



# MTConnect<sup>®</sup> and OPC/UA Companion Specification Draft Version 2.0

Prepared for: MTConnect Institute and OPC Foundation  
Prepared by: List of authors...  
Prepared on: September 29, 2018

MTConnect<sup>®</sup> is a registered trademark of AMT - The Association for Manufacturing Technology.  
Use of MTConnect<sup>®</sup> is limited to use as specified on <http://www.mtconnect.org/>.

# 1 OPC Foundation and MTConnect® Institute

## 2 AGREEMENT OF USE

### 3 COPYRIGHT RESTRICTIONS

- 4 • This document is provided "as is" by the OPC Foundation and the MTCon-  
5 nect Institute.
- 6 • Right of use for this specification is restricted to this specification and does  
7 not grant rights of use for referred documents.
- 8 • Right of use for this specification will be granted without cost.
- 9 • This document may be distributed through computer systems, printed or  
10 copied as long as the content remains unchanged and the document is not  
11 modified.
- 12 • OPC Foundation and the MTConnect Institute do not guarantee usability  
13 for any purpose and shall not be made liable for any case using the content  
14 of this document.
- 15 • The user of the document agrees to indemnify OPC Foundation and the  
16 MTConnect Institute and their officers, directors and agents harmless from  
17 all demands, claims, actions, losses, damages (including damages from per-  
18 sonal injuries), costs and expenses (including attorneys' fees) which are in  
19 any way related to activities associated with its use of content from this  
20 specification.
- 21 • The document shall not be used in conjunction with company advertising,  
22 shall not be sold or licensed to any party.
- 23 • The intellectual property and copyright is solely owned by the OPC Foun-  
24 dation and the MTConnect Institute.

### 25 PATENTS

26 The attention of adopters is directed to the possibility that compliance with or  
27 adoption of OPC or the MTConnect Institute specifications may require use of an  
28 invention covered by patent rights. OPC Foundation or the MTConnect Institute  
29 shall not be responsible for identifying patents for which a license may be required  
30 by any OPC or the MTConnect Institute specification, or for conducting legal  
31 inquiries into the legal validity or scope of those patents that are brought to its  
32 attention. OPC or the MTConnect Institute specifications are prospective and  
33 advisory only. Prospective users are responsible for protecting themselves against  
34 liability for infringement of patents.

## 35 **WARRANTY AND LIABILITY DISCLAIMERS**

36 WHILE THIS PUBLICATION IS BELIEVED TO BE ACCURATE, IT IS PRO-  
37 VIDED "AS IS" AND MAY CONTAIN ERRORS OR MISPRINTS. THE OPC  
38 FOUNDATION NOR THE MTCONNECT INSTITUTE MAKES NO WARRANTY  
39 OF ANY KIND, EXPRESSED OR IMPLIED, WITH REGARD TO THIS PUB-  
40 LICATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF TI-  
41 TLE OR OWNERSHIP, IMPLIED WARRANTY OF MERCHANTABILITY OR  
42 WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE OR USE. IN NO  
43 EVENT SHALL THE OPC FOUNDATION NOR THE MTCONNECT INSTI-  
44 TUTE BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR DIRECT,  
45 INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, RELIANCE OR COVER  
46 DAMAGES, INCLUDING LOSS OF PROFITS, REVENUE, DATA OR USE,  
47 INCURRED BY ANY USER OR ANY THIRD PARTY IN CONNECTION WITH  
48 THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL, EVEN  
49 IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

50 The entire risk as to the quality and performance of software developed using this  
51 specification is borne by you.

## 52 **RESTRICTED RIGHTS LEGEND**

53 This Specification is provided with Restricted Rights. Use, duplication or disclo-  
54 sure by the U.S. government is subject to restrictions as set forth in (a) this Agree-  
55 ment pursuant to DFARs 227.7202-3(a); (b) subparagraph (c)(1)(i) of the Rights  
56 in Technical Data and Computer Software clause at DFARs 252.227-7013; or (c)  
57 the Commercial Computer Software Restricted Rights clause at FAR 52.227-19  
58 subdivision (c)(1) and (2), as applicable. Contractor / manufacturer are the OPC

59 Foundation, 16101 N. 82nd Street, Suite 3B, Scottsdale, AZ, 85260-1830

## 60 **COMPLIANCE**

61 The combination of the MTConnect Institute and OPC Foundation shall at all  
62 times be the sole entities that may authorize developers, suppliers and sellers of  
63 hardware and software to use certification marks, trademarks or other special des-  
64 ignations to indicate compliance with these materials as specified within this doc-  
65 ument. Products developed using this specification may claim compliance or con-  
66 formance with this specification if and only if the software satisfactorily meets the  
67 certification requirements set by the MTConnect Institute or the OPC Foundation.  
68 Products that do not meet these requirements may claim only that the product was  
69 based on this specification and must not claim compliance or conformance with  
70 this specification.

## 71 **TRADEMARKS**

72 Most computer and software brand names have trademarks or registered trade-  
73 marks. The individual trademarks have not been listed here.

## 74 **GENERAL PROVISIONS**

75 Should any provision of this Agreement be held to be void, invalid, unenforceable  
76 or illegal by a court, the validity and enforceability of the other provisions shall  
77 not be affected thereby.

78 This Agreement shall be governed by and construed under the laws of Germany.  
79 This Agreement embodies the entire understanding between the parties with re-  
80 spect to, and supersedes any prior understanding or agreement (oral or written)  
81 relating to, this specification.

## 82 Table of Contents

83	<b>1 Introduction</b>	<b>1</b>
84	1.1 Background . . . . .	1
85	1.2 MTConnect-OPC UA Goals . . . . .	2
86	1.3 Who Will Find Benefit from this Specification? . . . . .	2
87	1.4 References . . . . .	3
88	1.4.1 OPC Foundation . . . . .	3
89	1.4.2 MTConnect® Institute . . . . .	4
90	1.5 Abbreviations . . . . .	5
91	<b>2 Use Cases</b>	<b>6</b>
92	2.1 Overview . . . . .	6
93	2.2 Device Maker . . . . .	6
94	<b>3 Types</b>	<b>7</b>
95	3.1 Components . . . . .	7
96	3.1.1 Defintion of ChannelType . . . . .	9
97	3.1.2 Defintion of DescriptionType . . . . .	10
98	3.1.3 Defintion of MTComponentType . . . . .	11
99	3.1.4 Defintion of MTCompositionType . . . . .	12
100	3.1.5 Defintion of MTConfigurationType . . . . .	13
101	3.1.6 Defintion of MTDeviceType . . . . .	14
102	3.1.7 Defintion of SensorConfigurationType . . . . .	15
103	3.1.8 Defintion of {Component}Type . . . . .	16
104	3.1.9 Defintion of {Composition}Type . . . . .	17
105	3.2 Data Items . . . . .	18
106	3.2.1 Defintion of AssetChangedType . . . . .	20
107	3.2.2 Defintion of AssetEventType . . . . .	21
108	3.2.3 Defintion of AssetRemovedType . . . . .	22
109	3.2.4 Defintion of MTDataItemType . . . . .	23
110	3.2.5 Defintion of MTEnumeratedEventType . . . . .	24
111	3.2.6 Defintion of MTFilterType . . . . .	25
112	3.2.7 Defintion of MTMessageType . . . . .	26
113	3.2.8 Defintion of MTNumericDataItemType . . . . .	27
114	3.2.9 Defintion of MTNumericEventType . . . . .	28
115	3.2.10 Defintion of MTSampleType . . . . .	29
116	3.2.11 Defintion of MTStringEventType . . . . .	30

