MTConnect® Standard
Part 2.0 - Devices Information Model

Version 1.4.0

Prepared for: MTConnect Institute

Prepared on: March 31, 2018

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

MTConnect® Specification and Materials

AMT - The Association For Manufacturing Technology ("AMT") owns the copyright in this MTConnect® Specification or Material. AMT grants to you a non-exclusive, non- transferable, revocable, non-sublicensable, fully-paid-up copyright license to reproduce, copy and redistribute this MTConnect® Specification or Material, provided that you may only copy or redistribute the MTConnect® Specification or Material in the form in which you received it, without modifications, and with all copyright notices and other notices and disclaimers contained in the MTConnect® Specification or Material.

If you intend to adopt or implement an MTConnect® Specification or Material in a product, whether hardware, software or firmware, which complies with an MTConnect® Specification, you shall agree to the MTConnect® Specification Implementer License Agreement ("Implementer License") or to the MTConnect® Intellectual Property Policy and Agreement ("IP Policy"). The Implementer License and IP Policy each sets forth the license terms and other terms of use for MTConnect® Implementers to adopt or implement the MTConnect® Specifications, including certain license rights covering necessary patent claims for that purpose. These materials can be found at www.MTConnect.org or by contacting info@MTConnect.org

MTConnect® Institute and AMT have no responsibility to identify patents, patent claims or patent applications which may relate to or be required to implement a Specification, or to determine the legal validity or scope of any such patent claims brought to their attention. Each MTConnect® Implementer is responsible for securing its own licenses or rights to any patent or other intellectual property rights that may be necessary for such use, and neither AMT nor MTConnect® Institute have any obligation to secure any such rights.

This Material and all MTConnect® Specifications and Materials are provided "as is" and MTConnect® Institute and AMT, and each of their respective members, officers, affiliates, sponsors and agents, make no representation or warranty of any kind relating to these materials or to any implementation of the MTConnect® Specifications or Materials in any product, including, without limitation, any expressed or implied warranty of non-infringement, merchantability, or fitness for particular purpose, or of the accuracy, reliability, or completeness of information contained herein. In no event shall MTConnect® Institute or AMT be liable to any user or implementer of MTConnect® Specifications or Materials for the cost of procuring substitute goods or services, lost profits, loss of use, loss of data or any incidental, consequential, indirect, special or punitive damages or other direct damages, whether under contract, tort, warranty or otherwise, arising in any way out of access, use or inability to use the MTConnect® Specification or other MTConnect® Materials, whether or not they had advance notice of the possibility of such damage.

Table of Contents

Table of Figures

1 Purpose of This Document

This document, *Part 2.0 – Devices Information Model* of the MTConnect® Standard, establishes the rules and terminology to be used by designers to describe the function and operation of a piece of equipment and to define the data that is provided by an *MTConnect Agent* from the equipment. The *Devices Information Model* also defines the structure for the XML document that is returned from a *MTConnect Agent* in response to a *Probe Request*.

In the MTConnect Standard, *equipment* represents any tangible property that is used in the operations of a manufacturing facility. Examples of *equipment* are machine tools, ovens, sensor units, workstations, software applications, and bar feeders.

Note: See *Part 3.0 – Streams Information Model* of the MTConnect Standard for details on the XML documents that are returned from a *MTConnect Agent* in response to a *Sample* or *Current* Request.

2 Terminology and Conventions

Refer to *Section 2* of *Part 1.0 – Overview and Functionality* for a dictionary of terms, reserved language, and document conventions used in the MTConnect Standard.

3 Devices Information Model

The *Devices Information Model* represents the physical and logical configuration for a piece of equipment used for a manufacturing process or for any other purpose. It also provides the definition of data that may be reported by that equipment.

Using information defined in the *Devices Information Model*, a software application can determine the configuration and reporting capabilities of a piece of equipment. To do this, the software application issues a *Probe Request* (defined in *Section 8.1.1* of *Part 1.0 – Overview and Functionality* of the MTConnect Standard) to a *MTConnect® Agent* associated with a piece of equipment. A *MTConnect Agent* responds to the *Probe Request* with an `MTConnectDevices` XML document that contains information describing both the physical and logical structure of the piece of equipment and a detailed description of each *Data Entity* that can be reported by the *Agent* associated with the piece of equipment. This information allows the client software application to interpret the document and to extract the data with the same meaning, value, and context that it had at its original source.

The `MTConnectDevices` XML document is comprised of two sections: `Header` and `Devices`.

The `Header` section contains protocol related information as defined in *Section 6.5.1* of *Part 1.0 – Overview and Functionality* of the MTConnect Standard.

The `Devices` section of the `MTConnectDevices` document contains a `Device` XML container for each piece of equipment described in the document. Each `Device` container is comprised of two primary types of XML elements – *Structural Elements* and *Data Entities*.

Structural Elements are defined as XML elements that organize information that represents the physical and logical parts and sub-parts of a piece of equipment (See *Section 4* of this document for more details).

Data Entities are defined as XML elements that describe data that can be reported by a piece of equipment. In the *Devices Information Model*, *Data Entities* are defined as `DataItem` elements (See *Section 7* and *8* of this document).

The *Structural Elements* and *Data Entities* in the `MTConnectDevices` document provide information representing the physical and logical structure for a piece of equipment and the types of data that the piece of equipment can report relative to that structure. The `MTConnectDevices` document does not contain values for the data types reported by the piece of equipment. The `MTConnectStreams` document defined in *Part 3.0 – Streams Information Model* provides the data values that are reported by the piece of equipment. As such, most *Structural Elements* and *Data Entities* in the `MTConnectDevices` document do not contain `CDATA`. XML elements that provide values or information in the CDATA will be specifically identified in *Sections 4*, *7*, and *9* of this document.

Note: The MTConnect Standard also defines the information model for `Assets`. An *Asset* is something that is used in the manufacturing process, but is not permanently associated with a single piece of equipment, can be removed from the piece of equipment without compromising its function, and can be associated with other pieces of equipment during its lifecycle. See *Part 4.0 – Assets* of the MTConnect Standard for more details on *Assets*.

4 *Structural Elements* for MTConnectDevices

Structural Elements are XML elements that form the logical structure for the MTConnectDevices XML document. These elements are used to organize information that represents the physical and logical architecture of a piece of equipment. Refer to *Figure 1* below for an overview of the *Structural Elements* used in an MTConnectDevices document.

A variety of *Structural Elements* are defined to describe a piece of equipment. Some of these elements **MUST** always appear in the MTConnectDevices XML document, while others are optional and **MAY** be used, as required, to provide additional structure.

The first, or highest level, *Structural Element* in a MTConnectDevices XML document is Devices. Devices is a container type XML element used to group one or more pieces of equipment into a single XML document. Devices **MUST** always appear in the MTConnectDevices document.

Device is the next *Structural Element* in the MTConnectDevices XML document. Device is also a container type XML element. A separate Device container is used to identify each piece of equipment represented in the MTConnectDevices document. Each Device container provides information on the physical and logical structure of the piece of equipment and the data associated with that equipment. Device can also represent any logical grouping of pieces of equipment that function as a unit or any other data source that provides data through a *MTConnect Agent*.

One or more Device element(s) **MUST** always appear in an MTConnectDevices document.

Components is the next *Structural Element* in the MTConnectDevices XML document. Components is also a container type XML element. Components is used to group information describing *Lower Level* physical parts or logical functions of a piece of equipment.

If the Components container appears in the XML document, it **MUST** contain one or more Component type XML elements.

Component is the next level of *Structural Element* in the MTConnectDevices XML document. Component is both an abstract type XML element and a container type element.

As an abstract type element, Component will never appear in the XML document describing a piece of equipment and will be replaced by a specific Component type defined in *Section 5*. Each Component type is also a container type element. As a container, the Component type element is used to organize information describing *Lower Level Structural Elements* or *Data Entities* associated with the Component.

If *Lower Level Structural Elements* are described, these elements are by definition child Component elements of a parent Component. At this next level, the *Lower Level* child Component elements are grouped into an XML container called Components.

This *Lower Level* `Components` container is comprised of one or more child `Component` XML elements representing the sub-parts of the parent `Component`. Just like the parent `Component` element, the child `Component` element is an abstract type XML element and will never appear in the XML document – only the different *Lower Level* child `Component` types will appear.

This parent-child relationship can continue to any depth required to fully define a piece of equipment.

The following example is an XML document structure that demonstrates the relationship between a parent `Component` and *Lower Level* child components:

```
<Devices>
  <Device>
    <Components>
      <Axes>   (Parent component)
        <Components>
          <Rotary>   (Child component to Axes and Parent component to
                      Lower Level components)
            <Components>
              <Chuck>   (Child component to Rotary)
```

The following XML Tree demonstrates the various *Structural Elements* provided to describe a piece of equipment and the relationship between these elements.

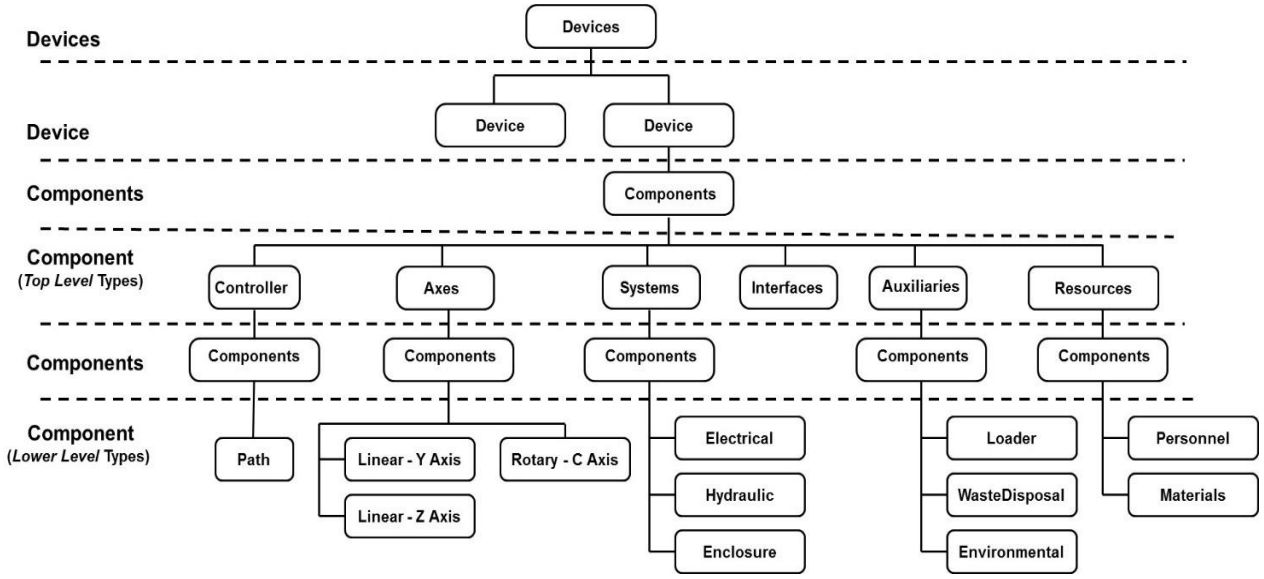


Figure 1: Example `Device` *Structural Elements*

`Component` type XML elements **MAY** be further decomposed into `Composition` type XML elements. `Composition` elements describe the lowest level basic structural or functional building blocks contained within a `Component`. Any number of `Composition` elements **MAY** be used. Data provided for a `Component` provides more specific meaning when it is associated with one of the `Composition` elements of the `Component`. The different `Composition` types that **MAY** appear in the XML document are defined in *Section 6*.

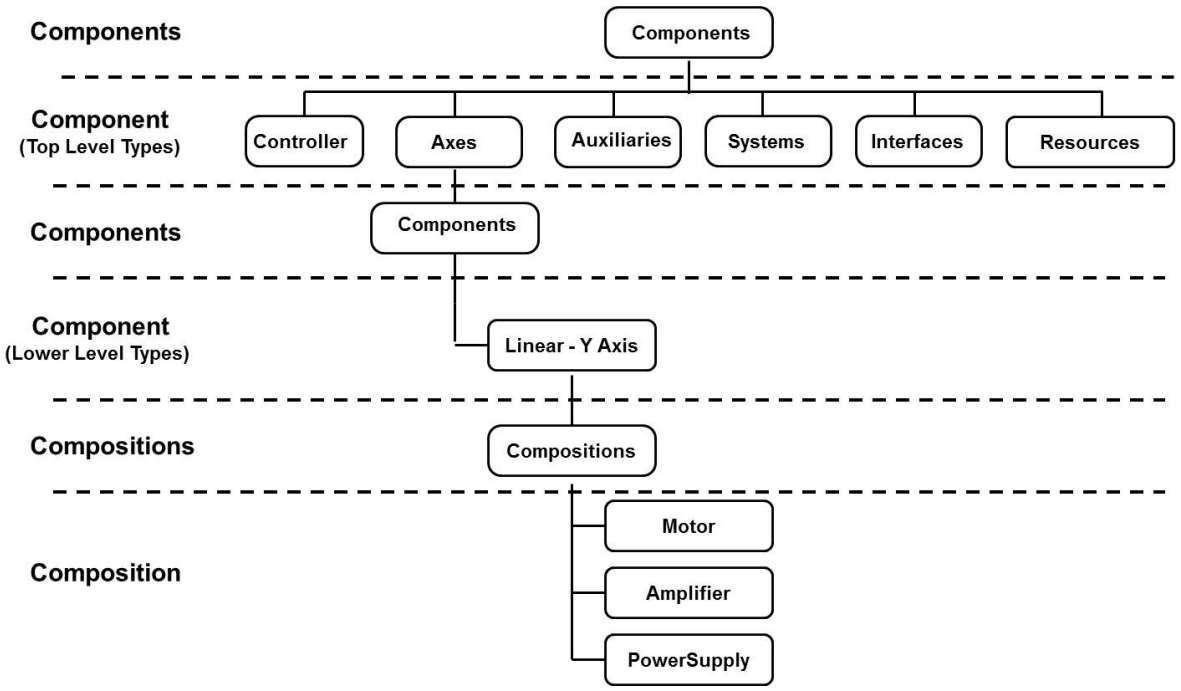
The `Composition` elements are organized into a `Compositions` container. The `Compositions` container **MAY** appear in the XML document further describing a `Component`. If one or more `Composition` element(s) is provided to describe a `Component`, a `Compositions` container **MUST** be defined for the `Component`.

