fieldbus-specific and
 1021 therefore defined by the IEC 62453-3xy document describing the protocol profile integration in
 1022 FDT. However, FDT defines the means to transform the information into a protocol-
 1023 independent format to enable Frame Applications without protocol-specific knowledge to use
 1024 it. The transformation is implementation technology dependent and therefore defined in the
 1025 IEC 62453-4z documents describing the mapping to specific implementation technologies.

1026 4.8.3 Connected Hardware Identification

1027 Furthermore a DTM supports service HardwareInformation (see 7.2.3.3) that enables to read
 1028 device information online from the connected device (see Figure 21).



1030 **Figure 21 – Connected Hardware Identification**

1031 The service returns device type related information like manufacturer, type, hardware and
 1032 embedded software version as well as device instance related information like fieldbus
 1033 address, tag and serial number. This information is also fieldbus specific like the information
 1034 returned by GetIdentificationInformation (see 7.2.3.2). Therefore, concrete format and
 1035 transformation to protocol-independent format is also defined by the IEC 62453-3xy document
 1036 describing the protocol profile integration in FDT. See examples in 6.2.4 and 6.2.5.

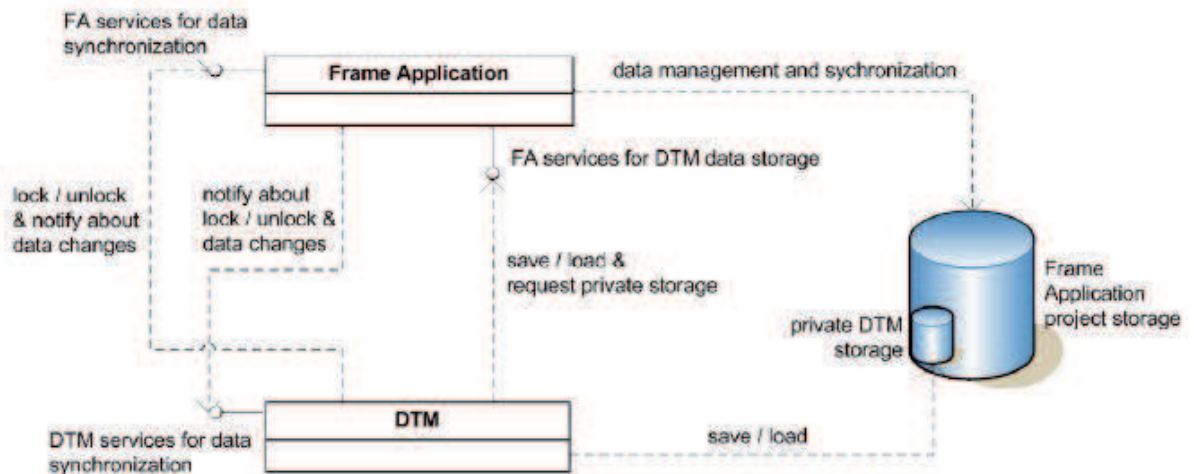
1037 4.9 DTM data persistence and synchronization

1038 Basically, three kinds of DTM related data can be seen:

- 1039 • instance-related data (called “instance data”). Instance-related data belongs to the DTM
 1040 itself. It is up to the DTM which data it stores but the DTM has to guarantee that it is able
 1041 to represent the stored device instance by loading these data;
- 1042 • non-instance-related data. Non-instance-related data belongs to a project or is global or
 1043 DTM Device Type specific;
- 1044 • bulk data. DTM specific data, for example historical data, temporary data.

1045 The format of all kinds of data is DTM-specific, but a DTM shall expose an Universal Unique
1046 identifier specifying the format which is used to save its instance data. Furthermore, a DTM
1047 shall provide a list of compatible data formats it is able to load. These format identifiers are
1048 included in the DTM Device Type information returned by GetTypeInformation (see 7.2.3.1). A
1049 Frame Application may use the format identifiers to find correct DTM after a DTM software
1050 upgrade (see 8.9).