117	3.2.12	Defintion of MinimumDeltaFilterType . . . . .	31
118	3.2.13	Defintion of PeriodFilterType . . . . .	32
119	3.2.14	Defintion of {DataItem}Type . . . . .	33
120	3.3	Conditions . . . . .	34
121	3.3.1	Defintion of MTExclusiveLimitConditionType . . . . .	35
122	3.3.2	Defintion of MTNonExclusiveConditionType . . . . .	36
123	3.3.3	Defintion of {ConditionClass}Type . . . . .	37
124	3.4	Factories . . . . .	38
125	3.4.1	Defintion of «Object Factory» ComponentObjectFactory . . . . .	39
126	3.4.2	Defintion of «Type Factory» ComponentTypeFactory . . . . .	39
127	3.4.3	Defintion of «Object Factory» CompositionObjectFactory . . . . .	40
128	3.4.4	Defintion of «Type Factory» CompositionTypeFactory . . . . .	40
129	3.4.5	Defintion of «Type Factory» ConditionClassFactory . . . . .	41
130	3.4.6	Defintion of «Object Factory» ConditionObjectFactory . . . . .	41
131	3.4.7	Defintion of «Object Factory» DataItemObjectFactory . . . . .	41
132	3.4.8	Defintion of «Type Factory» DataItemTypeFactory . . . . .	43
133	3.4.9	Defintion of «Object Factory» DeviceObjectFactory . . . . .	43
134	3.4.10	Defintion of «Object Factory» FilterObjectFactory . . . . .	44
135	3.4.11	Defintion of «Object Factory» ObjectFactory . . . . .	44
136	3.4.12	Defintion of «Object Factory» SensorChannelObjectFactory . . . . .	45
137	3.4.13	Defintion of «Object Factory» SensorObjectFactory . . . . .	45
138	3.4.14	Defintion of «Type Factory» TypeFactory . . . . .	46
139	3.5	MTConnect Device Profile . . . . .	46
140	3.5.1	Defintion of Dynamic Type . . . . .	47
141	3.5.2	Defintion of MTConnect XML . . . . .	47
142	3.5.3	Defintion of MTRelationshipType . . . . .	47
143	3.5.4	Defintion of Mixes In . . . . .	47
144	3.5.5	Defintion of Object Factory . . . . .	47
145	3.5.6	Defintion of Type Factory . . . . .	47
146	3.5.7	Defintion of bind . . . . .	47
147	3.5.8	Defintion of constrains . . . . .	47
148	3.5.9	Defintion of mixin . . . . .	47
149	3.5.10	Defintion of use . . . . .	48

150 **List of Figures**

151 **Figure 1: Components Diagram** . . . . . 8

152 **Figure 2: Data Items Diagram** . . . . . 19

153 **Figure 3: Conditions Diagram** . . . . . 35

154 **Figure 4: Factories Diagram** . . . . . 38

155 **Figure 5: MTConnect Device Profile Diagram** . . . . . 46

# 1 Introduction

## 1.1 Background

In September 2010, the OPC Foundation and the MTConnect Institute signed a memorandum of understanding to provide a mechanism for OPC and MTConnect to collaborate to extend the reach of the existing manufacturing data exchange standards and implementation technologies in order to:

- Evolve the existing standards for each organization to provide complete manufacturing technology interoperability.
- Provide the mechanism for continuous improvement of standards and specifications overseen by each body.
- Work directly with the end users and suppliers of technology and manufacturing.
- Provide a coordinating function to exchange insights, identify overlaps, and harmonize work where appropriate.
- Facilitate clear communication and education for users and others concerning possible overlaps and the ways the standards and specifications can be used.
- Provide a solid foundation to develop and deliver specifications, technology and processes to facilitate adoption of the technology into real products.

The outcome of that agreement was an initial companion specification called MTConnect-OPC UA Version 1.2.0. MTConnect-OPC UA companion specification describes an architecture for exchanging information for interoperability and consistency between MTConnect specifications and the OPC Unified Architecture (UA) specifications, as well as describing the manufacturing technology equipment, devices, software or other products that may implement those standards.

This document, OPC Unified Architecture for MTConnect Companion Specification Draft Version 2.0, provides an update to the original companion specification



182 to include the latest capabilities and functionality of the standards provided by the  
183 MTConnect Institute and the OPC Foundation.

## 1.2 MTConnect-OPC UA Goals

184 The OPC Unified Architecture for MTConnect Companion Specification is de-  
185 signed with the following goals in mind, in the interest of wide and rapid adoption  
186 by vendors of equipment and software:

- 187     • Incremental adoption: the technical barrier to MTConnect-OPC UA enable-  
188       ment will be greatly reduced with this companion specification and the  
189       source code and binaries available in the MTConnect-OPC UA reference  
190       port.
- 191     • Evolution: MTConnect and OPC UA can incrementally evolve without  
192       jeopardizing backwards compatibility of previous MTConnect-OPC UA ver-  
193       sions.
- 194     • Customizability: MTConnect-OPC UA's extensibility enables integrators to  
195       create value-added software and tools that are machine-specific or installation-  
196       specific, without jeopardizing compatibility with other equipment or soft-  
197       ware.
- 198     • Non-proprietary: built on open standards, backed by both the OPC Founda-  
199       tion and the MTConnect Institute which represents hundreds of companies,  
200       individuals, government organizations and non-profits all working toward  
201       the goal of increased productivity in the manufacturing arena.

## 1.3 Who Will Find Benefit from this Specification?

202 To adopt the OPC Unified Architecture for MTConnect Companion Specification  
203 one will need to have a clear understanding of both MTConnect and OPC UA.  
204 From the technical side, we will discuss MTConnect-OPC UA from:

- 205     • The backend or OPC UA Server and MTConnect agent/adaptor architecture.

- The client or software application side, we will discuss how one develops an application that is MTConnect-OPC UA enabled.
- Applying MTConnect semantics to devices containing an embedded OPC UA Server.

From the business side, we will reference a companion business MTConnect-OPC UA white paper that addresses the concerns from the owners and top management of the business as well as the operations and engineering management. It is the objective of this white paper to provide information primarily to MTConnect and OPC UA software developers. We do not make assumptions about the level of programming expertise beyond what would be considered to be "reasonable" level of expertise. It is for this reason that we include enough details about both MTConnect and OPC UA to provide the ability to implement this companion specification without having references back to other documents. However, the OPC and MTConnect standards are critical and become much more meaningful with the appropriate overview from this document.

## 1.4 References

### 1.4.1 OPC Foundation

The following specifications from the OPC foundation are referenced by this specification.