The following illustration represents an XML document structure that demonstrates the relationship between a parent `Component` and its `Composition` elements:

```
1.  <Devices>
2.   <Device>
3.     <Components>
4.      <Axes> (Component)
5.       <Components>
6.         <Linear> (Component)
7.          <Compositions>
8.            <Composition>
9.            <Composition>
10.           <Composition>
```

147 The following XML Tree demonstrates this relationship between a `Component` and some of its
148 potential `Composition` elements.

149



150

151

152 **Figure 2: Example `Composition` *Structural Elements***

153

154 ## 4.1 Devices

155 `Devices` is a container type XML element that **MUST** contain only `Device` elements.
156 `Devices` **MUST** contain at least one `Device` element, but **MAY** contain multiple `Device`
157 elements. *Data Entities* **MAY NOT** be directly associated with the `Devices` container.

Element	Description	Occurrence
`Devices`	The first, or highest level, *Structural Element* in a `MTConnectDevices` document. `Devices` is a container type XML element.	1

158

4.2 Device

`Device` is an XML container type element that organizes the *Structural Elements* and *Data Entities* associated with a piece of equipment. *Data Entities* **MAY** be directly associated with the `Device` container. `Device` **MUST** provide the data item `AVAILABILITY`, which represents the *Agent's* ability to communicate with the data source.

In the `MTConnectDevices` XML document, `Device` is a unique type of *Structural Element*. `Device` carries all of the properties of a `Component` (see *Section 4.4*). Additionally, `Device` **MUST** have a `uuid` attribute that uniquely identifies the piece of equipment. The value for the `uuid` **SHOULD NOT** change over time. The value for `uuid` **MUST** be universally unique and **MUST** only appear once in any MTConnect installation. All *Structural Elements* and *Data Entities* associated with a piece of equipment are therefore uniquely identified through their association with the `Device` container.

Element	Description	Occurrence
`Device`	The primary container element for each piece of equipment. `Device` is organized within the `Devices` container. There **MAY** be multiple `Device` elements in an XML document.	1..INF

Note: Some data sources may not be integral to a specific piece of equipment. These data sources may function independently or produce data that is not relevant to a specific piece of equipment. An example would be a temperature sensor installed in a plant to monitor the ambient air temperature. In such a case, these individual data sources, if they singularly or together perform a unique function, **MAY** be modeled in a MTConnect XML document as a `Device`. When modeled as a `Device`, these data sources **MUST** provide all of the data and capabilities defined for a `Device`.

It is possible for a piece of equipment to be defined as both a `Component` of a `Device` and simultaneously function independently as a separate `Device` reporting data directly through a *MTConnect Agent* using its own `uuid`. An example would be a temperature monitoring system that is defined as a `Device` reporting data about the environment within a facility and simultaneously reporting data for a `Component` of another piece of equipment that it is monitoring.

4.2.1 XML Schema Structure for `Device`

The following XML schema represents the structure of the `Device` XML element showing the attributes defined for `Device` and the elements that may be associated with `Device`.

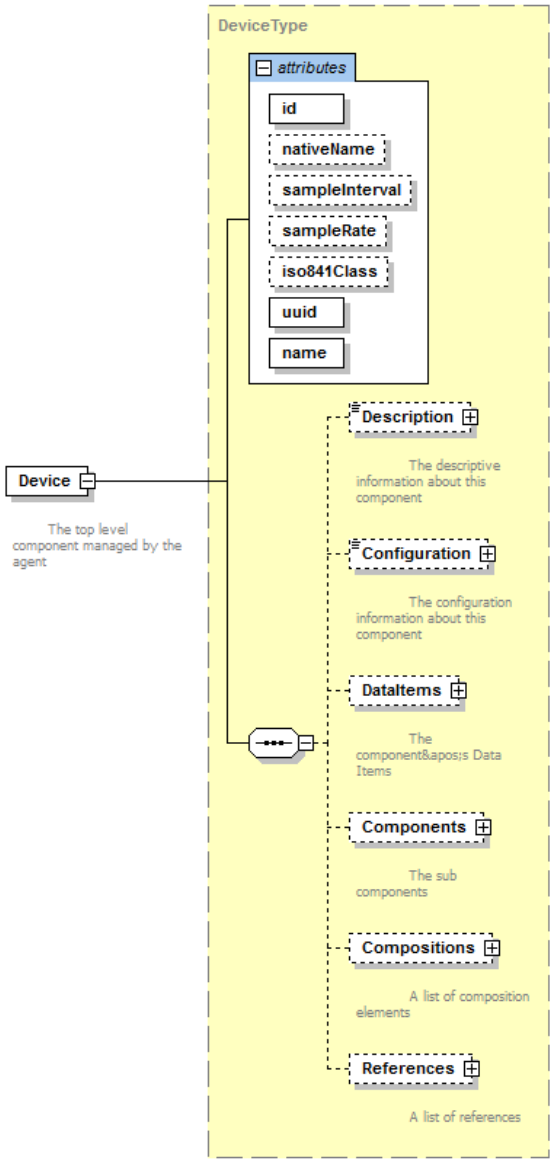


Figure 3: `Device` Schema Diagram

4.2.2 Attributes for `Device`

The following table defines the attributes that may be used to provide additional information for a `Device` type element.

Attribute	Description	Occurrence
`id`	The unique identifier for this XML element. `id` is a required attribute. An `id` **MUST** be unique across all the `id` attributes in the document. An XML ID-type.	1
`nativeName`	The common name normally associated with this piece of equipment. `nativeName` is an optional attribute.	0..1
`sampleInterval`	An optional attribute that is an indication provided by a piece of equipment describing the interval in milliseconds between the completion of the reading of the data associated with the `Device` element until the beginning of the next sampling of that data. This indication is reported as the number of milliseconds between data captures. This information may be used by client software applications to understand how often information from a piece of equipment is expected to be refreshed. The refresh rate for all data from the piece of equipment will be the same as for the `Device` element unless specifically overridden by another `sampleInterval` provided for a `Component` of the `Device` element. If the value of `sampleInterval` is less than one millisecond, the value will be represented as a floating-point number. For example, an interval of 100 microseconds would be 0.1.	0..1**
~~`sampleRate`~~	**DEPRECATED** in *MTConnect Version. 1.2*. Replaced by `sampleInterval`.	0..1***
~~`iso841Class`~~	**DEPRECATED** in *MTConnect Version 1.1*.	0..1***

Attribute	Description	Occurrence
uuid	A unique identifier for this XML element. uuid is a required attribute. The uuid **MUST** be unique amongst all uuid identifiers used in an MTConnect installation. For example, this may be a combination of the manufacturer's code and serial number. The uuid **SHOULD** be alphanumeric and not exceed 255 characters. An NMTOKEN XML type.	1*
name	The name of the piece of equipment represented by the Device element. name is a required attribute. This name **MUST** be unique for each Device XML element defined in the MTConnectDevices document. An NMTOKEN XML type.	1

Notes:* A uuid **MUST** be provided for each Device element. It is optional for all other Structural Elements.

** The sampleInterval is used to aid a client software application in interpreting values provided by some Data Entities. This is the desired sample interval and may vary depending on the capabilities of the piece of equipment.

*** Remains in schema for backwards compatibility.

4.2.3 Elements for `Device`

The following table lists the elements defined to provide additional information for a `Device` element. These elements are organized in the `Device` container.

Element	Description	Occurrence
`Description`	An XML element that can contain any descriptive content.	0..1
`Configuration`	An XML element that contains technical information about a piece of equipment describing its physical layout or functional characteristics.	0..1
`DataItems`	A container for the *Data Entities* (See *Section 7 and 8* of this document for more details) provided by this `Device` element.	1 *
`Components`	A container for the `Component` elements associated with this `Device` element.	0..1
`Compositions`	A container for the `Composition` elements associated with this `Device` element.	0..1
`References`	A container for the `Reference` elements associated with this `Device` element.	0..1

Note: * `DataItems` **MUST** be provided since every piece of equipment **MUST** report `AVAILABILITY`.

4.2.3.1 `Description` for `Device`

The following XML schema represents the structure of the `Description` XML element showing the attributes defined for `Description`. `Description` can contain any descriptive content for this piece of equipment. This element is defined to contain mixed content and additional XML elements (indicated by the `any` element in the schema below) **MAY** be added to extend the schema for `Description`.

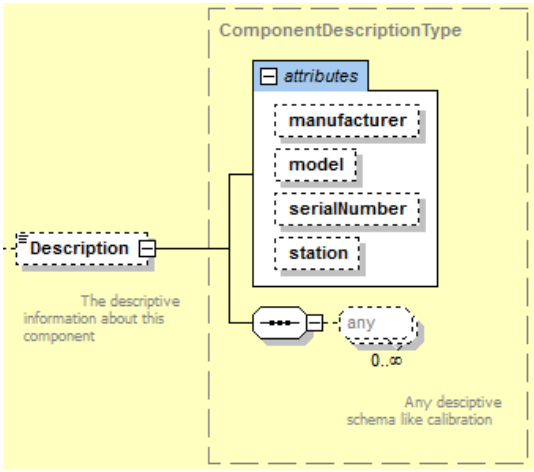


Figure 4: `Description` Schema Diagram

219 The following table lists the attributes defined for the `Description` XML element.

Attribute	Description	Occurrence
`manufacturer`	The name of the manufacturer of the piece of equipment represented by the `Device` element. `manufacturer` is an optional attribute.	0..1
`model`	The model description of the piece of equipment represented by the `Device` element. `model` is an optional attribute.	0..1
`serialNumber`	The serial number associated with piece of equipment represented by the `Device` element. `serialNumber` is an optional attribute.	0..1
`station`	The station where the equipment represented by the `Device` element is located when it is part of a manufacturing unit or cell with multiple stations. `station` is an optional attribute.	0..1

220

221 The content of `Description` **MAY** include any additional descriptive information the
222 implementer chooses to include regarding a piece of equipment. This content **SHOULD** be
223 limited to information not included elsewhere in the `MTConnectDevices` XML document.

224 An example of a `Description` is as follows:

```
225  1.  <Description manufacturer="Example Co" serialNumber="A124FFF"
226  2.    station="2"> Example Co Simulated Vertical 3 Axis Machining center.
227  3.  </Description>
```

228 #### 4.2.3.2 `Configuration` for `Device`

229 The `Configuration` XML element contains technical information about a piece of
230 equipment. `Configuration` **MAY** include any information describing the physical layout or
231 functional characteristics of the piece of equipment, such as capabilities, testing, installation,
232 operation, calibration, or maintenance.

233

Not all types of equipment support `Configuration`. When `Configuration` is supported, details on the schema for `Configuration` will be included in the applicable sections of the MTConnect Standard.

Element	Description	Occurrence
`Configuration`	An XML element that contains technical information about a piece of equipment describing its physical layout or functional characteristics.	0..1

Configuration data for `Device` is structured in the `MTConnectDevices` XML document as shown below. `AbstractConfiguration` is an abstract type XML element. It will never appear in the XML document representing a piece of equipment. When `Configuration` is supported for a type of equipment, that configuration will appear in the XML document. Currently, `Sensor` is the only type of equipment that supports `Configuration`. `SensorConfiguration` is described in detail in *Section 9.4*.

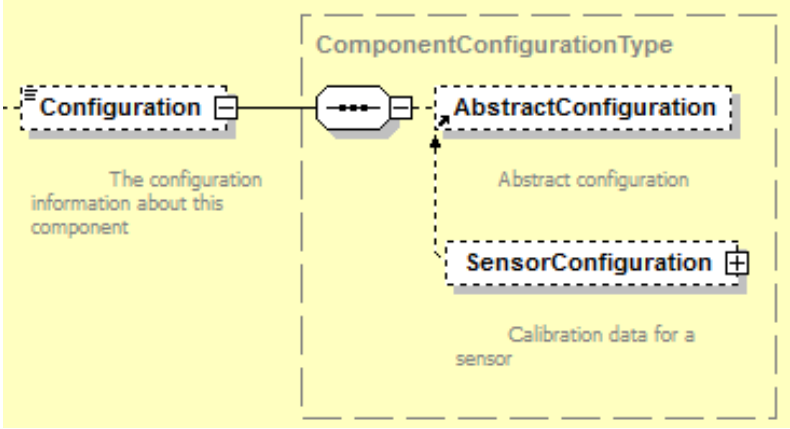


Figure 5: `Configuration` Schema Diagram

4.2.3.3 `DataItems` for `Device`

`DataItems` is an XML container that provides structure for organizing the data reported by a piece of equipment that is associated with the `Device` element.

`DataItems` **MUST** be provided since every piece of equipment **MUST** report the data item `AVAILABILITY`.

See *Sections 7 and 8* of this document for details on the `DataItems` XML element.

4.2.3.4 Components within Device

The use of the XML container Components within a Device element provides the ability to break down the structure of a Device element into *Top Level* and *Lower Level* physical and logical sub-parts. If a Components XML element is provided, then only one Components element **MUST** be defined for a Device element.

4.2.3.5 Compositions for Device

Compositions is an XML container used to organize Composition elements associated with a Device element. See *Section 4.5* for details on Compositions.

4.2.3.6 References for Device

References is an XML container used to organize Reference elements associated with a Device element. See *Section 4.7* for details on References.

4.3 Components

Components is an XML container used to group information describing physical parts or logical functions of a piece of equipment. Components contains one or more Component XML elements.

Element	Description	Occurrence
Components	XML container that consists of one or more types of Component XML elements. If a Components XML element is provided, then only one Components element **MUST** be defined for a Device element.	0..1

4.4 Component

A Component XML element is a container type XML element used to organize information describing a physical part or logical function of a piece of equipment. It also provides structure for describing the *Lower Level Structural Elements* associated with the Component. Component is an abstract type XML element and will never appear directly in the MTConnect XML document. As an abstract type XML element, Component will be replaced in the XML document by specific Component types. XML elements representing Component are described in *Section 5* and include elements such as Axes, Controller, and Systems.

Element	Description	Occurrence
Component	An abstract XML element. Replaced in the XML document by types of Component elements representing physical parts and logical functions of a piece of equipment. There can be multiple types of Component XML elements in the document.	1..INF

4.4.1 XML Schema Structure for `Component`

The following XML schema represents the structure of a `Component` XML element showing the attributes defined for `Component` and the elements that **MAY** be associated with `Component`.