1051 Two types of DTM data persistence mechanism are defined in FDT. A DTM shall either use
1052 the Frame Application project storage or a private DTM storage (see Figure 22).



IEC 1088/09

Figure 22 – FDT storage and synchronization mechanisms

1055 The services LoadInstanceData (see 7.7.5.1) and service SaveInstanceData (see 7.7.5.2)
1056 enable the DTM to save and load instance-related data in the Frame Application project
1057 storage. The Frame Application has to guarantee the data consistency for multi-user and
1058 multi-client data access. The 'FA services for DTM data synchronization' (see 7.7.6) shall be
1059 used by the DTM to attempt locking of its instance-related data before alternation. The 'DTM
1060 services related to data synchronization' (see 7.2.13) shall be used by the Frame Application
1061 to notify a DTM about events, for example if data was locked by another DTM instance. See
1062 8.5 Multi-user scenarios.

1063 The services GetPrivateDtmStorageInformation (see 7.7.5.3) returns information about
1064 private DTM storage which is directly accessible for the DTM without the need to use further
1065 services provided by the Frame Application. The private DTM storage enables the DTM to
1066 store instance-related, non-instance-related, and bulk data. The DTM itself has to guarantee
1067 the data consistency for multi-user and multi-client data access. It is in the responsibility of
1068 the Frame Application to create a backup strategy for the private DTM storages.

1069 A DTM shall be able to work without any side effects if the private DTM storage is not
1070 available. It shall ensure that there is no impact to the instance data managed by service
1071 SaveInstanceData (see 7.7.5.1) and LoadInstanceData (see 7.7.5.2).

1072 The private DTM storage mechanism is implementation technology dependent and thus
1073 defined in the IEC 62453-4z documents defining mapping to specific technologies.

1074 **4.10 DTM device parameter access**

1075 A DTM supports services to read and write device parameters stored in the DTM instance
1076 data (see 7.2.8) and directly in the connected device (see 7.2.10).

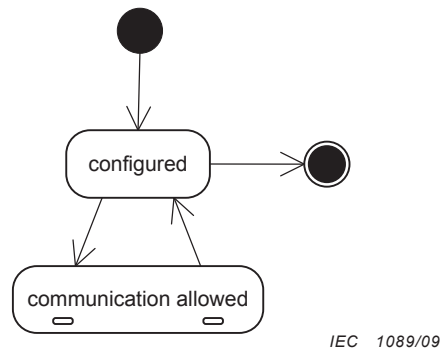
1077 It is specific to a DTM and depends on the device- and fieldbus-type which device parameters
1078 are available. Semantic information (see SemanticId in Table A.4) is given to the device
1079 parameters. By using these, for example a Frame Application is able to get the information

1080 regarding the meaning and usage of a single parameter or data structure. The definition
 1081 regarding the identifier is protocol-specific and thus defined in the IEC 62453-3xy
 1082 specifications defining the protocol profile integration in FDT.

1083 4.11 DTM state machine

1084 4.11.1 DTM states

1085 The following diagram (Figure 23) defines the different states of a DTM.



1086
1087 **Figure 23 – DTM state machine**

1088 The state machine consists of two states: 'configured' and 'communication allowed'. Which
 1089 DTM services are available in a certain state is defined in the service descriptions (see
 1090 clause 7).

1091 All state transitions are triggered by the Frame Application by calling corresponding DTM
 1092 services. The state transition succeeds, if the service returns `fdt:serviceSucceeded = TRUE`.
 1093 The DTM remains in its previous state, if state transition is rejected by returning
 1094 `fdt:serviceSucceeded = FALSE`. The circumstances under which the DTM is allowed to reject
 1095 the state transitions are defined in the service descriptions.

1096 **State Transition: Initial state – state 'configured'**

1097 The service Initialize (see 7.2.4.1) shall be called to bring the DTM from its initial state into
 1098 the state 'configured'.