[UA Part 1] OPC UA Specification: Part 1 – Concepts

<http://www.opcfoundation.org/UA/Part1/>

[UA Part 2] OPC UA Specification: Part 2 – Security Model

<http://www.opcfoundation.org/UA/Part2/>

[UA Part 3] OPC UA Specification: Part 3 – Address Space Model

<http://www.opcfoundation.org/UA/Part3/>

[UA Part 4] OPC UA Specification: Part 4 – Services

<http://www.opcfoundation.org/UA/Part4/>

- 231 [UA Part 5] OPC UA Specification: Part 5 – Information Model  
232 <http://www.opcfoundation.org/UA/Part5/>
- 233 [UA Part 6] OPC UA Specification: Part 6 – Mappings  
234 <http://www.opcfoundation.org/UA/Part6/>
- 235 [UA Part 7] OPC UA Specification: Part 7 – Profiles  
236 <http://www.opcfoundation.org/UA/Part7/>
- 237 [UA Part 8] OPC UA Specification: Part 8 – Data Access  
238 <http://www.opcfoundation.org/UA/Part8/>
- 239 [UA Part 9] OPC UA Specification: Part 9 – Alarms and Conditions  
240 <http://www.opcfoundation.org/UA/Part9/>
- 241 [UA Part 10] OPC UA Specification: Part 10 – Programs  
242 <http://www.opcfoundation.org/UA/Part10/>
- 243 [UA Part 11] OPC UA Specification: Part 11 – Historical Access  
244 <http://www.opcfoundation.org/UA/Part11/>
- 245 [UA Part 13] OPC UA Specification: Part 13 – Aggregates  
246 <http://www.opcfoundation.org/UA/Part13/>

## 1.4.2 MTConnect<sup>®</sup> Institute

- 247 [MT Part 1.0] MTConnect<sup>®</sup> Standard: Part 1.0 – Overview and Fundamentals,  
248 Version 1.4.0  
249 [https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/  
250 t/5acb81f96d2a73d3a01281c5/1523286521969/MTC\\_Part1\\_  
251 0\\_OverviewAndFundamentals1\\_4\\_0.pdf](https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/t/5acb81f96d2a73d3a01281c5/1523286521969/MTC_Part1_0_OverviewAndFundamentals1_4_0.pdf)
- 252 [MT Part 2.0] MTConnect<sup>®</sup> Standard: Part 2.0 – Devices Information Model,  
253 Version 1.4.0  
254 [https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/  
255 t/5acb822e352f53a44f10534b/1523286575786/MTC\\_Part2\\_  
256 0\\_Devices\\_1\\_4\\_0.pdf](https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/t/5acb822e352f53a44f10534b/1523286575786/MTC_Part2_0_Devices_1_4_0.pdf)

257 [MT Part 3.0] MTConnect<sup>®</sup> Standard: Part .0 – Streams Information Model, Ver-  
258 sion 1.4.0  
259 [https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/](https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/t/5acb651e88251b53486ed7f5/1523279135088/MTC_Part3_0_StreamsInformationModel_1_4_0.pdf)  
260 [t/5acb651e88251b53486ed7f5/1523279135088/MTC\\_Part3\\_](https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/t/5acb651e88251b53486ed7f5/1523279135088/MTC_Part3_0_StreamsInformationModel_1_4_0.pdf)  
261 [0\\_StreamsInformationModel\\_1\\_4\\_0.pdf](https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/t/5acb651e88251b53486ed7f5/1523279135088/MTC_Part3_0_StreamsInformationModel_1_4_0.pdf)

262 [MT Part 4.0] MTConnect<sup>®</sup> Standard: Part 4 – Assets Information Model, Version  
263 1.4.0  
264 [https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/](https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/t/5acb824a8a922dc773e19caf/1523286602677/MTC_Part4_0_AssetsInformationModel_1_4_0.pdf)  
265 [t/5acb824a8a922dc773e19caf/1523286602677/MTC\\_Part4\\_](https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/t/5acb824a8a922dc773e19caf/1523286602677/MTC_Part4_0_AssetsInformationModel_1_4_0.pdf)  
266 [0\\_AssetsInformationModel\\_1\\_4\\_0.pdf](https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/t/5acb824a8a922dc773e19caf/1523286602677/MTC_Part4_0_AssetsInformationModel_1_4_0.pdf)

267 [MT Part 4.1] MTConnect<sup>®</sup> Standard: Part 4.1 – Cutting Tools, Version 1.4.0  
268 [https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/](https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/t/5acb825a03ce649b2a64b282/1523286619088/MTC_Part4_1_CuttingTools_1_4_0.pdf)  
269 [t/5acb825a03ce649b2a64b282/1523286619088/MTC\\_Part4\\_](https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/t/5acb825a03ce649b2a64b282/1523286619088/MTC_Part4_1_CuttingTools_1_4_0.pdf)  
270 [1\\_CuttingTools\\_1\\_4\\_0.pdf](https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/t/5acb825a03ce649b2a64b282/1523286619088/MTC_Part4_1_CuttingTools_1_4_0.pdf)

271 [MT Part 5.0] MTConnect<sup>®</sup> Standard: Part 5.0 – Interfaces, Version 1.4.0  
272 [https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/](https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/t/5acb826a352f53a44f10606f/1523286635455/MTC_Part5_0_Interfaces_1_4_0.pdf)  
273 [t/5acb826a352f53a44f10606f/1523286635455/MTC\\_Part5\\_](https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/t/5acb826a352f53a44f10606f/1523286635455/MTC_Part5_0_Interfaces_1_4_0.pdf)  
274 [0\\_Interfaces\\_1\\_4\\_0.pdf](https://static1.squarespace.com/static/54011775e4b0bc1fe0fb8494/t/5acb826a352f53a44f10606f/1523286635455/MTC_Part5_0_Interfaces_1_4_0.pdf)

## 1.5 Abbreviations

275 The following abbreviation are used in this document:

- 276 • ERP – Enterprise Resource Planning
- 277 • HMI – Human Machine Interface
- 278 • HTTP – Hyper Text Transport Protocol
- 279 • MES – Management Execution Systems
- 280 • PLC – Programmable Logic Controller
- 281 • PMS – Production Management Systems

- 282     • SCADA – Supervisory Control And Data Acquisition
- 283     • TCP/IP – Transmission Control Protocol/Internet Protocol
- 284     • XML – eXtensible Mark-up Language

## 2 Use Cases

### 2.1 Overview

285 Before delving into the details of the specification it is useful to identify some  
 286 of the key use cases for the technology. The use cases defined here are not an  
 287 exhaustive list; however, they should help demonstrate how this specification is  
 288 expected to be used and to help illustrate the benefits of a common information  
 289 model.

### 2.2 Device Maker

290 The use case, shown in ??\_use\_casefigure:mfg\_use\_caseters on the manufacturer  
 291 of a piece of equipment or device that needs to provide connectivity to other sys-  
 292 tems. In some cases, the device manufacturer will be targeting markets other than  
 293 equipment (Machine Tool) and would benefit from a more generic specification  
 294 like OPC UA. On the other hand, the standardized semantics of MTConnect are  
 295 extremely important to standardized communications on the manufacturing shop  
 296 floor. The MTConnect-OPC UA specification and the resulting standard infor-  
 297 mation model allows the device manufacturers to standardize on OPC UA as the  
 298 network interface while making their information accessible to software appli-  
 299 cations that includes the enhanced meaning and structure provided by applying  
 300 the MTConnect semantics. Figure 1 shows several clients developed for different  
 301 purposes that can access information produced by the device via OPC UA.