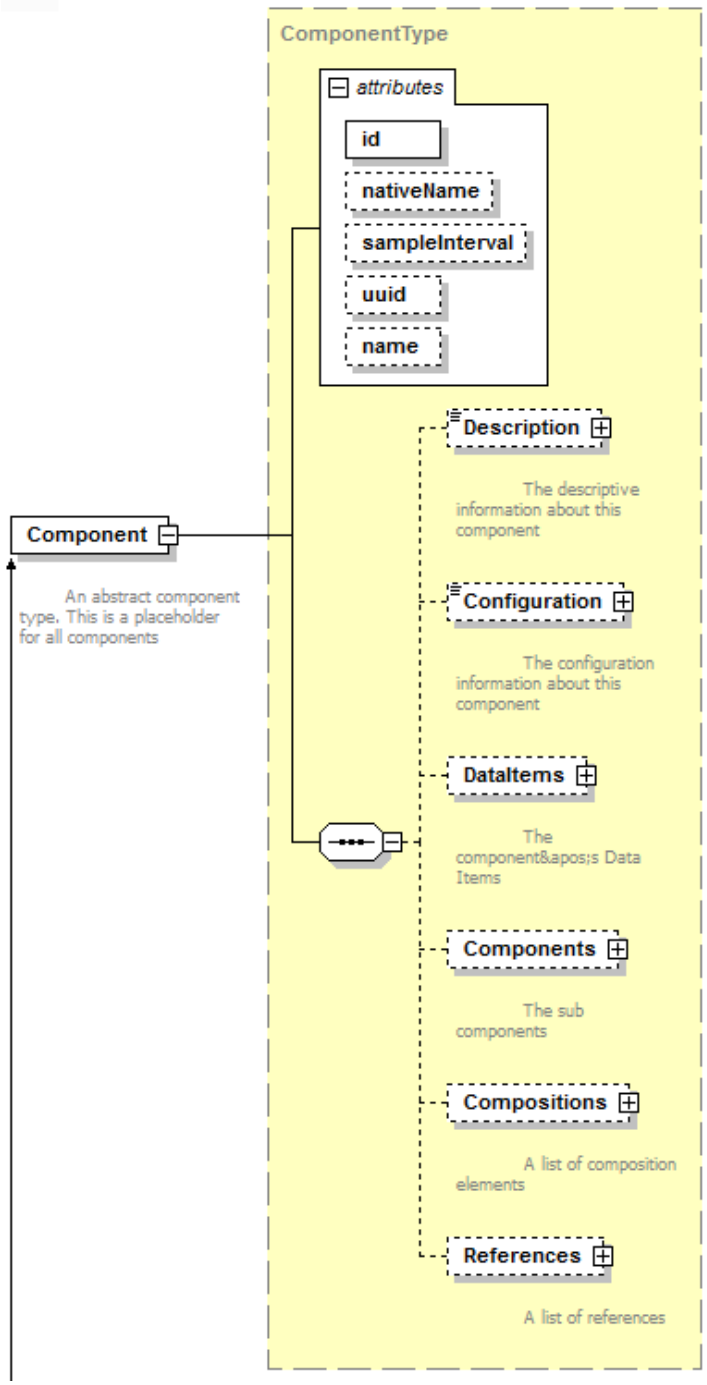


Figure 6: `Component` Schema

4.4.2 Attributes for `Component`

The following table defines the attributes that may be used to provide additional information for a `Component` type XML element.

Attribute	Description	Occurrence
`id`	The unique identifier for this XML element. `id` is a required attribute. An `id` **MUST** be unique across all the `id` attributes in the document. An XML ID-type.	1
`nativeName`	The common name normally associated with a specific physical or logical part of a piece of equipment. `nativeName` is an optional attribute.	0..1
`sampleInterval`	An optional attribute that is an indication provided by a piece of equipment describing the interval in milliseconds between the completion of the reading of the data associated with the `Component` element until the beginning of the next sampling of that data. This indication is reported as the number of milliseconds between data captures. This information may be used by client software applications to understand how often information from a piece of equipment for a specific `Component` element is expected to be refreshed. The refresh rate for data from all *Lower Level* `Component` elements will be the same as for the parent `Component` element unless specifically overridden by another `sampleInterval` provided for the *Lower Level* `Component` element. If the value of `sampleInterval` is less than one millisecond, the value will be represented as a floating-point number. For example, an interval of 100 microseconds would be 0.1.	0..1**
~~`sampleRate`~~	**DEPRECATED** in *MTConnect Version 1.2*. Replaced by `sampleInterval`.	0..1***
`uuid`	A unique identifier for this XML element. `uuid` is an optional attribute. The `uuid` **MUST** be unique amongst all `uuid` identifiers used in an MTConnect installation. For example, this may be a combination of the manufacturer's code and serial number. The `uuid` **SHOULD** be alphanumeric and not exceed 255 characters. An NMTOKEN XML type.	0..1*

Attribute	Description	Occurrence
name	The name of the Component element. name is an optional attribute. However, if there are multiple *Lower Level* components that have the same parent and are of the same component type (example Linear), then the name attribute **MUST** be provided for all *Lower Level* components of the same element type to differentiate between the similar components. When provided, name **MUST** be unique for all *Lower Level* components of a parent Component. An NMTOKEN XML type.	0..1

Notes: * While uuid **MUS**T be provided for the Device element, it is optional for Component elements.

** The sampleInterval is used to aid a client software application in interpreting values provided by some *Data Entities*. This is the desired sample interval and may vary depending on the capabilities of the piece of equipment.

***Remains in schema for backwards compatibility.

4.4.3 Elements of Component

The following table lists the elements defined to provide additional information for a Component type XML element.

Element	Description	Occurrence
Description	An element that can contain any descriptive content.	0..1
Configuration	An XML element that contains technical information about a component describing its physical layout or functional characteristics.	0..1
DataItems	A container for the *Data Entities* (defined in *Section 8*) associated with this Component element.	0..1*
Components	A container for *Lower Level* Component XML elements associated with this parent Component.	0..1*
Compositions	A container for the Composition elements (defined in *Section 6*) associated with this Component element.	0..1
References	A container for the Reference elements associated with this Component element.	0..1*

Notes: *At least one of Components, DataItems, or References **MUST** be provided.

4.4.3.1 `Description` for `Component`

The following XML schema represents the structure of the `Description` XML element showing the attributes defined for `Description`. `Description` can contain any descriptive content of this `Component`. This element is defined to contain mixed content and additional XML elements (indicated by the `any` element in the schema below) **MAY** be added to extend the schema for `Description`.

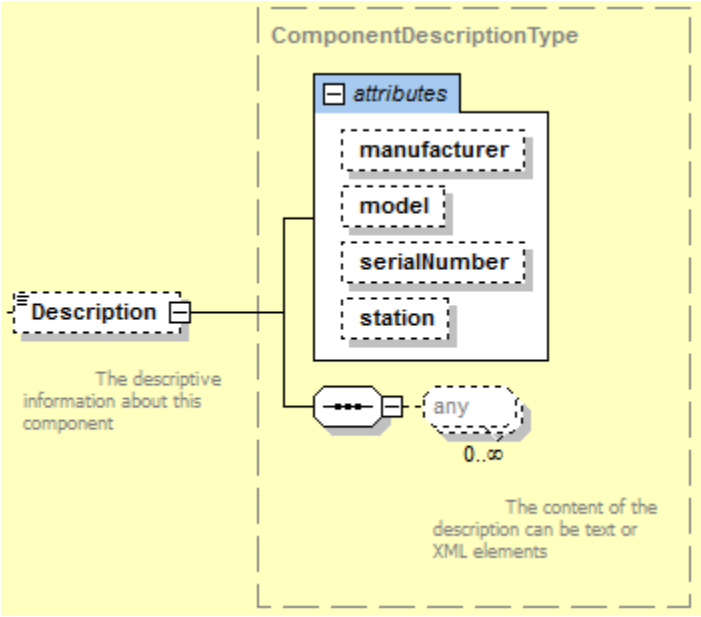


Figure 7: Schema for `Description` of `Component`

The following table lists the attributes defined for the `Description` XML element.

Attribute	Description	Occurrence
`manufacturer`	The name of the manufacturer of the physical or logical part of a piece of equipment represented by the `Component` element. `manufacturer` is an optional attribute.	0..1
`model`	The model description of the physical part or logical function of a piece of equipment represented by the `Component` element. `model` is an optional attribute.	0..1
`serialNumber`	The serial number associated with the physical part or logical function of a piece of equipment represented by the `Component` element. `serialNumber` is an optional attribute.	0..1
`station`	The station where the physical part or logical function of a piece of equipment represented by the `Component` element is located when it is part of a manufacturing unit or cell with multiple stations. `station` is an optional attribute.	0..1

The content of `Description` **MAY** include any additional descriptive information the implementer chooses to include regarding the `Component` element. This content **SHOULD** be limited to information not included elsewhere in the `MTConnectDevices` XML document.

An example of a `Description` element is as follows:

```
<Description manufacturer="Example Co"
  serialNumber="EXCO-TT-099PP-XXXX"> Advanced Pulse watt-hour transducer
  with pulse output
</Description>
```

4.4.3.2 `Configuration` for `Component`

The `Configuration` XML element contains technical information about a component. `Configuration` **MAY** include any information describing the physical layout or functional characteristics of a component, such as capabilities, testing, installation, operation, calibration, or maintenance.

Not all Component types support Configuration. When Configuration is supported, details on the schema for Configuration will be included in the applicable sections of the MTConnect Standard.

Element	Description	Occurrence
Configuration	An XML element that contains technical information about a component describing its physical layout or functional characteristics.	0..1

Configuration data for Component is structured in the MTConnectDevices XML document as shown below. AbstractConfiguration is an abstract type XML element. It will never appear in the XML document for a device. When Configuration is supported for a Component type, that configuration will appear in the XML document. Currently, Sensor is the only component type that supports Configuration. SensorConfiguration is described in detail in *Section 9.4*.

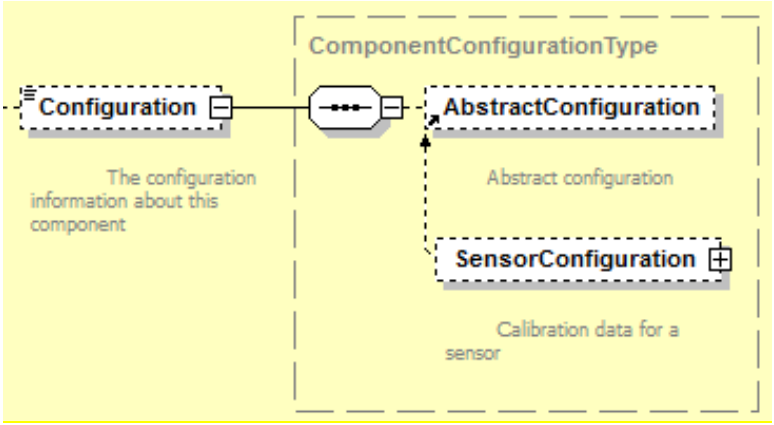


Figure 8: Component Configuration Schema

4.4.3.3 DataItems for Component

DataItems is an XML container that provides structure for organizing the data reported by a piece of equipment that is associated with the Component.

See *Section 7* of this document for details on the DataItems XML element.

4.4.3.4 Components within Component

The use of the XML container Components within a Component element provides the ability to further break down the structure of a Component element into even *Lower Level* physical and logical sub-parts. These *Lower Level* elements can add more clarity and granularity to the physical or logical structure of a piece of equipment and the data associated with that equipment.

This parent-child relationship can be extended down to any level necessary to fully describe a piece of equipment. These *Lower Level* `Component` elements use the same XML structure as `Component` defined in *Section 4.4.1* of this document.

A parent `Component` and the *Child Elements* are represented in a XML document as follows:

```
1.  <Devices>
2.    <Device>
3.      <Components>
4.        <Axes> (Component)
5.          <Components>
6.            <Linear> (Component)
7.              <Components>
8.                <Etc. > (Component)
```

4.4.3.5 `Compositions` for `Component`

`Compositions` is an XML container used to organize the lowest level structural building blocks contained within a `Component` as defined below.

4.4.3.6 `References` for `Component`

`References` is an XML container used to organize `Reference` elements associated with a `Component` element. See *Section 4.7* for details on `References`.

4.5 `Compositions`

`Compositions` is an XML container that defines the lowest level structural building blocks contained within a `Component` element.

`Compositions` contains one or more `Composition` XML elements.

Element	Description	Occurrence
`Compositions`	XML Container consisting of one or more types of `Composition` XML elements. Only one `Compositions` container **MAY** appear for a `Component` element.	0..1

4.6 `Composition`

`Composition` XML elements are used to describe the lowest level physical building blocks of a piece of equipment contained within a `Component`.

Like `Component` elements, `Composition` elements provide the ability to organize information describing *Lower Level* sub-parts of a higher-level `Component` element. However, unlike `Component`, `Composition` **MUST NOT** be further sub-divided and *Data Entities* **MUST NOT** be assigned to `Composition` elements.

Composition elements are used to add more clarity and granularity to the data being retrieved from a piece of equipment. The meaning of the data associated with a Component may be enhanced by designating a specific Composition element associated with that data.

An example of the additional detail provided when using Composition elements would be:

> A TEMPERATURE associated with a Linear type axis may be further clarified by referencing the MOTOR or AMPLIFIER type Composition element associated with that axis, which differentiates the temperature of the motor from the temperature of the amplifier.

Composition is a typed XML element and will always define a specific type of structural building block contained within a Component. XML elements representing the types of Composition elements are described in *Section 6* of this document and include elements describing such basic building blocks as motors, amplifiers, filters, and pumps.

A parent Component and child Composition elements are represented in an XML document as follows:

```
1.  <Devices>
2.   <Device>
3.     <Components>
4.      <Axes> (Component)
5.       <Components>
6.        <Linear> (Component)
7.         <Compositions>
8.           <Composition>
9.           <Composition>
10.          <Composition>
```

Element	Description	Occurrence
Composition	An XML element used to describe the lowest level structural building blocks contained within a Component element. Composition is a typed XML element. There can be multiple types of Composition XML elements defined for a Component element.	1..INF

4.6.1 XML Schema Structure for `Composition`

The following XML schema represents the structure of a `Composition` XML element showing the attributes defined for `Composition` and the elements that may be associated with `Composition` type XML elements.