1099 **State Transition: State 'configured' – state 'communication allowed'**

1100 The service EnableCommunication (see 7.2.4.3) shall be called with `TRUE` to bring the DTM
 1101 from state 'configured' into the state 'communication allowed'. A precondition for this call is
 1102 that the Frame Application has informed the DTM about the communication infrastructure by
 1103 calling the service SetLinkedCommunicationChannel (see 7.2.4.2).

1104 Only in this state the DTM is allowed to establish a communication link to its device by calling
 1105 the service Connect (see 7.6.1.2) of its linked Communication Channel. The sub-states within
 1106 this state are described in the communication state machine (see 4.11.2).

1107 **State Transition: State 'communication allowed' – state 'configured'**

1108 The service EnableCommunication (see 7.2.4.3) shall be called with `FALSE` to bring the DTM
 1109 from state 'communication allowed' into the state 'configured'.

1110 **State Transition: State 'configured' – final state**

1111 The service Terminate (see 7.2.4.6) shall be called to bring the DTM from state 'configured' into its final state.

1113 Table 3 shows an overview of these transitions

1114 **Table 3 – Transitions of DTM states**

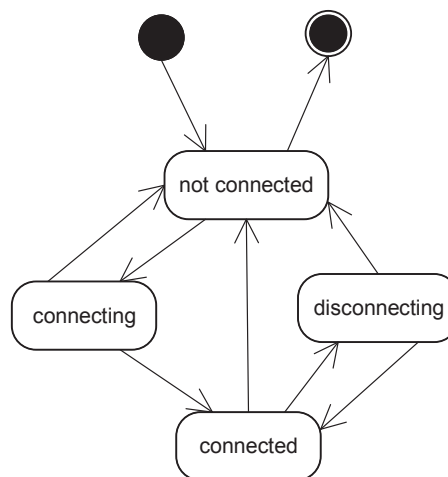
Start state	End state	Trigger	Condition
Initial state	Configured	Initialize	
Configured	Communication allowed	EnableCommunication(TRUE)	Communication infrastructure available to DTM
Communication allowed	Configured	EnableCommunication(FALSE)	
Configured	Final state	Terminate	

1115

1116 **4.11.2 'Communication allowed' sub-states**

1117 This state machine is describing the sub-states in the DTM state 'communicationAllowed' (see Figure 24).

1118



IEC 1090/09

1119

1120 **Figure 24 – Substates of communication allowed**

1121 Table 4 provides a description of the transitions of the DTM 'communication allowed' sub-states.

1122

1123 **Table 4 – Transitions of DTM 'communication allowed' sub states**

Start state	End state	Trigger	Condition	Action
not connected	connecting	Trigger of Online service of the DTM		Connect Request
connecting	not connected		Connection could not be established	
connecting	connected	Connect Response		
connected	not connected	Connection is terminated by communication		

Start state	End state	Trigger	Condition	Action
connected	disconnecting	Online Service of DTM is finished		Disconnect Service
disconnecting	connected		Connection could not be terminated	
disconnecting	disconnected	Disconnect Response		

1124

1125 **4.12 Basic operation phases**1126 **4.12.1 Roles and access rights**

1127 When a DTM is started by the Frame Application, the user role is passed to the DTM, which
 1128 shall restrict the parameter access according to the role.

1129 Also the availability of functions and appearance of user interfaces may be adapted.

1130 Examples:

- 1131 • an observer will not get access to calibration functions;
- 1132 • an observer shall not modify parameters.

1133 See 5.2.

1134 **4.12.2 Operation phases**

1135 If a Frame Application requests available functions from the DTM, it passes the operation
 1136 phase, which shall be used by a DTM to adapt the availability of functions and the
 1137 appearance of the user interface.