September 29, 2018

## **3 Types**

### **3.1 Components**

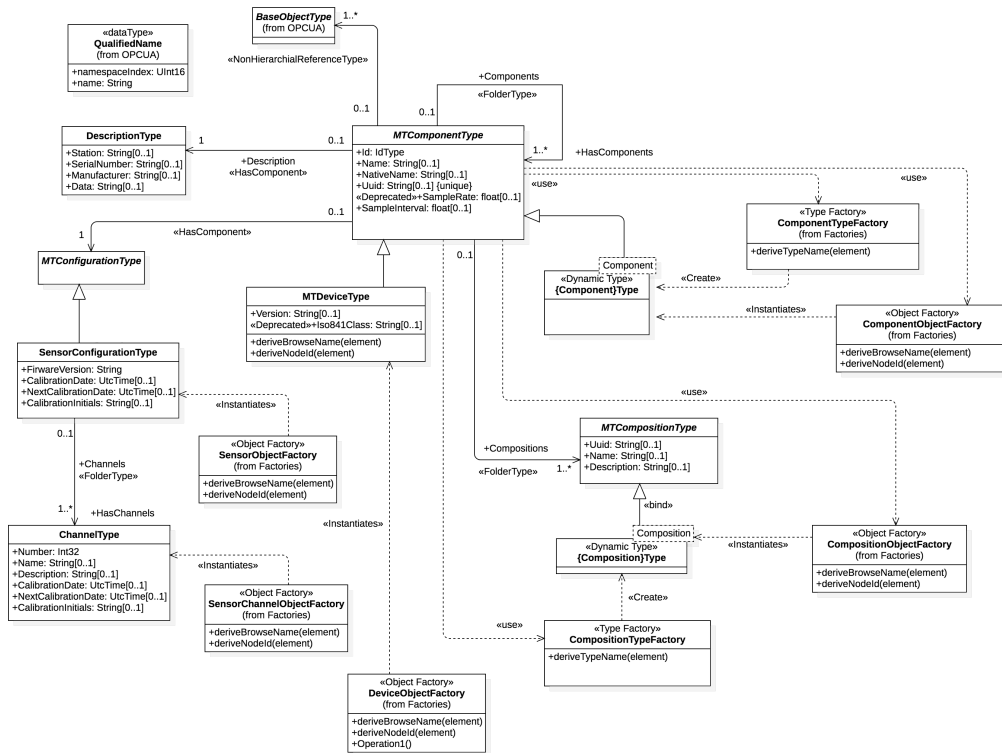


Figure 1: Components Diagram

302 The Components documents the Component models and the owned objects.

### 3.1.1 Defintion of ChannelType

303 An MTConnect Channel is a single data stream associated with a sensor. Each  
304 stream of data can be calibrated separately and allows for the specification of the  
305 meta information and descriptive information. The only required property of the  
306 Channel is the number which is the unique identifier.

307 The channels will be created by the SensorChannelObjectFactory that  
308 composes the BrowseName and the NodeId for each object. (See 3.4.12).



**Table 1:** ChannelType Definition

Attribute	Value				
BrowseName	ChannelType				
IsAbstract	False				
References	NodeClass	BrowseName	Data Type	TypeDefinition	Modeling Rule
Subtype of BaseObjectType (See OPCUA Documentation)					
HasProperty	Variable	Number	Int32	PropertyType	Mandatory
HasProperty	Variable	Name	String	PropertyType	Optional
HasProperty	Variable	MTDescription	String	PropertyType	Optional
HasProperty	Variable	CalibrationDate	UtcTime	PropertyType	Optional
HasProperty	Variable	NextCalibrationDate	UtcTime	PropertyType	Optional
HasProperty	Variable	CalibrationInitials	String	PropertyType	Optional

### 3.1.2 Defintion of DescriptionType

309 The desription provides some general information about the manufacture and se-  
310 rial number of the component. In the XML, the CDATA is freeform text that is  
311 represented in the Data Property of the Description Object. The description is  
312 related to the component with the OPC/UA HasComponent relationship.

**Table 2:** DescriptionType Definition

Attribute	Value				
BrowseName	DescriptionType				
IsAbstract	False				
References	NodeClass	BrowseName	Data Type	TypeDefinition	Modeling Rule
Subtype of BaseObjectType (See OPCUA Documentation)					
HasProperty	Variable	Station	String	PropertyType	Optional
HasProperty	Variable	SerialNumber	String	PropertyType	Optional
HasProperty	Variable	Manufacturer	String	PropertyType	Optional
HasProperty	Variable	Data	String	PropertyType	Optional

### 313 3.1.2.1 Operations

314 • `deriveBrowseName(element)`

315 Specification:

`"Description"`

316 • `deriveNodeId(element)`

317 Specification:

`concat(self.parent.NodeId, BrowseName)`

### 3.1.3 Defintion of **MTComponentType**

318 The base Component Type from which all MTConnect Components are derived.  
 319 The component type factory is used to create the specific OPC/UA Types as sub-  
 320 types of the MTConnect `MTComponentType`. The component types will be  
 321 created once for all Component objects of that type based on the `QName` of the  
 322 MTConnect XML element.

323 The object factory will instantiate the Component Objects and insert them into  
 324 the Components folder with a browse name of the Component `QName` and the  
 325 name element if specified surrounded by square brackets, `[ ]`. For example if the  
 326 MTConnect Element is:

327 `<Linear name='X'>...</...>`

328 The OPC/UA Object with browse name `Linear[X]` will be created with the  
 329 `HasTypeDefinition` referencing the `Linear` OPC/UA type.

330 The meta data for the component and it's relationships are static. The dynamic  
 331 data will be represented using the *OPC/UA Part 8*.

**Table 3:** MTComponentType Definition

Attribute	Value				
BrowseName	MTComponentType				
IsAbstract	True				
References	NodeClass	BrowseName	DataType	TypeDefinition	Modeling Rule
HasProperty	Variable	XmlId	IdType	PropertyType	Mandatory
HasProperty	Variable	Name	String	PropertyType	Optional
HasProperty	Variable	NativeName	String	PropertyType	Optional
HasProperty	Variable	Uuid	String	PropertyType	Optional
HasProperty	Variable	SampleRate	float	PropertyType	Optional
HasProperty	Variable	SampleInterval	float	PropertyType	Optional
HasComponent	Object	Description		DescriptionType	Optional
HasComponent	Object	Configuration		MTConfigurationType	Optional
Organizes	Object	Components	MTComponentType	FolderType	Optional
Organizes	Object	Compositions	MTCompositionType	FolderType	Optional
HasProperty	Variable	<Dynamic>	BaseObjectType	<Dynamic>	Optional
Organizes	Object	Conditions	AlarmConditionType	FolderType	Optional
HasProperty	Variable	<Dynamic>	DataItemType	<Dynamic>	Optional

### 3.1.3.1 Constraints

- Constraint node\_id:

```
context Component::NodeId : String
derive: concat(self.getDevice().uuid, self.getAttributes().id)
```

Documentation: The NodeId SHALL be derived from the combination of the device UUID and the id of the component.

### 3.1.4 Defintion of MTCompositionType

The MTCompositionType is the abstract supertype of the dynamically generated composition types based on the attribute type of the Composition element of the MTConnect Component. The Composition is then related to the DataItems that reference the Composition's id in their compositionId attribute.

**Table 4:** MTCompositionType Definition

Attribute	Value				
BrowseName	MTCompositionType				
IsAbstract	True				
References	NodeClass	BrowseName	Data Type	TypeDefinition	Modeling Rule
Subtype of BaseObjectType (See OPCUA Documentation)					
HasProperty	Variable	Uuid	String	PropertyType	Optional
HasProperty	Variable	Name	String	PropertyType	Optional
HasProperty	Variable	MTDescription	String	PropertyType	Optional
NonHierarchialReferenceType	Object	<DataItem>	DataItem Type	NonHierarchialReferenceType	Optional

341 The data items are added to the relationship where the DataItem to Composition  
342 relationship is represented by the BrowseName Composition property of the data  
343 item the data items are added by their browse names to the Composition.

### 3.1.5 Defintion of MTConfigurationType

344 The abstract MTConfigurationType currently has only one sub-type,  
345 SensorConfigurationType (see 3.1.7). In the future the configurations  
346 will also contain component and device configuration information as sub-types.