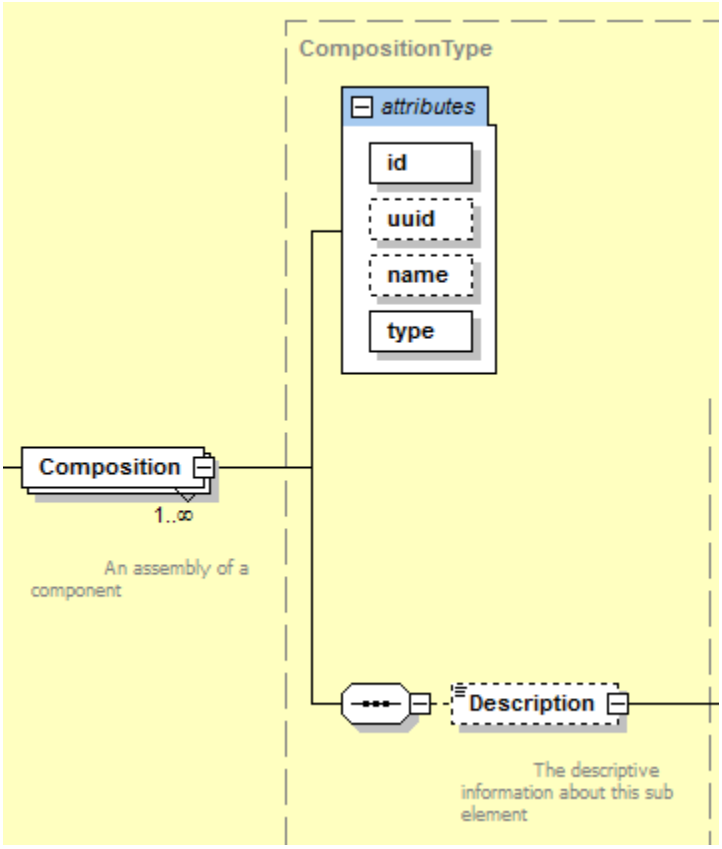


Figure 9: `Composition` Schema

4.6.2 Attributes for `Composition`

The following table defines the attributes that may be used to provide additional information for a `Composition` type XML element.

Attribute	Description	Occurrence
`id`	The unique identifier for this XML element. `id` is a required attribute. An `id` **MUST** be unique across all the `id` attributes in the document. An XML ID-type.	1
`uuid`	A unique identifier for this XML element. `uuid` is an optional attribute. The `uuid` **MUST** be unique amongst all `uuid` identifiers used in an MTConnect installation. For example, this may be a combination of the manufacturer's code and serial number. The `uuid` **SHOULD** be alphanumeric and not exceed 255 characters. An NMTOKEN XML type.	0..1
`name`	The name of the `Composition` element. `name` is an optional attribute. If provided, `name` **MUST** be unique within a `Component` element. An NMTOKEN XML type.	0..1
`type`	The type of `Composition` element. `type` is a required attribute. Examples of types are `MOTOR`, `FILTER`, `PUMP`, and `AMPLIFIER`. Refer to *Section 6* for a list of currently defined types.	1

4.6.3 Elements of `Composition`

The following table lists the elements defined to provide additional information for a `Composition` type XML element.

Element	Description	Occurrence
`Description`	An element that can contain any descriptive content.	0..1

4.6.3.1 `Description` for `Composition`

The following XML schema represents the structure of the `Description` XML element showing the attributes defined for `Description`. `Description` can contain any descriptive content for this `Composition` element. This element is defined to contain mixed content and additional XML elements (indicated by the `any` element in the schema below) **MAY** be added to extend the schema for `Description`.

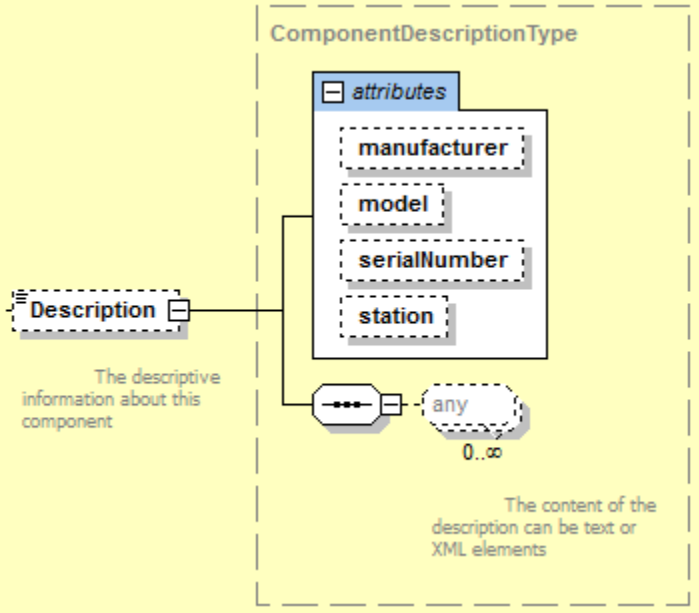


Figure 10: Schema for `Description` of `Composition`

438 The following table lists the attributes defined for the Description XML element.

439

Attribute	Description	Occurrence
manufacturer	The name of the manufacturer of the physical part of a piece of equipment represented by the Composition element. manufacturer is an optional attribute.	0..1
model	The model description of the physical part of a piece of equipment represented by the Composition element. model is an optional attribute.	0..1
serialNumber	The serial number associated with the physical part of a piece of equipment represented by the Composition element. serialNumber is an optional attribute.	0..1
station	The station where the physical part of a piece of equipment represented by the Composition element is located when it is part of a manufacturing unit or cell with multiple stations. station is an optional attribute.	0..1

440

441 The content of Description **MAY** include any additional descriptive information the
442 implementer chooses to include regarding the Composition element. This content **SHOULD**
443 be limited to information not included elsewhere in the MTConnectDevices XML document.

444 An example of a Description element is as follows:

```
445  11. <Description manufacturer="Example Co" serialNumber="A124FFF"
446  12. station="2"> Spindle motor associated with Path 2.
447  13. </Description>
```

448 4.7 References

449 References is an XML container that organizes pointers to information defined elsewhere
450 within the XML document for a piece of equipment.

451 References may be modeled as part of a Device, Component or Interface type
452 *Structural Element*.

453

454 References contains one or more Reference XML elements.

Element	Description	Occurrence
References	XML Container consisting of one or more types of Reference XML elements. Only one References container **MUST** appear for a Device, Component, or Interface element.	0..1

455

456 4.8 Reference

457 Reference is a pointer to information that is associated with another *Structural Element*
458 defined elsewhere in the XML document for a piece of equipment. That information may be
459 data from the other element or the entire structure of that element.

460 Reference is an efficient method to associate information with an element without duplicating
461 any of the data or structure. For example, a Bar Feeder System may make a request for the
462 BarFeederInterface and receive all the relevant data for the interface and the associated
463 spindle (ROTARY element) that is referenced as part of the BarFeederInterface.

464 Reference is an abstract type XML element and will never appear directly in the MTConnect
465 XML document. As an abstract type XML element, Reference will be replaced in the XML
466 document by a specific Reference type. The current supported types of Reference are
467 DataItemRef and ComponentRef XML elements.

468

The following XML schema represents the structure of the `Reference` XML element.

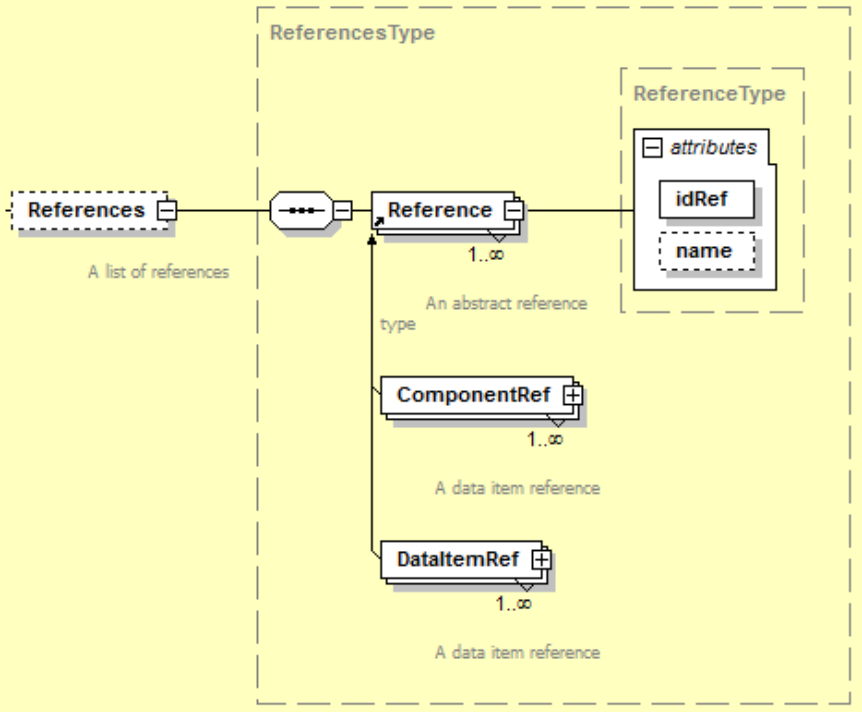


Figure 11: `Reference` Schema Diagram

4.8.1 `ComponentRef`

`ComponentRef` XML element is a pointer to all of the information associated with another *Structural Element* defined elsewhere in the XML document for a piece of equipment. `ComponentRef` allows all of the information (*Lower Level* `Components` and all *Data Entities*) that is associated with the other *Structural Element* to be directly associated with this XML element.

The following XML schema represents the structure of a `ComponentRef` XML element showing the attributes defined for `ComponentRef`.

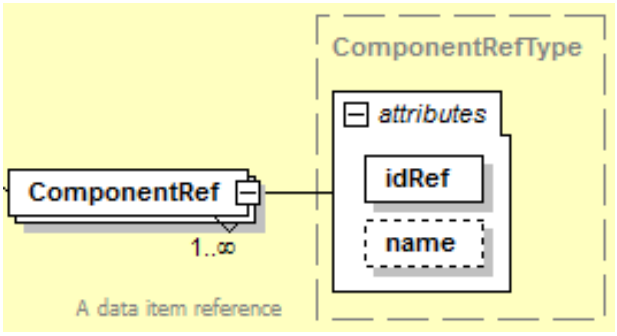


Figure 12: `ComponentRef` Schema Diagram

The following table lists the `attributes` defined for the `ComponentRef` element.

Attribute	Description	Occurrence
`idRef`	A pointer to the `id` attribute of the `Component` that contains the information to be associated with this XML element. `idRef` is a required attribute.	1
`name`	The name of the `ComponentRef` element. `name` is an optional attribute. However, if there are multiple `ComponentRef` elements defined for a component, the `name` attribute **MUST** be provided for all `ComponentRef` elements to differentiate between the similar elements. When provided, `name` **MUST** be unique for all `ComponentRef` elements associated with the *Parent Element*. An NMTOKEN XML type.	0..1

4.8.2 `DataItemRef`

`DataItemRef` XML element is a pointer to a *Data Entity* associated with another *Structural Element* defined elsewhere in the XML document for a piece of equipment. `DataItemRef` allows the data associated with a data item defined in another *Structural Element* to be directly associated with this XML element.

The following XML schema represents the structure of a `DataItemRef` XML element showing the attributes defined for `DataItemRef`.

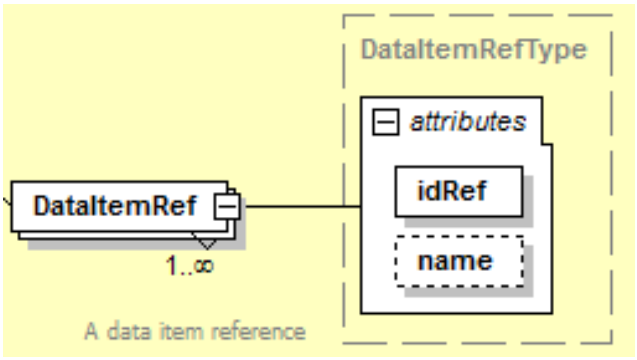


Figure 13: `DataItemRef` Schema Diagram

497 The following table lists the `attributes` defined for the `DataItemRef` element.

498

Attribute	Description	Occurrence
`idRef`	A pointer to the `id` attribute of the `DataItem` that contains the information to be associated with this XML element. `idRef` is a required attribute.	1
`name`	The name of the `DataItemRef` element. `name` is an optional attribute. However, if there are multiple `DataItemRef` elements defined for a component, the `name` attribute **MUST** be provided for all `DataItemRef` elements to differentiate between the similar elements. When provided, `name` **MUST** be unique for all `DataItemRef` elements associated with the *Parent Element*. An NMTOKEN XML type.	0..1

499

5 Component *Structural Elements*

Component *Structural Elements* are XML containers used to represent physical parts or logical functions of a piece of equipment.

Component *Structural Elements* are defined into two major categories:

- *Top Level* Component elements are used to group the *Structural Elements* representing the most significant physical or logical functions of a piece of equipment. The *Top Level* Component elements provided in an MTConnectDevices document **SHOULD** be restricted to those defined in the table below. However, these *Top Level* Component elements **MAY** also be used as *Lower Level* Component elements; as required.

- *Lower Level* Component elements are used to describe the sub-parts of the parent Component to provide more clarity and granularity to the physical or logical structure of the *Top Level* Component elements.

This section (*Section 5*) of the *Devices Information Model* provides guidance for the most common relationships between *Top Level* Component elements and *Lower Level* child components. However, all Component elements **MAY** be used in any configuration, as required, to fully describe a piece of equipment.

As described in *Section 4* above, Component is an abstract type *Structural Element* within the *Devices Information Model* and will never appear directly in the MTConnectDevices XML document. As abstract type XML elements, Component will be replaced in the XML document by a specific Component type defined below.

The following table defines the *Top Level* Component elements available to describe a piece of equipment.

Top Level Component Element **	Description
Axes	An XML container used to organize the *Structural Elements* of a piece of equipment that perform linear or rotational motion.
Controller	An XML container used to organize information about an intelligent or computational function within a piece of equipment.
Systems	An XML container used to organize information for *Lower Level* elements representing the major sub-systems that are permanently integrated into a piece of equipment.
Auxiliaries	An XML container used to organize information for *Lower Level* elements representing functional sub-systems that provide supplementary or extended capabilities for a piece of equipment, but they are not required for the basic operation of the equipment.