1138 There are five operation phases:

- 1139 • Engineering
 - 1140 Planning and configuring of a plant,
 - 1141 No online access to the plant;
- 1142 • Commissioning, workshop
 - 1143 Installing the plant infrastructure and the devices,
 - 1144 Scanning the network to verify a planned network,
 - 1145 Downloading project data into the plant,
 - 1146 Programming, diagnosis of the devices,
 - 1147 Adjusting parameters;
- 1148 • Runtime
 - 1149 The plant is completely commissioned and running,
 - 1150 Very restricted access to configuration and parameterization data,
 - 1151 Scan to verify the network,
 - 1152 Replacing defect devices,
 - 1153 Reading process values and diagnosis information;
- 1154 • Service
 - 1155 This operation phase is used for service tool Frame Applications.
 - 1156 Scan to create a topology. Scan to verify a network. All service related operations.
 - 1157 Unrestricted access to the device;
- 1158 • Not supported

1159 The application does not support the operation phases defined above.

1160 A DTM shall offer all functions if the Frame Application passes “not supported”.

1161 The following table (see Table 5) describes the usage of operation phases in different Frame
1162 Application types.

1163 **Table 5 – Operation phases**

System Frame Application	Service tool Frame Application	Other Frame Applications
Engineering	Service	notSupported
Commissioning		
Runtime		

1164 For example the DTM allows the maintenance actor during commissioning phase the
1165 complete Online parameterization, but during runtime phase it does not allow it or it allows it
1166 only for some of its parameters

1167 **4.13 FDT version interoperability**

1168 **4.13.1 Version interoperability overview**

1169 FDT is a component based concept, which is constantly enhanced to new improved versions.
1170 Control system environments typically run for 10-15 years in contrast. If hardware and
1171 software components in a control system have to be exchanged, a mixture of components
1172 designed for different FDT versions may emerge.

1173 This raises the question, how components based on different FDT versions can cooperate.

1174 This subclause 4.13 mainly deals with two topics concerning FDT version interoperability:

1175 a) Persistence

1176 New versions of FDT components (Frame Applications and DTMs) shall be able to load
1177 instance data of older versions.

1178 b) Component Interoperability

1179 Components of different FDT versions shall interoperate properly. DTMs of newer FDT
1180 versions shall run in older Frame Applications and vice versa. Communication shall work
1181 even if FDT versions of Device DTMs and Communication DTMs differ. Interaction
1182 between these components is technology dependent and therefore details defined within
1183 IEC 62453-4z documents.

1184 A limiting condition for future FDT-enhancements is to ensure maximum compatibility of
1185 different FDT versions. This results in a maximum interoperability of FDT components
1186 originally designed for different FDT versions.

1187 Within IEC 62453-4z documents, it is taken care for version interoperability on two different
1188 levels:

1189 a) Specification Level

1190 The FDT specification targets full compatibility of different versions of the standard.
1191 Properly designed FDT components will achieve compatibility without extra
1192 implementations.

1193 b) Implementation Level

1194 Within IEC 62453-4z documents the FDT version is composed by a major, a minor, a
1195 release and a build number (e.g. 1.2.0.3 where 1 is the major, 2 is the minor number, 0
1196 the release and 3 the build number). Compatibility is defined slightly different with respect
1197 to the major number:

- 1198 – compatibility between FDT components with the same FDT major version number is
 1199 assured by the IEC 62453-4z specification;
- 1200 – compatibility between FDT components with different FDT minor, release and build
 1201 version numbers can be achieved by extra code paths. FDT specification will provide
 1202 needed information and mechanisms to support full compatibility at this level. The
 1203 minor or release number is increased if new functionality, dependent of its appropriate
 1204 importance, is added to the FDT specification. The build number is increased if a
 1205 bugfix within the specification or the type library is necessary.

1206 4.13.2 DTM and device versions

1207 When providing a new DTM for a new major FDT version, it is highly recommended to use
 1208 new registration information (see 7.2.2). If registration information of a DTM has changed,
 1209 FDT IEC 62453-4z will provide a mechanism to upgrade to the newer.

1210 Within its device catalog, a new DTM version is able to provide the same list or a subset of
 1211 the devices of the old DTM version. In this case, a FDT Frame Application will handle the two
 1212 versions of a DTM as any other DTMs able to handle the same field device as two
 1213 distinguished DTMs.