**Table 5:** MTConfigurationType Definition

Attribute	Value				
BrowseName	MTConfigurationType				
IsAbstract	True				
References	NodeClass	BrowseName	Data Type	TypeDefinition	Modeling Rule
Subtype of BaseObjectType (See OPCUA Documentation)					

### 3.1.6 Defintion of MTDeviceType

347 The MTDevice is a special type whose object will be the root of the device graph.  
 348 The Device uses the component type factory and the component object factories  
 349 to create each of the first level components.

350 The compositions, relationships, and data items are then recursively created as  
 351 one decendes the MTConnect informaiton model.

**Table 6:** MTDeviceType Definition

Attribute	Value				
BrowseName	MTDeviceType				
IsAbstract	False				
References	NodeClass	BrowseName	Data Type	TypeDefinition	Modeling Rule
Subtype of MTComponentType (see section 3.1.3)					
HasProperty	Variable	Version	String	PropertyType	Optional
HasProperty	Variable	Iso841Class	String	PropertyType	Optional

### 352 **3.1.6.1 Operations**

- 353 • `deriveBrowseName(element)`

354     **Specification:**

`self.name`

- 355 • `deriveNodeId(element)`

356     **Specification:**

`self.uuid`

### 357 **3.1.6.2 Constraints**

- 358 • **Constraint `uuid_not_empty`:**

`uuid->notEmpty()`

359     **Documentation:** The UUID SHALL be provided.

- 360 • **Constraint `name_not_empty`:**

`name->notEmpty()`

361     **Documentation:** The name of the Device SHALL be given.

## **3.1.7 Defintion of `SensorConfigurationType`**

362 The `SensorConfiguration` browse name will be created as an Object relationship  
363 with the parent component.

**Table 7:** SensorConfigurationType Definition

Attribute	Value				
BrowseName	SensorConfigurationType				
IsAbstract	False				
References	NodeClass	BrowseName	DataType	TypeDefinition	Modeling Rule
Subtype of MTConfigurationType (see section 3.1.5)					
HasProperty	Variable	FirmwareVersion	String	PropertyType	Mandatory
HasProperty	Variable	CalibrationDate	UtcTime	PropertyType	Optional
HasProperty	Variable	NextCalibrationDate	UtcTime	PropertyType	Optional
HasProperty	Variable	CalibrationInitials	String	PropertyType	Optional
Organizes	Object	Channels	ChannelType	FolderType	Optional

### 3.1.8 Defintion of {Component} Type

**Table 8:** {Component}Type Definition

Attribute	Value				
BrowseName	ComponentType				
IsAbstract	False				
References	NodeClass	BrowseName	DataType	TypeDefinition	Modeling Rule
Subtype of MTComponentType (see section 3.1.3)					

### 3.1.9 Defintion of {Composition}Type

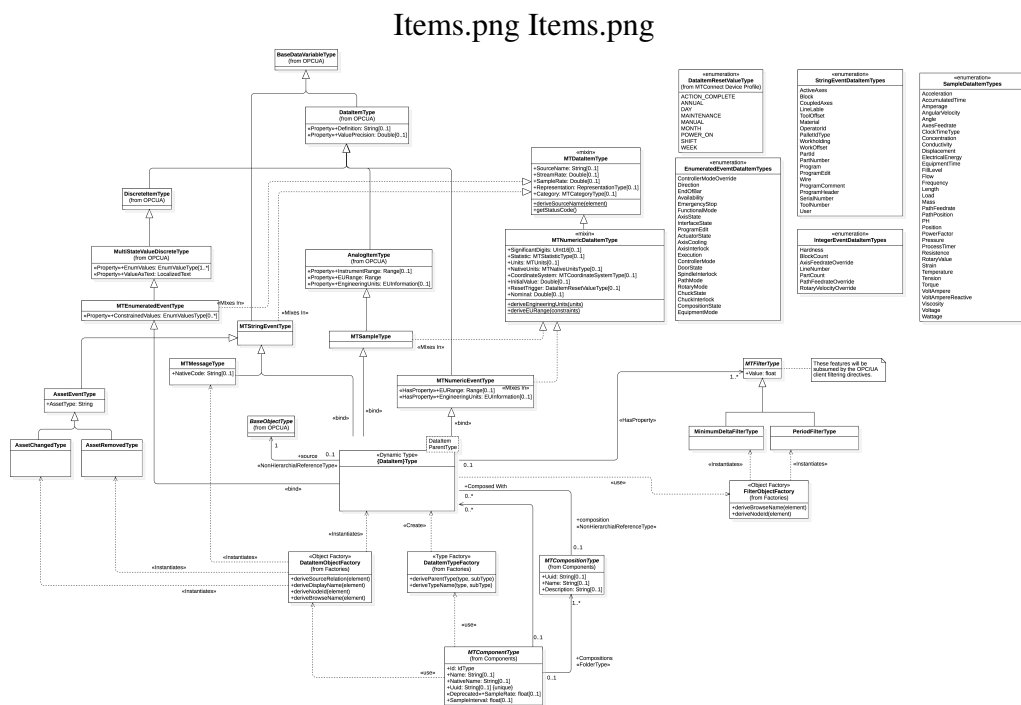
**Table 9:** {Composition}Type Definition

Attribute	Value				
BrowseName	CompositionType				
IsAbstract	False				
References	NodeClass	BrowseName	Data Type	TypeDefinition	Modeling Rule
Subtype of MTCompositionType (see section 3.1.4)					



September 29, 2018

## **3.2 Data Items**



### 3.2.1 Defintion of AssetChangedType

**Table 10:** AssetChangedType Definition

Attribute	Value				
BrowseName	AssetChangedType				
IsAbstract	False				
References	NodeClass	BrowseName	Data Type	TypeDefinition	Modeling Rule
Subtype of AssetEventType (see section 3.2.2)					

### 3.2.2 Defintion of AssetEventType

**Table 11:** AssetEventType Definition

Attribute	Value				
BrowseName	AssetEventType				
IsAbstract	False				
References	NodeClass	BrowseName	Data Type	TypeDefinition	Modeling Rule
Subtype of MTStringEventType (see section 3.2.11)					
HasProperty	Variable	AssetType	String	PropertyType	Mandatory

### 3.2.3 Defintion of AssetRemovedType

**Table 12:** AssetRemovedType Definition

Attribute	Value				
BrowseName	AssetRemovedType				
IsAbstract	False				
References	NodeClass	BrowseName	Data Type	TypeDefinition	Modeling Rule
Subtype of AssetEventType (see section 3.2.2)					

### **3.2.4 Defintion of MTDataItemType**

364 The data item mixin will inject the properties and the methods into the related  
365 classes. This facility is similar to the Ruby module mixin or the Scala traits.

**Table 13:** MTDataItemType Definition

Attribute	Value				
BrowseName	MTDataItemType				
IsAbstract	False				
References	NodeClass	BrowseName	Data Type	TypeDefinition	Modeling Rule
HasProperty	Variable	SourceName	String	PropertyType	Optional
HasProperty	Variable	StreamRate	Double	PropertyType	Optional
HasProperty	Variable	SampleRate	Double	PropertyType	Optional
HasProperty	Variable	Representation	RepresentationType	PropertyType	Optional
HasProperty	Variable	Category	MTCategoryType	PropertyType	Mandatory
HasProperty	Variable	<Dynamic>	MTFilterType	<Dynamic>	Optional
HasComponent	Object	source		BaseObjectType	Optional

### 3.2.4.1 Operations

- `deriveSourceName(element)`

Specification:

```
self.Source.CDATA
```

Documentation: Derive the source name from the Source element CDATA.

This will represent the alternative long name for the data item's source.