Top Level Component Element **	Description
Resources	An XML container used to organize information for *Lower Level* elements representing types of items, materials, and personnel that support the operation of a piece of equipment or work to be performed at a location. Resources also represents materials or other items consumed or transformed by a piece of equipment for production of parts or other types of goods.
Interfaces	An XML container that organizes information used to coordinate actions and activities between pieces of equipment that communicate information between each other.

** Note: The following components have been relocated or redefined since they are not classified as restricted *Top Level* components:

- Power was **DEPRECATED** in *MTConnect Version 1.1* and was replaced by the *Data Entity* called AVAILABILITY.

- Door has been redefined as a *Lower Level* component of a parent Component element or as a Composition element.

- Actuator, due to its uniqueness, has been redefined as a piece of equipment with the ability to be represented as a *Lower Level* component of a parent Component element or as a Composition element.

- Sensor, due to its uniqueness, has been redefined as a piece of equipment with the ability to be represented as a *Lower Level* component of a parent Component element (See *Section 9* for further detail).

- Stock has been redefined as a *Lower Level* component of the Resources *Top Level* Component element.

The common relationship between the *Top Level* Component elements and the *Lower Level* child Component elements are described below. It should be noted that as the MTConnect Standard evolves, more Component types will be added to organize information for new types of equipment and/or new physical or logical sub-parts of equipment.

5.1 Axes

Axes is a *Top Level* Component element. It is a container that organizes information representing the *Structural Elements* that perform linear or rotational motion for a piece of equipment.

Axes organizes information for the individual physical axes into Component types of Linear and Rotary based on the type of motion performed by each axis. Axes **MUST** contain at least one Linear or one Rotary type axis.

The following diagram defines the relationship between the `Axes` container and the individual axis type *Structural Elements*.

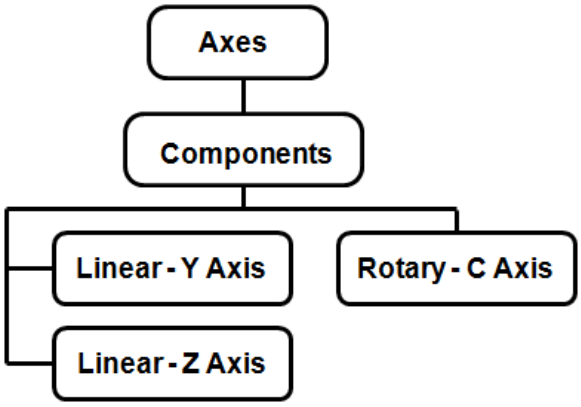


Figure 14: `Axes` Example with Two Linear Axes and One Rotary Axis

5.1.1 `Linear`

A `Linear` axis represents the movement of a physical piece of equipment, or a portion of the equipment, in a straight line.

Movement may be either in a positive or negative direction.

`Linear` type axes **MUST** be identified using a value for the `name` attribute as X, Y, or Z with numbers appended for additional axes in the same plane. Additional linear axes are often referred to as U, V, and W. However, MTConnect defines the secondary axes to X, Y, and Z as X2, Y2, and Z2.

If the piece of equipment is unable to provide information associated with the `name` attribute, then the `nativeName` attribute **MUST** be included to identify the axis.

5.1.2 `Rotary`

A `Rotary` axis represents any non-linear or rotary movement of a physical piece of equipment or a portion of the equipment.

`Rotary` type axes **MUST** be identified using a value for the `name` attribute as A, B, and C for axes that rotate around the X, Y, and Z axes respectively. As with the `Linear` axes, a number **MUST** be appended for additional axes in the same plane (C, C2, C3, C4, …).

If the piece of equipment is unable to provide information associated with the `name` attribute, then the `nativeName` attribute **MUST** be included to identify the axis.

An axis whose function is to provide rotary motion may function as a continuous rotation (SPINDLE mode), continuous-path contour rotary motion (CONTOUR mode), or positioning (INDEX mode) to discrete rotary positions. As such, a Rotary type axis **SHOULD** specify a ROTARY_MODE data item identifying the operating mode of the axis: SPINDLE, INDEX, or CONTOUR.

5.1.2.1 Chuck

Chuck is an XML container that provides the information about a mechanism that holds a part or stock material in place. It may also represent the information about any other type mechanism that holds items in place within a piece of equipment.

The operation of a Chuck when represented as a Component element is defined by CHUCK_STATE. The value of CHUCK_STATE **MAY** be OPEN, CLOSED, or UNLATCHED.

Chuck may be used in the MTConnectDevices document as either a *Lower Level* component or as a Composition element of a parent Component element.

5.2 Controller

Controller is a *Top Level* container that organizes information for an intelligent part of a piece of equipment that monitors and calculates information to alter the operating conditions of the equipment. Typical types of controllers for a piece of equipment include CNC (Computer Numerical Control), PAC (Programmable Automation Control), IPC (Industrialized Computer), or IC (Imbedded Computer).

Controller provides information regarding the execution of a control program(s), the mode of operation of the piece of equipment, and fault information regarding the operation of the equipment.

Note: *MTConnect Version 1.1.0* and later implementations **SHOULD** use a *Lower Level* Component element called Path to represent an individual tool path or other independent function within a Controller element. When the Controller element is capable of executing more than one simultaneous and independent programs, the implementation **MUST** specify a *Lower Level* Path element representing each of the independent functions of the Controller.

5.2.1 Path

Path is an XML container that represents the information for an independent operation or function within a Controller. For many types of equipment, Path represents a set of Axes, one or more Program elements, and the data associated with the motion of a control point as it moves through space. However, it **MAY** also represent any independent function within a Controller that has unique data associated with that function.

Path **SHOULD** provide an EXECUTION data item to define the operational state of the Controller component of the piece of equipment.

If the Controller is capable of performing more than one independent operation or function simultaneously, a separate Path component **MUST** be used to organize the data associated with each independent operation or function.

5.3 Systems

Systems is a *Top Level* XML container that provides structure for the information describing one or more *Lower Level* functional systems that perform as discrete operating modules of the equipment or provide utility type services to support the operation of the equipment. These systems are required for the piece of equipment to perform its intended function and are permanently integrated into the piece of equipment.

Since these systems operate as separate functional units, they are represented in the MTConnectDevices XML document as individual *Lower Level* Component elements of Systems based on the function or service provided.

5.3.1 Hydraulic System

Hydraulic is an XML container that represents the information for a system comprised of all the parts involved in moving and distributing pressurized liquid throughout the piece of equipment.

5.3.2 Pneumatic System

Pneumatic is an XML container that represents the information for a system comprised of all the parts involved in moving and distributing pressurized gas throughout the piece of equipment.

5.3.3 Coolant System

Coolant is an XML container that represents the information for a system comprised of all the parts involved in distribution and management of fluids that remove heat from a piece of equipment.

5.3.4 Lubrication System

Lubrication is an XML container that represents the information for a system comprised of all the parts involved in distribution and management of fluids used to lubricate portions of the piece of equipment.

5.3.5 Electric System

Electric is an XML container that represents the information for the main power supply for device piece of equipment and the distribution of that power throughout the equipment. The electric system will provide all the data with regard to electric current, voltage, frequency, etc. that applies to the piece of equipment as a functional unit. Data regarding electric power that is specific to a Component will be reported as *Data Entities* for that specific Component.

5.3.6 Enclosure System

Enclosure is an XML container that represents the information for a structure used to contain or isolate a piece of equipment or area. The Enclosure system may provide information regarding access to the internal components of a piece of equipment or the conditions within the enclosure. For example, Door may be defined as a *Lower Level* Component or Composition element of the Enclosure system.

5.3.7 Protective System

Protective is an XML container that represents the information for those functions that detect or prevent harm or damage to equipment or personnel. Protective does not include the information relating to the Enclosure system.

5.3.8 ProcessPower System

ProcessPower is an XML container that represents the information for a power source associated with a piece of equipment that supplies energy to the manufacturing process separate from the Electric system. For example, this could be the power source for an EDM machining process, an electroplating line, or a welding system.

5.3.9 Feeder System

Feeder is an XML container that represents the information for a system that manages the delivery of materials within a piece of equipment. For example, this could describe the wire delivery system for an EDM or welding process; conveying system or pump and valve system distributing material to a blending station; or a fuel delivery system feeding a furnace.

5.3.10 Dielectric System

Dielectric is an XML container that represents the information for a system that manages a chemical mixture used in a manufacturing process being performed at that piece of equipment. For example, this could describe the dielectric system for an EDM process or the chemical bath used in a plating process.

5.4 Auxiliaries

Auxiliaries is a *Top Level* XML container that provides structure for the information describing one or more *Lower Level* functional systems that provide supplementary or additional capabilities for the operation of a piece of equipment. These systems extend the capabilities of a piece of equipment, but are not required for the equipment to function.

Since these systems operate as independent units or are only temporarily associated with a piece of equipment, they are represented in the MTConnectDevices XML document as individual *Lower Level* Component elements of Auxiliaries based on the function or service provided to the equipment.

5.4.1 `Loader` System

`Loader` is an XML container that represents the information for a unit comprised of all the parts involved in moving and distributing materials, parts, tooling, and other items to or from a piece of equipment.

5.4.2 `WasteDisposal` System

`WasteDisposal` is an XML container that represents the information for a unit comprised of all the parts involved in removing manufacturing byproducts from a piece of equipment.

5.4.3 `ToolingDelivery` System

`ToolingDelivery` is an XML container that represents the information for a unit involved in managing, positioning, storing, and delivering tooling within a piece of equipment.

5.4.4 `BarFeeder` System

`BarFeeder` is an XML container that represents the information for a unit involved in delivering bar stock to a piece of equipment.

5.4.5 `Environmental` System

`Environmental` is an XML container that represents the information for a unit or function involved in monitoring, managing, or conditioning the environment around or within a piece of equipment.

5.4.6 `Sensor` System

`Sensor` is a XML container that represents the information for a piece of equipment that responds to a physical stimulus and transmits a resulting impulse or value from a sensing unit. When modeled as a component of `Auxiliaries`, `sensor` **SHOULD** represent an integrated *sensor unit* system that provides signal processing, conversion, and communications. A *sensor unit* may have multiple *sensing elements*; each representing the data for a variety of measured values. See *Section 9.2* for more details on *sensor unit*.

Note: If modeling an individual sensor, then `sensor` should be associated with the component that the measured value is most closely associated. See *Section 5.7.3.*

5.5 `Resources`

`Resources` is a *Top Level* XML container that groups items that support the operation of a piece of equipment. `Resources` also represents materials or other items consumed, transformed, or used for production of parts, materials, or other types of goods by a piece of equipment.

5.5.1 Materials

Materials is an XML container that provides information about materials or other items consumed or used by the piece of equipment for production of parts, materials, or other types of goods. Materials also represents parts or part stock that are present at a piece of equipment or location to which work is applied to transform the part or stock material into a more finished state.

5.5.1.1 Stock

Stock is an XML container that represents the information for the material that is used in a manufacturing process and to which work is applied in a machine or piece of equipment to produce parts.

Stock may be either a continuous piece of material from which multiple parts may be produced or it may be a discrete piece of material that will be made into a part or a set of parts.

5.5.2 Personnel

Personnel is an XML container that provides information about an individual or individuals who either control, support, or otherwise interface with a piece of equipment.

5.6 Interfaces

Interfaces is a *Top Level* XML *Structural Element* in the MTConnectDevices XML document. Interfaces organizes the information provided by a piece of equipment used to coordinate activities with other pieces of equipment. As such, Interfaces represents the inter-device communication information between a piece of equipment and other pieces of equipment.

See Part 5.0 – Interfaces of the MTConnect Standard for detailed information on Interfaces.

5.7 Other Components

While most component elements **SHOULD** be modeled in a specific manner, there are some types of component elements that are used ubiquitously in equipment and **MAY** be associated with any number of different types of *parent* component elements.

These components **MAY** be modeled as *Lower Level* components of the *Parent Element*.

5.7.1 Actuator

Actuator is an XML container that represents the information for an apparatus for moving or controlling a mechanism or system. It takes energy usually provided by air, electric current, or liquid and converts the energy into some kind of motion.

5.7.2 Door

`Door` is an XML container that represents the information for a mechanical mechanism or closure that can cover, for example, a physical access portal into a piece of equipment. The closure can be opened or closed to allow or restrict access to other parts of the equipment.

When `Door` is represented as a `Component`, it **MUST** have a data item called `DOOR_STATE` to indicate if the door is `OPEN`, `CLOSED`, or `UNLATCHED`. A `Component` **MAY** contain multiple `Door` components.

5.7.3 Sensor

`Sensor` is a XML container that represents the information for a piece of equipment that responds to a physical stimulus and transmits a resulting impulse or value. If modeling individual sensors, then `sensor` should be associated with the component that the measured value is most closely associated.

See *Section 9* for more details on the use of `Sensor`.

6 Composition Type *Structural Elements*

Composition *Structural Elements* are used to describe the lowest level physical building blocks of a piece of equipment contained within a Component. By referencing a specific Composition element, further clarification and meaning to data associated with a specific Component can be achieved.

Both Component and Composition elements are *Lower Level* child Component XML elements representing the sub-parts of the parent Component. However, there are distinct differences between Component and Composition type elements.

Component elements may be further defined with *Lower Level* Component elements and may have associated *Data Entities*.

Composition elements represent the lowest level physical part of a piece of equipment. They **MUST NOT** be further defined with *Lower Level* Component elements and they **MUST NOT** have *Data Entities* directly associated with them. They do provide additional information that can be used to enhance the specificity of *Data Entities* associated with the parent Component.

The following table defines Composition type elements that are currently available to describe sub-parts of a Component element.

Element Type	Description
ACTUATOR	A mechanism for moving or controlling a mechanical part of a piece of equipment. It takes energy usually provided by air, electric current, or liquid and converts the energy into some kind of motion.
AMPLIFIER	An electronic component or circuit for amplifying power, electric current, or voltage.
BALLSCREW	A mechanical structure for transforming rotary motion into linear motion.
BELT	An endless flexible band used to transmit motion for a piece of equipment or to convey materials and objects.
BRAKE	A mechanism for slowing or stopping a moving object by the absorption or transfer of the energy of momentum, usually by means of friction, electrical force, or magnetic force.
CHOPPER	A mechanism used to break material into smaller pieces.
CIRCUIT_BREAKER	A mechanism for interrupting an electric circuit.