1214 A DTM of a higher FDT version which does not use new registration information shall support
 1215 at least all DTM Device Types of that were supported by previous versions of this DTM.

1216 4.13.3 Persistence

1217 If the version of the Frame Application changes or if the version of the DTM changes
 1218 persistence (loading of stored projects) may be a problem.

1219 A Frame Application shall be able to load old instance data and to provide the unchanged
 1220 instance data to the DTM even if the DTM version has changed. A DTM shall be able to load
 1221 old instance data as long as the registration information of the DTM does not change. If
 1222 registration information of a DTM has been changed, FDT specification provides a technology
 1223 dependent mechanism to upgrade to the newer DTM defined within IEC 62453-4z documents.
 1224 By this way, it is possible to load an existing old instance data of a DTM into another DTM
 1225 object. The new DTM is responsible to convert the instance data accordingly.

1226 A new DTM shall provide information what DTM instance data it can load (compatibility of
 1227 dataset). It shall support all DTM Device Types of the old DTM. If the Frame Application
 1228 recognizes the new DTM, it may inform the user (e.g. “A new compatible DTM was installed,
 1229 do you want to use the new version instead of the old version?”). If the user confirms the new
 1230 DTM object is instantiated instead of the old and it loads and converts the old instance data.

1231 4.13.4 Nested communication

1232 4.13.4.1 Nested communication overview

1233 Since DTMs may be chained with respect to the FDT topology, they need to communicate
 1234 properly. Because the involved DTMs may support different FDT versions, the following
 1235 restrictions shall be considered.

1236 4.13.4.2 Information exchange

1237 Information exchanged via communication services may contain version dependent
 1238 parameters.

1239 Therefore, a DTM shall only use communication service parameters according to the FDT
 1240 version of its parent component. This version information of the parent component shall be
 1241 provided within the service Connect (see 7.6.1.2) response information. A DTM shall then use
 1242 only communication parameters compatible to this FDT version.

1243 **4.13.4.3 Communication Channel upgrade**

1244 A Communication Channel shall support the older FDT minor versions if it is upgraded to a
1245 higher minor version. This is due to the fact that some of its already existing children may not
1246 support the higher minor version interfaces.

1247 **4.13.4.4 Version compatibility**

1248 Within nested communication the version number returned by a Gateway DTM depends on
1249 the functionality of its parent component.