- `getStatusCode()`

Documentation: The OPC/UA status code will be created using the following process:

- If the value of the data item is UNAVAILABLE a status code of `Uncertain_NoCommunicationLastUsable`
- When a reset trigger is specified, new `Good_` status codes will be created. See `ResetTrigger` enumeration.

### 3.2.5 Defintion of MTEnumeratedEventType

All Data Items with Category EVENT having a Controlled Vocabularies will be of this type. Otherwise, MTString

**Table 14:** MTEnumeratedEventType Definition

Attribute	Value				
BrowseName	MTEnumeratedEventType				
IsAbstract	False				
References	NodeClass	BrowseName	Data Type	TypeDefinition	Modeling Rule
Subtype of MultiStateValueDiscreteType (See OPCUA Documentation)					
HasProperty	Variable	ConstrainedValues	EnumValuesType	PropertyType	Mandatory

### 3.2.6 Defintion of MTFilterType

380 These features will be subsumed by the OPC/UA client filtering directives.

**Table 15:** MTFilterType Definition

Attribute	Value				
BrowseName	MTFilterType				
IsAbstract	True				
References	NodeClass	BrowseName	Data Type	TypeDefinition	Modeling Rule
HasProperty	Variable	Value	float	PropertyType	Mandatory



### 381 3.2.6.1 Operations

- 382 • deriveBrowseName(element)

383 Specification:

concat(parent.BrowseName, pascalCase(element.type))

- 384 • deriveNodeId(element)

385 Specification:

concat(parent.NodeId, pascalCase(element.type))

## 3.2.7 Defintion of MTMessageType

**Table 16:** MTMessageType Definition

Attribute	Value				
BrowseName	MTMessageType				
IsAbstract	False				
References	NodeClass	BrowseName	DataType	TypeDefinition	Modeling Rule
Subtype of MTStringEventType (see section 3.2.11)					
HasProperty	Variable	NativeCode	String	PropertyType	Optional

### **3.2.8 Defintion of MTNumericDataItemType**

386 These are the additional attributes that are relevent to numeric data items. The  
387 factory will evaluate these values and will set the engineering units and the range  
388 associated with the parent entity.

**Table 17:** MTNumericDataItemType Definition

Attribute	Value				
BrowseName	MTNumericDataItemType				
IsAbstract	False				
References	NodeClass	BrowseName	Data Type	TypeDefinition	Modeling Rule
Subtype of MTDataItemType (see section 3.2.4)					
HasProperty	Variable	SignificantDigits	UInt16	PropertyType	Optional
HasProperty	Variable	Statistic	MTStatisticType	PropertyType	Optional
HasProperty	Variable	Units	MTUnits	PropertyType	Optional
HasProperty	Variable	NativeUnits	MTNativeUnitsType	PropertyType	Optional
HasProperty	Variable	CoordinateSystem	MTCordinateSystemType	PropertyType	Optional
HasProperty	Variable	InitialValue	Double	PropertyType	Optional
HasProperty	Variable	ResetTrigger	DataItemResetValueType	PropertyType	Optional
HasProperty	Variable	Nominal	Double	PropertyType	Optional

### 3.2.8.1 Operations

- `deriveEngineeringUnits(units)`

Specification:

```
EngineeringUnits <- self.units
```

- `deriveEURange(constraints)`

Specification:

```
EURange.Low <- self.Constraints.Minimum
EURange.High <- self.Constraints.Maximum
```

Documentation: Uses the MTConnect Constraints element if present to derive the minimum and maximum values for the numeric values. This applies to both the Numeric Event and the Sample types.

## 3.2.9 Defintion of MTNumericEventType

All data items with category EVENT and a numeric value.

**Table 18:** MTNumericEventType Definition

Attribute	Value				
BrowseName	MTNumericEventType				
IsAbstract	False				
References	NodeClass	BrowseName	Data Type	TypeDefinition	Modeling Rule
Subtype of DataItemType (See OPCUA Documentation)					
HasProperty	Variable	EURange	Range	PropertyType	Optional
HasProperty	Variable	EngineeringUnits	EUInformation	PropertyType	Optional

### 3.2.10 Defintion of MTSampleType

398 Data Items with category SAMPLE

**Table 19:** MTSampleType Definition

Attribute	Value				
BrowseName	MTSampleType				
IsAbstract	False				
References	NodeClass	BrowseName	Data Type	TypeDefinition	Modeling Rule
Subtype of AnalogItemType (See OPCUA Documentation)					

### 3.2.11 Defintion of MTStringEventType

399 All data items with category EVENT where the data is freeform text. The set\_  
 400 data\_type constraint derives makes the data type a string for this type.

**Table 20:** MTStringEventType Definition

Attribute	Value				
BrowseName	MTStringEventType				
IsAbstract	False				
References	NodeClass	BrowseName	Data Type	TypeDefinition	Modeling Rule
Subtype of BaseDataVariableType (See OPCUA Documentation)					

### 401 3.2.11.1 Constraints

- 402     • Constraint `set_data_type`:
- ```
derive: DataType <-String
```

## 3.2.12 Defintion of `MinimumDeltaFilterType`

**Table 21:** `MinimumDeltaFilterType` Definition

| Attribute                                               | Value                  |            |          |                |               |
|---------------------------------------------------------|------------------------|------------|----------|----------------|---------------|
| BrowseName                                              | MinimumDeltaFilterType |            |          |                |               |
| IsAbstract                                              | False                  |            |          |                |               |
| References                                              | NodeClass              | BrowseName | DataType | TypeDefinition | Modeling Rule |
| Subtype of <code>MFilterType</code> (see section 3.2.6) |                        |            |          |                |               |

### 3.2.13 Defintion of PeriodFilterType

**Table 22:** PeriodFilterType Definition

| Attribute                                   | Value            |            |           |                |               |
|---------------------------------------------|------------------|------------|-----------|----------------|---------------|
| BrowseName                                  | PeriodFilterType |            |           |                |               |
| IsAbstract                                  | False            |            |           |                |               |
| References                                  | NodeClass        | BrowseName | Data Type | TypeDefinition | Modeling Rule |
| Subtype of MTFilterType (see section 3.2.6) |                  |            |           |                |               |

### 3.2.14 Definition of {DataItem}Type

403 For each DataItem the Sub Type, and the Type will be composed to be the HasTypeDefinition  
 404 relationship of the object. The BrowseName will also include the Composition  
 405 Type if a composition Id is provided.

**Table 23:** {DataItem}Type Definition

| Attribute                                         | Value        |            |           |                |               |
|---------------------------------------------------|--------------|------------|-----------|----------------|---------------|
| BrowseName                                        | DataItemType |            |           |                |               |
| IsAbstract                                        | False        |            |           |                |               |
| References                                        | NodeClass    | BrowseName | Data Type | TypeDefinition | Modeling Rule |
| Subtype of MTNumericEventType (see section 3.2.9) |              |            |           |                |               |