Element Type	Description
CHAIN	An interconnected series of objects that band together and are used to transmit motion for a piece of equipment or to convey materials and objects.
CHUCK	A mechanism that holds a part, stock material, or any other item in place.
CHUTE	An inclined channel for conveying material.
CLAMP	A mechanism used to strengthen, support, or fasten objects in place.
COMPRESSOR	A pump or other mechanism for reducing volume and increasing pressure of gases in order to condense the gases to drive pneumatically powered pieces of equipment.
DOOR	A mechanical mechanism or closure that can cover a physical access portal into a piece of equipment allowing or restricting access to other parts of the equipment.
DRAIN	A mechanism that allows material to flow for the purpose of drainage from, for example, a vessel or tank.
ENCODER	A mechanism used to measure rotary position.
FAN	Any mechanism for producing a current of air.
FILTER	Any substance or structure through which liquids or gases are passed to remove suspended impurities or to recover solids.
GRIPPER	A mechanism that holds a part, stock material, or any other item in place.
HOPPER	A chamber or bin in which materials are stored temporarily, typically being filled through the top and dispensed through the bottom.
MOTOR	A mechanism that converts electrical, pneumatic, or hydraulic energy into mechanical energy.
OIL	A viscous liquid.
PUMP	An apparatus raising, driving, exhausting, or compressing fluids or gases by means of a piston, plunger, or set of rotating vanes.
LINEAR_POSITION_FEEDBACK	A mechanism that measures linear motion or position.
POWER_SUPPLY	A unit that provides power to electric mechanisms.
PULLEY	A mechanism or wheel that turns in a frame or block and serves to change the direction of or to transmit force.

Element Type	Description
SENSING_ELEMENT	A mechanism that provides a signal or measured value.
STORAGE_BATTERY	A component consisting of one or more cells, in which chemical energy is converted into electricity and used as a source of power.
SWITCH	A mechanism for turning on or off an electric current or for making or breaking a circuit.
TANK	A receptacle or container for holding material.
TENSIONER	A mechanism that provides or applies a stretch or strain to another mechanism.
TRANSFORMER	A mechanism that transforms electric energy from a source to a secondary circuit.
VALVE	Any mechanism for halting or controlling the flow of a liquid, gas, or other material through a passage, pipe, inlet, or outlet.
WATER	A fluid.
WIRE	A string like piece or filament of relatively rigid or flexible material provided in a variety of diameters.

768

769 Note: As the MTConnect Standard evolves, more `Composition` types will be added.

7 *Data Entities* for `Device`

In the `MTConnectDevices` XML document, *Data Entities* are XML elements that describe data that can be reported by a piece of equipment and are associated with `Device` and `Component` *Structural Elements*. While the *Data Entities* describe the data that can be reported by a piece of equipment in the `MTConnectDevices` document, the actual data values are provided in the *Streams Information Model*. See *Part 3.0 – Streams Information Model* for the details on the reported values.

Each *Data Entity* **SHOULD** be modeled in the `MTConnectDevices` document such that it is associated with the *Structural Element* that the reported data directly applies.

When *Data Entities* are associated with a *Structural Element*, they are organized in a `DataItems` XML element. `DataItems` is a container type XML element. `DataItems` provides the structure for organizing individual `DataItem` elements that represent each *Data Enitity*. The `DataItems` container is comprised of one or more `DataItem` type XML element(s).

`DataItem` describes specific types of *Data Entities* that represent a numeric value, a functioning state, or a health status reported by a piece of equipment. `DataItem` provides a detailed description for each *Data Entity* that is reported; it defines the type of data being reported and an array of optional attributes that further describes that data. The different types of `DataItem` elements are defined in *Section 8*.

The following XML Tree demonstrates the relationship between *Data Entities* (`DataItem`) and the various *Structural Elements* in the `MTConnectDevices` XML document.

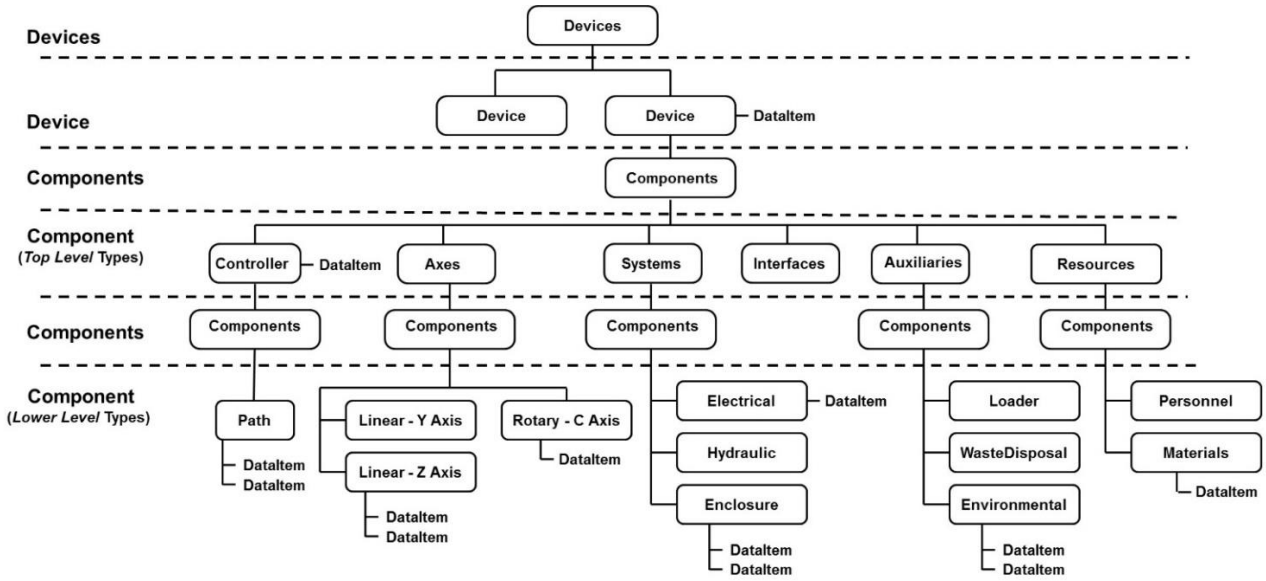


Figure 15: Example *Data Entities* for `Device` (`DataItem`)

7.1 DataItems

The `DataItems` XML element is the first, or highest, level container for the *Data Entities* associated with a `Device` or `Component` XML element. `DataItems` **MUST** contain only `DataItem` type elements. `DataItems` **MUST** contain at least one `DataItem` type element, but **MAY** contain multiple `DataItem` type elements.

Element	Description	Occurrence
`DataItems`	XML Container consisting of one or more types of `DataItem` XML elements. Only one `DataItems` container **MUST** appear for each *Structural Element* in the XML document.	0..1

7.2 DataItem

A DataItem XML element represents each *Data Entity* that **MAY** be reported by a piece of equipment through a *MTConnect Agent*. DataItem provides a detailed description for each *Data Entity* that is reported and defines the type of data being reported along with an array of optional attributes that further define that data. XML elements representing DataItem will include elements such as TEMPERATURE, PRESSURE, and VELOCITY.

Element	Description	Occurrence
DataItem	*Data Entity* describing a piece of information reported about a piece of equipment.	1..INF

7.2.1 XML Schema Structure for `DataItem`

The following XML schema represents the structure of a `DataItem` XML element showing the attributes defined for `DataItem` and the elements that may be associated with `DataItem` type XML elements.

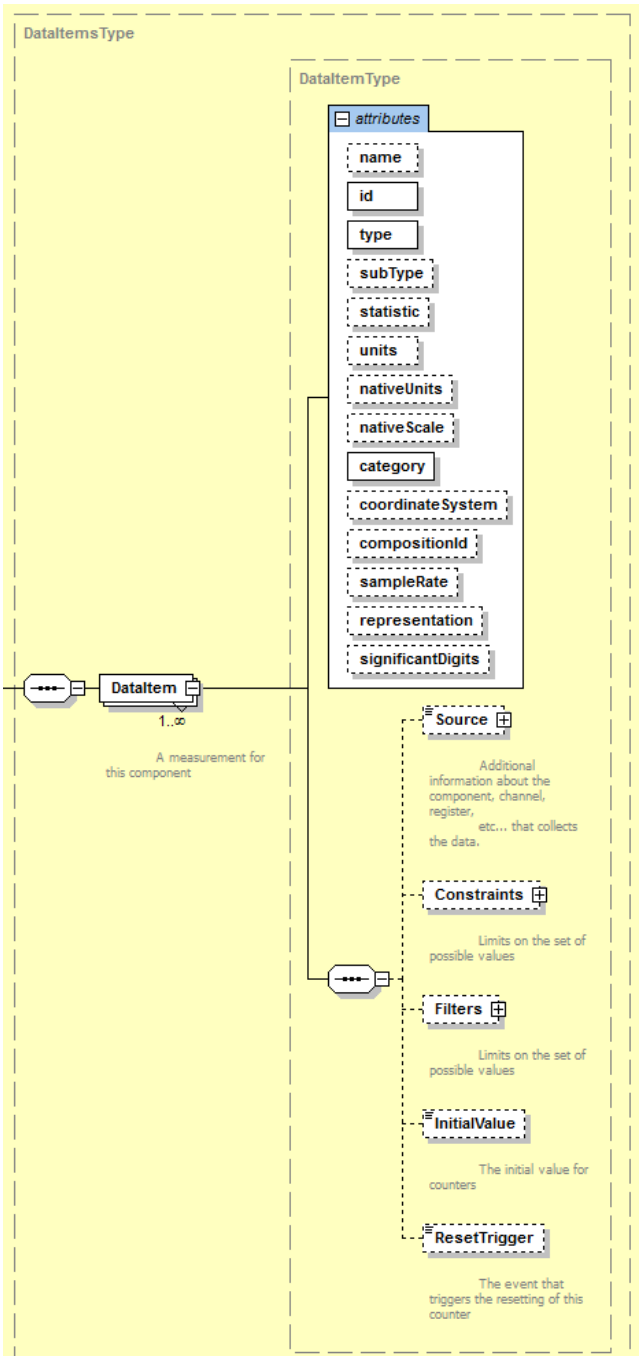


Figure 16: `DataItem` Schema Diagram

7.2.2 Attributes for `DataItem`

The following table lists the attributes defined to provide information for a `DataItem` type XML element.

`DataItem` **MUST** specify the `type` of data being reported, the `id` of the `DataItem`, and the `category` of the `DataItem`.

Attribute	Description	Occurrence
`name`	The name of the data item. `name` is provided as an additional human readable identifier for this data item in addition to the `id`. `name` is an optional attribute and will be implementation dependent. An NMTOKEN XML type.	0..1
`id`	The unique identifier for this data item. `id` is a required attribute. The `id` attribute **MUST** be unique within the `MTConnectDevices` document. An XML ID-type.	1
`type`	The type of data being measured. `type` is a required attribute. Examples of types are `POSITION`, `VELOCITY`, `ANGLE`, `BLOCK`, and `ROTARY_VELOCITY`.	1
`subType`	A sub-categorization of the data item `type`. `subType` is an optional attribute. For example, the `subType` of `POSITION` can be `ACTUAL` or `COMMANDED`. Not all `type` attributes have a `subType`.	0..1
`statistic`	Describes the type of statistical calculation performed on a series of data samples to provide the reported data value. `statistic` is an optional attribute. Examples of statistic are `AVERAGE`, `MINIMUM`, `MAXIMUM`, `ROOT_MEAN_SQUARE`, `RANGE`, `MEDIAN`, `MODE`, and `STANDARD_DEVIATION`.	0..1

Attribute	Description	Occurrence
units	The unit of measurement for the reported value of the data item. units is an optional attribute. Data items in the Sample category **MUST** report the standard units for the measured values. See *Section 7.2.2.7* for a list of available standard units identified in the MTConnect Standard.	0..1
nativeUnits	The native units of measurement for the reported value of the data item. nativeUnits is an optional attribute. See *Section 7.2.2.8* for a list of available native units identified in the MTConnect Standard	0..1
nativeScale	The nativeUnits may not be scaled to directly represent the original measured value. nativeScale **MAY** be used to convert the reported value to represent the original measured value. nativeScale is an optional attribute. As an example, the nativeUnits may be reported as GALLON/MINUTE. The measured value may actually be in 1000 GALLON/MINTUE. The value of the reported data **MAY** be divided by the nativeScale to convert the reported value to its original measured value and units. If provided, the value **MUST** be numeric.	0..1
category	Specifies the kind of information provided by a data item. category is a required attribute. The available options are SAMPLE, EVENT, or CONDITION.	1
coordinateSystem	For measured values relative to a coordinate system like POSITION, the coordinate system being used may be reported. coordinateSystem is an optional attribute. The available values for coordinateSystem are WORK and MACHINE.	0..1
compositionId	The identifier attribute of the Composition element that the reported data is most closely associated. compositionID is an optional attribute.	0..1

Attribute	Description	Occurrence
sampleRate	The rate at which successive samples of a data item are recorded by a piece of equipment. sampleRate is an optional attribute. sampleRate is expressed in terms of samples per second. If the sampleRate is smaller than one, the number can be represented as a floating point number. For example, a rate 1 per 10 seconds would be 0.1	0..1**
representation	Description of a means to interpret data consisting of multiple data points or samples reported as a single value. representation is an optional attribute. representation will define a unique format for each set of data. representation for TIME_SERIES, DISCRETE, and VALUE are defined below in *Section 7.2.2.12*. If representation is not specified, it **MUST** be determined to be VALUE.	0..1
significantDigits	The number of significant digits in the reported value. significantDigits is an optional attribute. This **SHOULD** be specified for all numeric values.	0..1

7.2.2.1 name Attribute for DataItem

The attribute name is provided as an additional human readable identifier for a data item. It is not required and is implementation dependent.

7.2.2.2 id Attribute for DataItem

Each DataItem element **MUST** be identified with an id. The id attribute **MUST** be unique across the entire MTConnectDevices document for a piece of equipment, including the identifiers for all *Structural Elements*. This unique id provides the information required by a client software application to uniquely identify each *Data Entity*.

For example, an XML document may provide three different *Data Entities* representing the position of the axes on a machine (x axis position, y axis position, and z axis position). All three may be modeled in the XML document as Position type data items for the Axes components. The unique id allows the client software application to distinguish the data for each of the axes.