1250 If the Gateway DTM is at a higher version number than the parent component then the
1251 Gateway DTM:

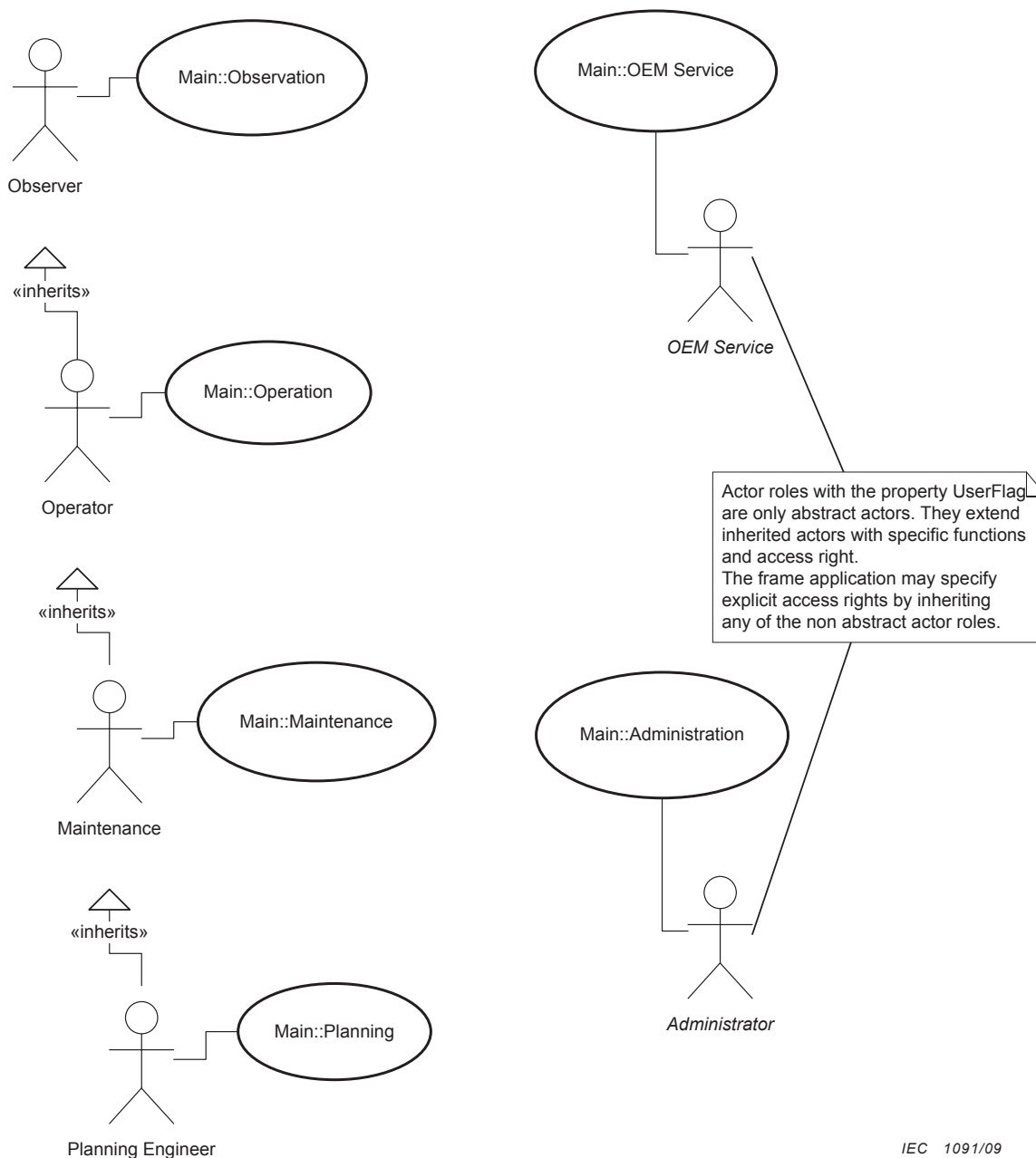
- 1252 • shall return the same version as the parent and provide the functionality according to this
1253 version or
- 1254 • shall emulate the higher functionality if possible. Then the Gateway DTM shall guarantee
1255 that all required functionality is provided by its parent.

1256 **5 FDT session model and use cases**

1257 **5.1 Session model overview**

1258 This clause describes the use cases that were considered when designing the FDT
1259 Specification. A Frame Application may support only a subset of these use cases. Also a
1260 Frame Application may support additional use cases that are not defined in this clause.

1261 Figure 25 shows the main use cases.



IEC 1091/09

1262

1263

Figure 25 – Main Use Case Diagram

1264 They are specified in more detail in the following subclauses, including textual descriptions
 1265 and optionally some necessary scenarios.

1266 5.2 Actors

1267 The actor types in the above displayed use case diagram are structured in a hierarchical way.

1268 The “Maintenance” actor inherits the “Operator” actor. The “Planning Engineer” actor inherits
 1269 the use cases of the “Maintenance” actor. Both the “Administrator” and the “OEM Service” can
 1270 extend the inherited actors by additional use cases. Use cases, which are passed on to an
 1271 actor of higher level, may act in an extended way to match their intention.