September 29, 2018

### **3.3 Conditions**

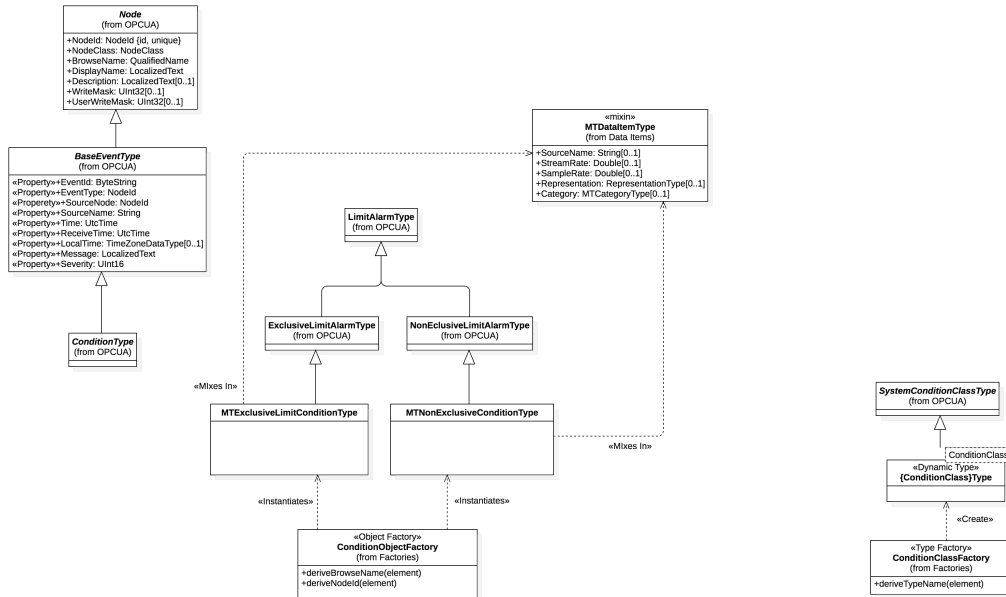


Figure 3: Conditions Diagram

### 3.3.1 Defintion of MTEExclusiveLimitConditionType

Table 24: MTEExclusiveLimitConditionType Definition

| Attribute                                                    | Value                         |            |           |                |               |
|--------------------------------------------------------------|-------------------------------|------------|-----------|----------------|---------------|
| BrowseName                                                   | MTExclusiveLimitConditionType |            |           |                |               |
| IsAbstract                                                   | False                         |            |           |                |               |
| References                                                   | NodeClass                     | BrowseName | Data Type | TypeDefinition | Modeling Rule |
| Subtype of ExclusiveLimitAlarmType (See OPCUA Documentation) |                               |            |           |                |               |

### 3.3.2 Defintion of MTNonExclusiveConditionType

**Table 25:** MTNonExclusiveConditionType Definition

| Attribute                                                      | Value                       |            |           |                |               |
|----------------------------------------------------------------|-----------------------------|------------|-----------|----------------|---------------|
| BrowseName                                                     | MTNonExclusiveConditionType |            |           |                |               |
| IsAbstract                                                     | False                       |            |           |                |               |
| References                                                     | NodeClass                   | BrowseName | Data Type | TypeDefinition | Modeling Rule |
| Subtype of NonEclusiveLimitAlarmType (See OPCUA Documentation) |                             |            |           |                |               |

### 3.3.3 Defintion of {ConditionClass}Type

**Table 26:** {ConditionClass}Type Definition

| Attribute                                                     | Value              |            |           |                |               |
|---------------------------------------------------------------|--------------------|------------|-----------|----------------|---------------|
| BrowseName                                                    | ConditionClassType |            |           |                |               |
| IsAbstract                                                    | False              |            |           |                |               |
| References                                                    | NodeClass          | BrowseName | Data Type | TypeDefinition | Modeling Rule |
| Subtype of SystemConditionClassType (See OPCUA Documentation) |                    |            |           |                |               |

### 3.4 Factories

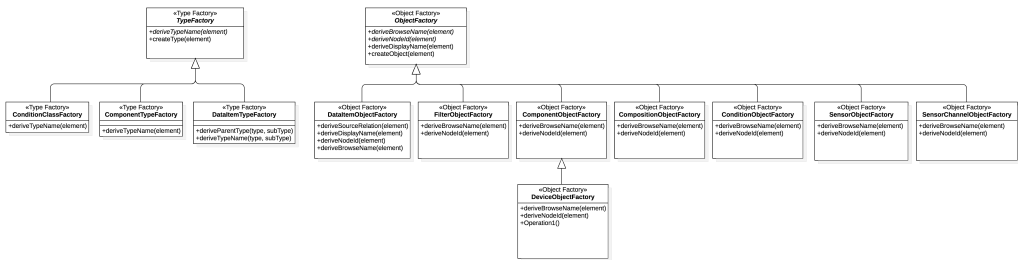


Figure 4: Factories Diagram

406 The factories are not part of the OPC/UA information model. They are a set  
 407 of helper classes that are used to create dynamic types and objects. Since the  
 408 MTConnect information model can be layered on top of the OPC/UA abstractions,  
 409 the factories provide the rules for creating the browse and display names for each  
 410 type.

411 The factories also create dynamic objects when required for variables of various  
 412 classes when they are required, such as the Data Items and the Components. Some  
 413 of the relationships are more complex since they require a dynamic super-type  
 414 relationship that relies on the correct placement of the MTConnect elements to be  
 415 correctly represented using the OPC/UA base types.

416 This is especially evident when mapping the DataItems and the Conditions to the  
 417 MTConnect Information Models and providing sufficient definition to allow for  
 418 unambiguous implementation.

### 3.4.1 Defintion of «Object Factory» ComponentObjectFactory

#### 419 3.4.1.1 Operations

420 • deriveBrowseName(element)

421 Specification:

```
concat(element.QName, (if self.name.notEmpty() then concat('[', self.name
```

422 • deriveNodeId(element)

423 Specification:

```
concat(self.findDevice().uuid, element.id)
```

### 3.4.2 Defintion of «Type Factory» ComponentTypeFactory

424 The ‘ComponentTypeFactory’ creates component types using the MTConnect  
 425 XML element as an input. The factory takes the ‘QName’ (or qualified name)  
 426 of the XML element and then appends ‘Type’. For example an ‘<Controller

427 `id='...'></...>` element will create an OPC/UA `'ControllerType'` type definition  
 428 as an extension of the base `'MTControllerType'`.

429 Currently there is no additional abstractions or super types required by the com-  
 430 panion specification. The types will be a single level where each Component is a  
 431 sub-type of the base `'MTComponentType'`.

### 432 **3.4.2.1 Operations**

433 • `deriveTypeName(element)`

434 Specification:

```
derive: Component <- element.QName
```

435 Documentation: The QName of the element for the component will be used  
 436 to derive the type of the node.

## **3.4.3 Defintion of «Object Factory» CompositionObjectFactory**

### 437 **3.4.3.1 Operations**

438 • `deriveBrowseName(element)`

439 Specification:

```
concat(pascalCase(element.type), (if self.name.notEmpty() then concat('['
```

440 • `deriveNodeId(element)`

441 Specification:

```
concat(self.findDevice().uuid, element.id)
```

## **3.4.4 Defintion of «Type Factory» CompositionTypeFactory**

### 442 **3.4.4.1 Operations**

443     • `deriveTypeName(element)`

444     Specification:

```
derive: Composition <-  
  pascalCase(element.type)
```

445     Documentation: The type for the composition will be created using the pas-  
446     cal case of the 'type' from the composition element.

### 3.4.5 Defintion of «Type Factory» `ConditionClassFactory`

#### 447 3.4.5.1 Operations

448     • `deriveTypeName(element)`

449     Documentation: Create condition classes based on the OPC/UA three con-  
450     dition types.

### 3.4.6 Defintion of «Object Factory» `ConditionObjectFactory`

#### 451 3.4.6.1 Operations

452     • `deriveBrowseName(element)`

453     • `deriveNodeId(element)`

### 3.4.7 Defintion of «Object Factory» `DataItemObjectFactory`

#### 454 3.4.7.1 Operations

455     • `deriveSourceRelation(element)`

456     Documentation: Use the source composition, component id, or data item id  
457     to locate the source node id for this relationship. If one exists, add an object  
458     with browse name "source" that relates to the entity referenced by the id.