7.2.2.3 type and subType Attributes for DataItem

The attribute type specifies the kind of data that is represented by the data item.

837 The attribute `type` **MUST** be specified for every data item.

838 A data item **MAY** further qualify the data being reported by specifying a `subType`. `subType`
839 is required for certain data item `types`. For example, `POSITION` has the `subType` of
840 `ACTUAL` and `PROGRAMMED`. Both data values can be represented in the document as two
841 separate and different `DataItem` XML elements – `POSITION` with `subType` `ACTUAL` and
842 `POSITION` with `subType` `PROGRAMMED`.

843 The `type` and `subType` **SHOULD** be used to further identify the meaning of the `DataItem`
844 associated with a `Component` element when a `subType` is applicable. There **SHOULD NOT**
845 be more than one `DataItem` with the same `type`, `subType`, and `compositionId` within a
846 `Component` element.

847 *Section 8* of this document provides a detailed listing of the data item `type` and `subType`
848 elements defined for each `category` of data item available for a piece of equipment: `SAMPLE`,
849 `EVENT`, and `CONDITION`.

850 #### 7.2.2.4 `statistic` Attribute for `DataItem`

851 A piece of equipment may further process some data types using a statistical calculation like
852 average, mean, or square root. In this case, the `statistic` attribute **MAY** be used to indicate
853 how the data was processed.

854 `statistic` may be defined for any `SAMPLE` type `DataItem`. All `statistic` data is
855 reported in the standard units of the `DataItem`.

856 `statistic` data is always the result of a calculation using data that has been measured over a
857 specified period of time.

858 The value of `statistic` may be periodically reset. When a piece of equipment reports a
859 `DataItem` with a value that is a `statistic`, the information provided in the XML document
860 for that *Data Entity* **MUST** include an additional attribute called `duration`. The attribute
861 `duration` defines the period of time over which the `statistic` has been calculated. Refer
862 to *Part 3.0 – Streams Information Model* of the MTConnect Standard for more information about
863 `duration`.

864 The following are the `statistic` calculations that can be defined for a `DataItem`.

865

Statistic	Description
`AVERAGE`	Mathematical Average value calculated for the data item during the calculation period.
`KURTOSIS`	A measure of the “peakedness” of a probability distribution; i.e., the shape of the distribution curve.

Statistic	Description
MAXIMUM	Maximum or peak value recorded for the data item during the calculation period.
MEDIAN	The middle number of a series of numbers.
MINIMUM	Minimum value recorded for the data item during the calculation period.
MODE	The number in a series of numbers that occurs most often.
RANGE	Difference between the Maximum and Minimum value of a data item during the calculation period. Also represents Peak-to-Peak measurement in a waveform.
ROOT_MEAN_SQUARE	Mathematical Root Mean Square (RMS) value calculated for the data item during the calculation period.
STANDARD_DEVIATION	Statistical Standard Deviation value calculated for the data item during the calculation period.

866

867 #### 7.2.2.5 `units` Attribute for `DataItem`

868 The following table lists the units that are defined as the standard unit of measure for each type
869 of `DataItem`. All `SAMPLE` type data items **MUST** report data values in standard units.

Units	Description
AMPERE	Amps
CELSIUS	Degrees Celsius
COUNT	A counted event
DECIBEL	Sound Level
DEGREE	Angle in degrees
DEGREE/SECOND	Angular degrees per second
DEGREE/SECOND^2	Angular acceleration in degrees per second squared
HERTZ	Frequency measured in cycles per second
JOULE	A measurement of energy.

Units	Description
KILOGRAM	Kilograms
LITER	Liters
LITER/SECOND	Liters per second
MICRO_RADIAN	Measurement of Tilt
MILLIMETER	Millimeters
MILLIMETER/SECOND	Millimeters per second
MILLIMETER/SECOND^2	Acceleration in millimeters per second squared
MILLIMETER_3D	A point in space identified by X, Y, and Z positions and represented by a space-delimited set of numbers each expressed in millimeters.
NEWTON	Force in Newtons
NEWTON_METER	Torque, a unit for force times distance.
OHM	Measure of Electrical Resistance
PASCAL	Pressure in Newtons per square meter
PASCAL_SECOND	Measurement of Viscosity
PERCENT	Percentage
PH	A measure of the acidity or alkalinity of a solution
REVOLUTION/MINUTE	Revolutions per minute
SECOND	A measurement of time.
SIEMENS/METER	A measurement of Electrical Conductivity
VOLT	Volts
VOLT_AMPERE	Volt-Ampere (VA)
VOLT_AMPERE_REACTIVE	Volt-Ampere Reactive (VAR)
WATT	Watts
WATT_SECOND	Measurement of electrical energy, equal to one Joule

870

7.2.2.6 `nativeUnits` Attribute for `DataItem`

The `nativeUnits` attribute provides additional information about the original measured value for a *Data Entity* reported by a piece of equipment. `nativeUnits` **MAY** be specified to provide additional information about the data if the units of the measured value supplied by the piece of equipment differ from the value provided for that data when converted to standard units.

The following table defines the `nativeUnits` currently supported by the `MTConnectDevices` XML document:

Native Units	Description
CENTIPOISE	A measure of Viscosity
DEGREE/MINUTE	Rotational velocity in degrees per minute
FAHRENHEIT	Temperature in Fahrenheit
FOOT	Feet
FOOT/MINUTE	Feet per minute
FOOT/SECOND	Feet per second
FOOT/SECOND^2	Acceleration in feet per second squared
FOOT_3D	A point in space identified by X, Y, and Z positions and represented by a space-delimited set of numbers each expressed in feet.
GALLON/MINUTE	Gallons per minute.
INCH	Inches
INCH/MINUTE	Inches per minute
INCH/SECOND	Inches per second
INCH/SECOND^2	Acceleration in inches per second squared
INCH_3D	A point in space identified by X, Y, and Z positions and represented by a space-delimited set of numbers each expressed in inches.
INCH_POUND	A measure of torque in inch pounds.
KELVIN	A measurement of temperature
KILOWATT	A measurement in kilowatt.

Native Units	Description
KILOWATT_HOUR	Kilowatt hours which is 3.6 mega joules.
LITER	Measurement of volume of a fluid
LITER/MINUTE	Measurement of rate of flow of a fluid
MILLIMETER/MINUTE	Velocity in millimeters per minute
POUND	US pounds
POUND/INCH^2	Pressure in pounds per square inch (PSI).
RADIAN	Angle in radians
RADIAN/SECOND	Velocity in radians per second
RADIAN/SECOND^2	Rotational acceleration in radian per second squared
RADIAN/MINUTE	Velocity in radians per minute.
REVOLUTION/SECOND	Rotational velocity in revolution per second
OTHER	Unsupported units

879

880 #### 7.2.2.7 `nativeScale` Attribute for `DataItem`

881 The units of measure for some measured values may be different from the `nativeUnits`
882 defined in *Section 7.2.2.8* above. In the cases where the units of measure use a different
883 weighting or range than is provided by `nativeUnits`, the `nativeScale` attribute can be
884 used to define the original units of measure.

885 As an example, a velocity measured in units of 100 ft/min can be represented as
886 `nativeUnits="FEET/MINUTE"` and `nativeScale="100"`.

887 #### 7.2.2.8 `category` Attribute for `DataItem`

888 Many `DataItem` types provide two forms of data, a value (reported as either a `SAMPLE` or
889 `EVENT` category) and a health status (reported as a `CONDITION` category). Therefore, each
890 occurrence of a `DataItem` in the XML document **MUST** report a `category` attribute. This
891 `category` attribute provides the information required by a client software application to
892 determine the specific meaning of the data provided.

893 Each *Data Entity* provided by a piece of equipment **MUST** be identified with one of the
894 following:

SAMPLE A SAMPLE is the reading of the value of a continuously variable or analog data value. A continuous value can be measured at any point-in-time and will always produce a result. An example of a continuous data value is the position of the Linear X Axis.

The data provided for a SAMPLE category data item is always a floating point number or integers that have an infinite number of possible values. This is different from a state or discrete type data item that has a limited number of possible values. A data item of category SAMPLE **MUST** also provide the units attribute.

EVENT An EVENT is a data item representing a discrete piece of information from the piece of equipment. EVENT does not have intermediate values that vary over time, as does SAMPLE. An EVENT is information that, when provided at any specific point in time, represents the current state of the piece of equipment.

There are two types of EVENT: those representing state, with two or more discrete values, and those representing messages that contain plain text data.

An example of a state type EVENT is the value of the data item DOOR_STATE, which can be OPEN, UNLATCHED, or CLOSED. (Note: No other values are valid to represent the value of DOOR_STATE.)

An example of a message type EVENT is the value for a data item PROGRAM. The value representing PROGRAM can be any valid string of characters.

CONDITION A CONDITION is a data item that communicates information about the health of a piece of equipment and its ability to function. A valid value for a data item in the category CONDITION can be one of NORMAL, WARNING, or FAULT.

A data item of category CONDITION **MAY** report multiple values (CONDITION) at one time whereas a data item of category SAMPLE or EVENT can only have a single value at any one point in time.

7.2.2.9 coordinateSystem Attribute for DataItem

The values reported by a piece of equipment for some types of data will be associated to a specific positioning measurement system used by the equipment. The coordinateSystem attribute **MAY** be used to specify the coordinate system used for the measured value.

The coordinateSystem attribute is used by a client software application to interpret the spatial relationship between values reported by a piece of equipment.

If coordinateSystem is not provided, all values representing positional data for Axes **MUST** be interpreted using the MACHINE coordinate system and all values representing positional data for Path **MUST** be interpreted using the WORK coordinate system.

The following table defines the types of coordinateSystem currently supported by the MTConnectDevices XML document:

Coordinate System	Description
MACHINE	An unchangeable coordinate system that has machine zero as its origin.
WORK	The coordinate system that represents the working area for a particular workpiece whose origin is shifted within the MACHINE coordinate system. If the WORK coordinates are not currently defined in the piece of equipment, the MACHINE coordinates will be used.

7.2.2.10 compositionId Attribute for DataItem

compositionId attribute identifies the id of the Composition element where the reported data is most closely associated.

An example would be a TEMPERATURE associated with a Linear type axis may be further clarified by referencing the MOTOR or AMPLIFIER type Composition element associated with that axis, which differentiates the temperature of the motor from the temperature of the amplifier.

The compositionId attribute provides the information required by a client software application to interpret the data with a greater specificity and to disambiguate between multiple *Data Entities* of the same data type associated with a Component element.

7.2.2.11 sampleRate Attribute for DataItem

The value for some data types provided by a piece of equipment may be reported as a single set of data containing a series of values that have been recorded at a fixed sample rate. When such data is reported, the sampleRate defines the rate at which successive samples of data were recorded.

The sampleRate attribute provides the information required by a client software application to interpret the data and the sampling time relationship between successive values contained in the set of data.

sampleRate is expressed in terms of samples per second. If the sample rate is smaller than one, the number can be represented as a floating point number. For example, a rate 1 per 10 seconds would be 0.1

7.2.2.12 `representation` Attribute for `DataItem`

Some data types provide data that may consist of a series of values or a file of data, not a single value. Other data types provide a series of data values that may require additional information so that the data may be correctly understood by a client software application.

When such data is provided, the `representation` attribute **MUST** be used to define the format for the data provided.

The types of `representation` defined are provided in the table below.

Note: See *Part 3.0 - Streams Information Model* of the MTConnect Standard for more information on the structure and format of each `representation`.

Representation	Description
`VALUE`	The measured value of the sampled data. If no representation is specified for a data item, the representation **MUST** be determined to be VALUE.
`TIME_SERIES`	A series of sampled data. The data is reported for a specified number of samples and each sample is reported with a fixed period.
`DISCRETE`	A *Data Entity* where each discrete occurrence of the data may have the same value as the previous occurrence of the data. There is no reported state change between occurrences of the data. In this case, duplicate occurrences of the same data value **SHOULD NOT** be suppressed. An example of a `DISCRETE` data type would be a parts counter that reports the completion of each part versus the accumulation of parts. Another example would be a `Message` that does not typically have a reset state and may re-occur each time a specific message is triggered.

7.2.2.13 `significantDigits` Attribute for `DataItem`

`significantDigits` is used to specify the level of precision (number of significant digits) for the value provided for a data item.

`significantDigits` attribute is not required for a data item, but it is recommended and **SHOULD** be used for any data item reporting a numeric value.

7.2.3 Elements for DataItem

The following table lists the elements defined to provide additional information for a DataItem type XML element.

Element	Description	Occurrence
Source	Source is an optional XML element that identifies the Component, DataItem, or Composition representing the part of the piece of equipment from which a measured value originates.	0..1
Constraints	Constraints is an optional container that provides a set of expected values that can be reported for this DataItem. Constraints are used by a software application to evaluate the validity of the reported data.	0..1
Filters	An optional container for the Filter elements associated with this DataItem element.	0..1
InitialValue	InitialValue is an optional XML element that defines the starting value for a data item as well as the value to be set for the data item after a reset event. Only one InitialValue element may be defined for a data item. The value will be constant and cannot change. If no InitialValue element is defined for a data item that is periodically reset, then the starting value for the data item **MUST** be a value of 0.	0..1
ResetTrigger	ResetTrigger is an optional XML element that identifies the type of event that may cause a reset to occur. It is additional information regarding the meaning of the data that establishes an understanding of the time frame that the data represents so that the data may be correctly understood by a client software application.	0..1

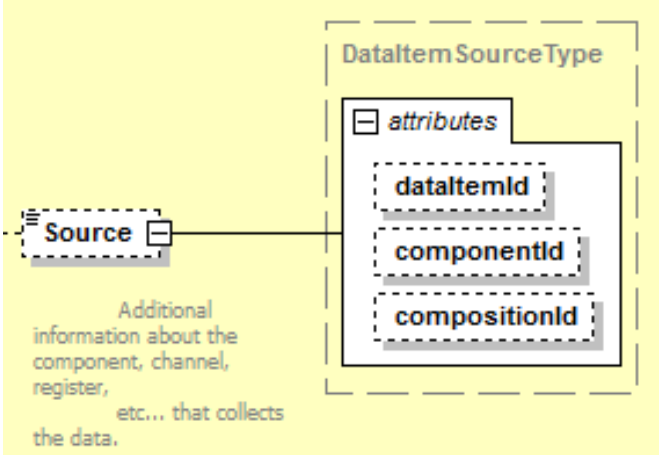
7.2.3.1 Source Element for DataItem

Source is an optional XML element that identifies the physical part of a piece of equipment where the data represented by DataItem originated.