1272 The following table (Table 6) gives a brief description of the actors roles:

1273

Table 6 – Actors

Actor Name:	Observer
Brief Description:	This actor stands for a person that may only observe the current process
Inherits:	None
User Level:	FDTUserInformation.userLevel = observer
Access Verification:	The access as “Observer” actor may not have a password
Use Cases:	None
Actor Name:	Operator
Brief Description:	This actor stands for a person who has to observe and manage the current process. The “Operator” may therefore check the current status of the device, modify set values and check if the device is functioning well. The use cases for this actor role should enable the user to perform a complete diagnosis, watch the actual status and parameter set as well as the current process variables
Inherits:	Observer
User Level:	FDTUserInformation.userLevel = operator
Access Verification:	The access as “Operator” requires a password
Use Cases:	User Login, Online View, Audit Trail, Archive, Report Generation, Asset Management
Actor Name:	Maintenance
Brief Description:	A “Maintenance” person should have the possibility to perform all necessary maintenance operations including device exchange, teaching, calibration, adjustment, ... The person may therefore download verified parameter sets, modify a subset of parameters online or offline, perform device-specific online operations and at the end of the processing, may have the possibility to upload the complete parameter set
Inherits:	Operator
User Level:	FDTUserInformation.userLevel = maintenance
Access Verification:	The access as “Maintenance” actor requires a password
Use Cases:	Simulation, Online Operation, Repair, Offline Operation, Bulk Data Handling User Login*, Online View*, Audit Trail*, Archive*, Report Generation*, Asset Management* (* Inherited use cases)
Actor Name:	Planning Engineer
Brief Description:	The actor “Planning Engineer” stands for person like a plant engineer, a specialist (s. VDI/VDE2187) or any fully authorized person. This Person has access to the complete set of functions of the Frame Application and can use DTM functions without any restrictions. Only OEM-specific operations are locked
Inherits:	Maintenance
User Level:	FDTUserInformation.userLevel = planningEngineer
Access Verification:	The access as “Planning Engineer” actor requires a password
Use Cases:	DTM Instance Handling, Simulation*, Online Operation*, Repair*, Offline Operation*, Bulk Data Handling*, User Login*, Online View*, Audit Trail*, Archive*, Report Generation*, Asset Management* (* Inherited use cases)

1274

1275

1276

Actor Name:	Administrator (abstract actor)
Brief Description:	The "Administrator" actor stands for person that has to perform administrative operations within an engineering environment He is responsible for adding and removing of software components
Inherits:	Depends on the Frame Application
User Flag*:	FDTUserInformation.administrator = TRUE
Access Verification:	The access as "Administrator" actor requires a password
Use Cases:	Integration and all inherited use cases
Actor Name:	OEM Service (abstract actor)
Brief Description:	The actor "OEM Service" may perform the complete set of DTM specific functions. OEM specific operation may be accessible to an "OEM Service" actor, like resetting of internal counters and exchanging firmware. The "OEM Service" has only inherited use cases, but the inherited use cases (especially the "Online Operation" use case) may have significant extensions
Inherits:	Depends on the Frame Application
User Flag*:	FDTUserInformation.oemService = TRUE
Access Verification:	The access verification for an "OEM Service" shall have two security levels: 1. Frame Application password: enables access to the Frame Application functionality 2. Device-specific password: enables access to OEM operation with the device. The verification of the device specific password lies in the responsibility of the DTM or even the device itself
Use Cases:	Inherited use cases only
* "User Flags" may be combined with any "User Level", to specify the inherited actor role of an "Administrator" or "OEM Service" actor.	

1277

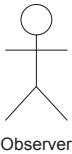
1278 5.3 Use cases

1279 5.3.1 Use case overview

1280
1281
1282

The following subclause describes all use cases of the use case diagram in tabular form. To reduce information, all attributes of included use cases, which are identical with the related main use case, are not displayed.

1283 5.3.2 Observation



IEC 1092/09

1284

1285 Figure 26 – Observation Use Cases

1286
1287
1288

Only the observation use cases can be executed by the actor "Observer" (see Figure 26). The "Observer" stands for a person that has the lowest access level. No use case is associated with this person.

1289 5.3.3 Operation

1290 The Operation use cases are available for the actor "Operator" (see Figure 27).