The most specific identity should be used in the following order:

- DataItemId
- CompositionId
- ComponentId

Since the data item implies composition and component and the composition implies component, there should only be one attribute given for the source.

- `deriveDisplayName(element)`

Documentation: Same as the `BrowseName`.

- `deriveNodeId(element)`

Documentation: The `nodeId` will be given by the device uuid and the `DataItemId` attribute.

- `deriveBrowseName(element)`

Documentation: The browse name will be composed of the following parts of the model:

1. If the `compositionId` is present, the `compositionId` will be resolved the the `Composition` element and the pascal case of the type attribute will be placed first.
2. If the `subType` is present, the pascal case of the `subType` will be placed next.
3. The pascal case of the type will be placed last.

For example, for a data item with the following attributes:

- `type`: `TEMPERATURE`
- `composition type`: `STORAGE_BATTERY`

will have the following browse name: `StorageBatteryTemperature`

For the data item with the following attributes:

- `type`: `ANGLE`
- `subType`: `ACTUAL`

487           – composition type: ENCODER  
 488           will have the following browse name: EncoderActualAngle

### 3.4.8 Defintion of «Type Factory» **DataItemTypeFactory**

489 Based on the data item category, type, and subType, this class creates a new  
 490 OPC/UA type and also provides the template parameter for the ParentType from  
 491 which this type is derived.

#### 492 3.4.8.1 Operations

493     • `deriveParentType(type, subType)`  
 494       Documentation: The parent type is derived from the category as follows:

495       – SAMPLE -> SampleType  
 496       – EVENT ->

497           \* Enumerated Value -> MTEnumeratedEventType  
 498           \* Integer Value -> MTNumericEventType  
 499           \* Otherwise -> MTStringEventType

500     • `deriveTypeName(type, subType)`  
 501       Specification:

`concat(pascalCase(subType), pascalCase(type))`

502       Documentation: Used to derive the class name for creating a pascal case  
 503       name from the sub type and the type. For example type ROTARY\_VELOCITY  
 504       and subType ACTUAL will become ActualRotaryVelocity.

### 3.4.9 Defintion of «Object Factory» **DeviceObjectFactory**

505 The model instantiation for MTConnect begins with the ‘Device‘ MTConnect  
 506 element and then recursively traverses the sub-elements. The device will the ca-

507 pabilities in the component factory to generate all the data items and component  
508 types.

#### 509 **3.4.9.1 Operations**

510 • `deriveBrowseName(element)`

511     **Specification:**

`derive: element.name`

512 • `deriveNodeId(element)`

513     **Specification:**

`derive: element.uuid`

### **3.4.10 Defintion of «Object Factory» FilterObjectFactory**

514 Creates filters based on the type attribute of the Filter element.

#### 515 **3.4.10.1 Operations**

516 • `deriveBrowseName(element)`

517 • `deriveNodeId(element)`

518     **Documentation:** The node id is composed of the data item id and the browse  
519     name.

### **3.4.11 Defntion of «Object Factory» ObjectFactory**

#### 520 **3.4.11.1 Operations**

521 • `deriveBrowseName(element)`

- 522     • `deriveNodeId(element)`
- 523     • `deriveDisplayName(element)`
- 524         Specification:
  - `deriveBrowseName(element)`
- 525     • `createObject(element)`

### 3.4.12 Defintion of «Object Factory» `SensorChannelObjectFactory`

#### 526 3.4.12.1 Operations

- 527     • `deriveBrowseName(element)`
- 528         Specification:
  - `concat('Channel', self.number)`
- 529     • `deriveNodeId(element)`
- 530         Specification:
  - `concat(self.parent.NodeId, BrowseName)`

### 3.4.13 Defintion of «Object Factory» `SensorObjectFactory`

#### 531 3.4.13.1 Operations

- 532     • `deriveBrowseName(element)`
- 533         Specification:
  - `element.QName`
- 534     • `deriveNodeId(element)`
- 535         Specification:
  - `concat(self.parent.NodeId, BrowseName)`

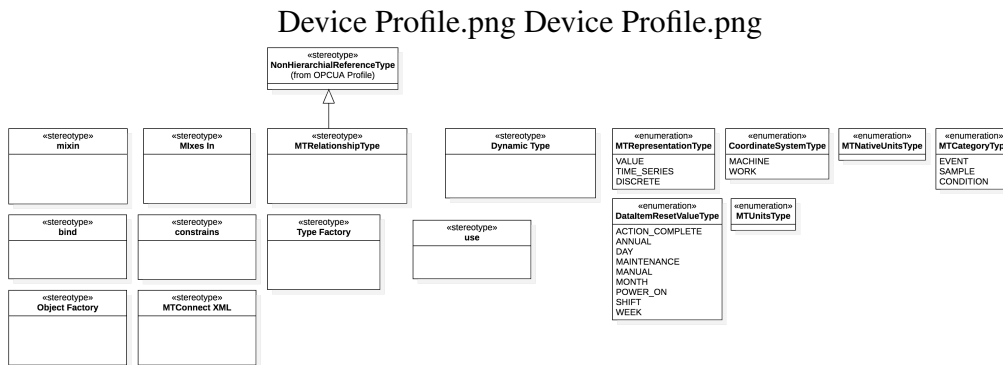


Figure 5: MTConnect Device Profile Diagram

### 3.4.14 Defintion of «Type Factory» TypeFactory

#### 536 3.4.14.1 Operations

- 537     • `deriveTypeName(element)`
- 538     • `createType(element)`

## 3.5 MTConnect Device Profile

539 The device profile documents the common data types and stereotypes that are used  
 540 to construct the model. A stereotype is a design or modeling pattern that provides  
 541 additional information about the type or the relationship between types.

542 It can also identify the behavior of a property or the role the type or relation will  
 543 play in the model.

544 Stereotypes are used throughout the model to provide additional information that  
 545 will halp provide context and definition to aid in better understanding the data  
 546 model.

### **3.5.1 Defintion of Dynamic Type**

### **3.5.2 Defintion of MTConnect XML**

### **3.5.3 Defintion of MTRelationshipType**

### **3.5.4 Defintion of Mixes In**

547 This stereotype is associated with the dependency between a type and a mixin.  
548 See Section 3.5.9 for a complete description of the mixin.

### **3.5.5 Defintion of Object Factory**

### **3.5.6 Defintion of Type Factory**

### **3.5.7 Defintion of bind**

549 When a dynamic type (See Section 3.5.1) creates an instance where the super-type  
550 can be associated based on the data item category and type, the Type Factory  
551 will specify which supertype is to be referenced.

552 The bind stereotype indicates the relationship between the dynamic sub-type and  
553 the parent type are resolved baed on the MTConnect DataItem meta data.

### **3.5.8 Defintion of constrains**

### **3.5.9 Defintion of mixin**

554 The mixin pattern injects the properties and operations into the types that are  
555 related to the using the Mixes In dependency. Mixins allow for lightweight

556 multiple inheritance. Since OPC/UA does not allow for multiple inheritance and  
557 the MTConnect types require the same set of properties when they are sub-typed  
558 from existing OPC/UA types, this mechanism allows for this relationship to be  
559 expressed.

### **3.5.10 Defintion of use**

560 The use stereotype indicates that one class uses as a helper to perform a specific  
561 operation or activity. This stereotype is mainly used to indicate that a specific  
562 factory is being employed by another type to create dynamic properties or rela-  
563 tionships.