As an example, data related to a servo motor on an Axes component may actually originate from a measurement made in the Controller element.

In the case where the real name associated with a DataItem element is either complex or does not meet the format requirements of a NMTOKEN XML type, the real name of the element may not be able to be expressed in the name attribute. When this occurs, a short name or nickname can be used for the name attribute and the real name can be provided as the CDATA for Source.

988 The following XML schema represents the structure of the `Source` XML element showing the
989 attributes defined for `Source`.
990



991
992 **Figure 17: `Source` Schema Diagram**

993

994 #### 7.2.3.1.1 Attributes for `Source`

995 The following table identifies the attributes available to identify `Source` for a measured value:

Attribute	Description	Occurrence
componentId	The identifier attribute of the Component element that represents the physical part of a piece of equipment where the data represented by the DataItem element originated. A valid data value reported for componentId **MUST** be the value of the Id attribute for the Component element identified. componentId is an optional attribute.	0..1*
dataItemId	The identifier attribute of the DataItem that represents the originally measured value of the data referenced by this data item. A valid data value reported for dataItemId **MUST** be the value of the Id attribute for the DataItem element identified. dataItemId is an optional attribute.	0..1*
compositionId	The identifier attribute of the Composition element that represents the physical part of a piece of equipment where the data represented by the DataItem element originated. A valid data value reported for compositionId **MUST** be the value of the Id attribute for the Composition element identified. compositionId is an optional attribute.	0..1*

Note: * One of componentId, componsitionId, or dataItemId **MUST** be provided.

7.2.3.2 Constraints Element for DataItem

For some types of DataItem elements, the expected value(s) for the data reported for the DataItem **MAY** be restricted to specific values or a range of values.

Constraints is an optional XML element that provides a way to define the expected value(s) or the upper and lower limits for the range of values that are expected to be reported in response to a Current or Sample request.

Constraints are used by a software application to evaluate the validity of the data reported.

The value associated with each Constraint element is reported in the CDATA for that element.

7.2.3.2.1 Schema for `Constraints`

The following XML schema represents the structure of the `Constraints` XML element and the elements defined for `Constraints`.

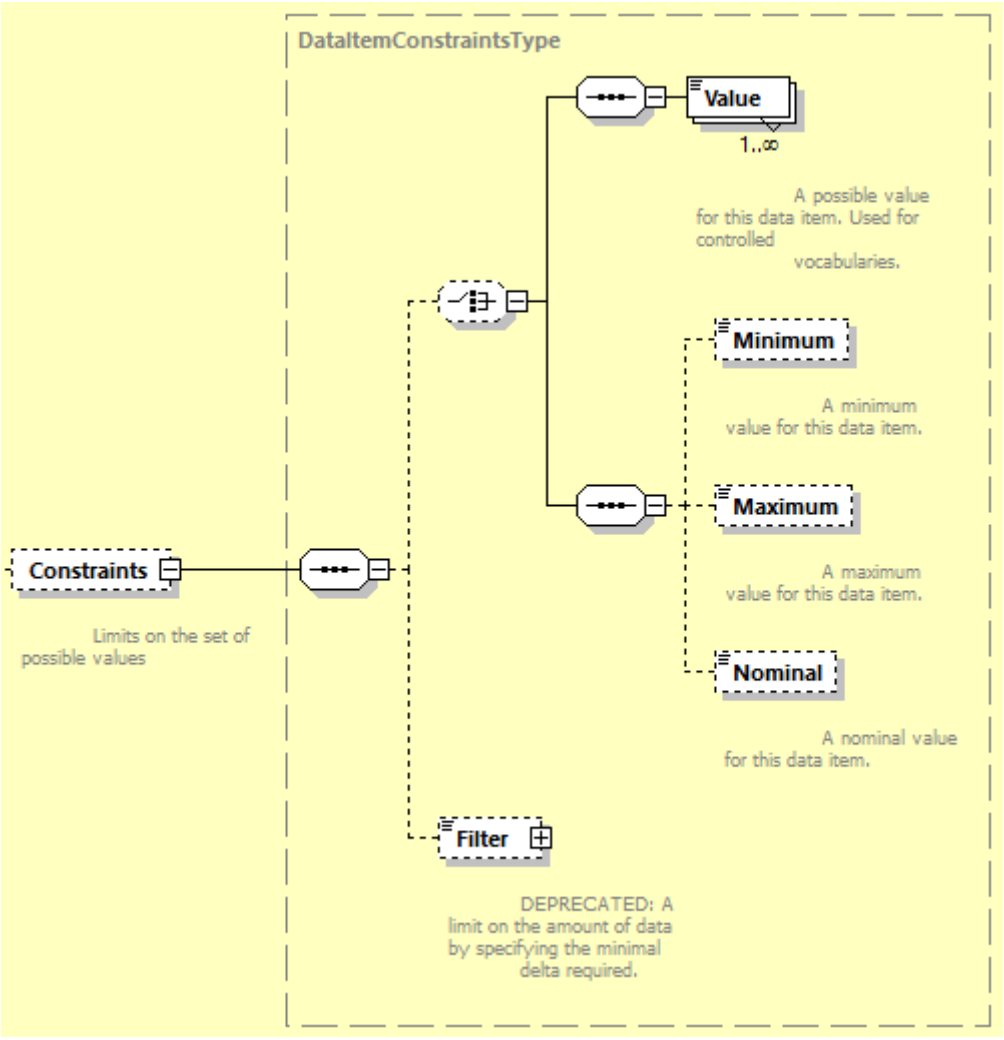


Figure 18: `Constraints` Schema Diagram

1017 The following table identifies the elements available to identify `Constraints` for a measured
1018 value:

1019

Element	Description	Occurrence
`Value`	`Value` represents a single data value that is expected to be reported for a `DataItem` element. The data value is provided in the CDATA for this element and may be any numeric or text content. When there are multiple data values that may be expected to be reported for a `DataItem` element, multiple `Value` elements may be defined. In the case where only one `Value` element is defined, the data returned in response to a `Current` or `Sample` request **MUST** be the data value defined for `Value` element. `Value` **MUST NOT** be used in conjunction with any other `Constraint` elements.	0..INF
`Maximum`	If the data reported for a data item is a range of numeric values, the expected value reported **MAY** be described with an upper limit defined by this constraint. The data value is provided in the `CDATA` for this element and **MUST** be an absolute value using the same units as the reported data.	0..1
`Minimum`	If the data reported for a data item is a range of numeric values, the expected value reported **MAY** be described with a lower limit defined by this constraint. The data value is provided in the `CDATA` for this element and **MUST** be an absolute value using the same units as the reported data.	0..1
`Nominal`	The target or expected value for this data item. The data value is provided in the `CDATA` for this element and **MUST** be an absolute value using the same units as the reported data.	0..1
~~`Filter`~~	**DEPRECATED** in *Version 1.4* – Moved to the `Filters` element of a `DataItem`. ~~If the data reported for a DataItem is a numeric value, a new value MUST NOT be reported if the change from the last reported value is less than the delta given as the CDATA of this element. Filter is an abstract type XML element. As such, Filter will never appear in the XML document, but will be replaced by a Filter type. The only currently supported Filter type is MINIMUM_DELTA. The CDATA MUST be an absolute value using the same Units as the reported data. Additional filter types MAY be supported in the future.~~	~~0..1*~~

1020 Note: * Remains in schema for backwards compatibility.

7.2.3.3 `Filters` Element for `DataItem`

`Filters` is an optional XML container that organizes the `Filter` elements for `DataItem`.

`Filters` contains one or more `Filter` XML elements.

Element	Description	Occurrence
`Filters`	An XML container consisting of one or more types of `Filter` XML elements. Only one `Filters` container **MAY** appear for a `DataItem` element.	0..1

7.2.3.3.1 Filter

`Filter` provides a means to control when a *MTConnect Agent* records updated information for a data item. Currently, there are two types of `Filter` elements defined in the MTConnect Standard - `MINIMUM_DELTA` and `PERIOD`. More `Filter` types may be added in the future.

The value associated with each `Filter` element is reported in the `CDATA` for that element.

The following XML schema represents the structure for `Filter` XML element.

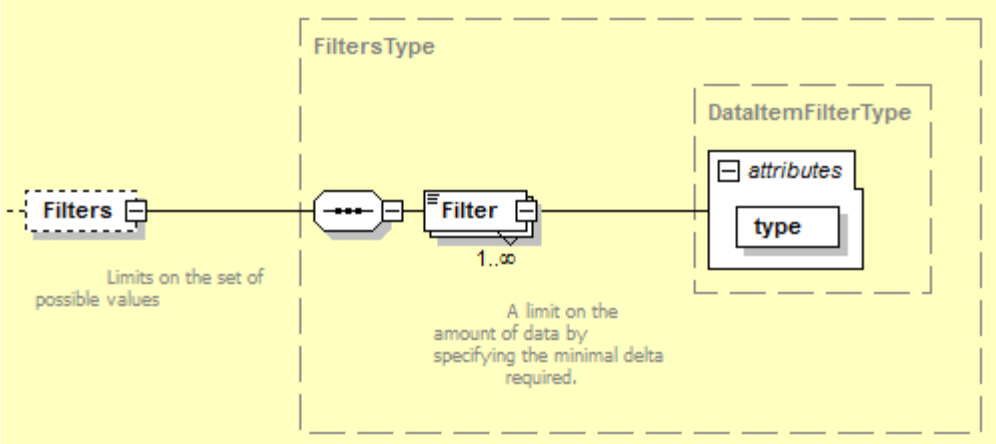


Figure 19: `Filter` Schema Diagram

The following table describes the types of `Filter` defined for a `DataItem` element and the expected behavior of a *MTConnect Agent* when a `Filter` is applied to `DataItem` element.

Type	Description	Occurrence
`MINIMUM_DELTA`	For a `MINIMUM_DELTA` type `Filter`, a new value **MUST NOT** be reported for a data item unless the measured value has changed from the last reported value by at least the delta given as the `CDATA` of this element. The `CDATA` **MUST** be an absolute value using the same units as the reported data.	0..1 *
`PERIOD`	For a `PERIOD` type `Filter`, the data reported for a data item is provided on a periodic basis. The `PERIOD` for reporting data is defined in the `CDATA` for the `Filter`. The `CDATA` **MUST** be an absolute value reported in seconds representing the time between reported samples of the value of the data item. If the `PERIOD` is smaller than one second, the number can be represented as a floating point number. For example, a `PERIOD` of 100 milliseconds would be 0.1	0..1 *

Note: * Either `MINIMUM_DELTA` or `PERIOD` can be defined, not both.

7.2.3.4 `InitialValue` Element for `DataItem`

`InitialValue` is an XML element that defines the value to be set for the data item after a reset event.

The value associated with the `InitialValue` element is reported in the `CDATA` for this element and **MUST** be an absolute value using the same units as the reported data.

7.2.3.5 `ResetTrigger` Element for `DataItem`

The value of some data types is periodically reset to the value of the `InitialValue` element. These reset events may be based upon a specific elapsed time or may be triggered by a physical or logical reset action that causes the reset to occur. `ResetTrigger` provides additional information regarding the meaning of the data – establishing an understanding of the time frame that the data represents so that the data may be correctly understood by a client software application.

Element	Description	Occurrence
`ResetTrigger`	`ResetTrigger` is an XML element that describes the reset action that causes a reset to occur. It is additional information regarding the meaning of the data that establishes an understanding of the time frame that the data represents so that the data may be correctly understood by a client software application.	0..1

The reset action that **MAY** cause a reset to occur is provided in the `CDATA` for this element.

The reset actions that may cause a reset to occur are described in the following table.

Reset Actions	Description
`ACTION_COMPLETE`	The value of the *Data Entity* that is measuring an action or operation is to be reset upon completion of that action or operation.
`ANNUAL`	The value of the *Data Entity* is to be reset at the end of a 12-month period.
`DAY`	The value of the *Data Entity* is to be reset at the end of a 24-hour period.
`LIFE`	The value of the data item is not reset and accumulates for the entire life of the piece of equipment.
`MAINTENANCE`	The value of the data item is to be reset upon completion of a maintenance event.
`MONTH`	The value of the *Data Entity* is to be reset at the end of a monthly period.
`POWER_ON`	The value of the *Data Entity* is to be reset when power was applied to the piece of equipment after a planned or unplanned interruption of power has occurred.
`SHIFT`	The value of the *Data Entity* is to be reset at the end of a work shift.
`WEEK`	The value of the *Data Entity* is to be reset at the end of a 7-day period.

8 Listing of Data Items

In the MTConnect Standard, DataItem elements are defined and organized based upon the category and type attributes.

The category attribute provides a high level grouping for DataItem elements based on the kind of information that is reported by the data item.

These categories are:

SAMPLE A SAMPLE reports a continuously variable or analog data value.

EVENT An EVENT reports information representing a functional state, with two or more discrete values, associated with a component or it contains a message. The data provided may be a numeric value or text.

CONDITION A CONDITION reports information about the health of a piece of equipment and its ability to function.

The type attribute specifies the specific kind of data that is reported. For some types of data items, a subType attribute may also be used to differentiate between multiple data items of the same type where the information reported by the data item has a different, but related, meaning.

Many types of data items provide two forms of data: a value (reported as either a SAMPLE or EVENT) and a health status (reported as a CONDITION). These DataItem types **MAY** be defined in more than one category based on the data that they report.

The following sections define the types and subtypes of DataItem elements that are defined for each of the above categories.

8.1 Data Items in category SAMPLE

The types of DataItem elements in the SAMPLE category report data representing a continuously changing or analog data value. This data can be measured at any point-in-time and will always produce a result. The data provided may be a scalar floating point number or integers that have an infinite number of possible values. The units attribute **MUST** be defined and reported for each DataItem in this category.

1083 The table below defines the types and subtypes of DataItem elements defined for the
1084 SAMPLE category. The subtypes are indented below their associated types.

1085

DataItem type/subType	Description	Units
ACCELERATION	Rate of change of velocity	MILLIMETER/SECOND^2
ACCUMULATED_TIME	The measurement of accumulated time for an activity or event. **DEPRECATION WARNING:** May be deprecated in the future. Recommend using PROCESS_TIMER and MACHINE_TIMER.	SECOND
ANGULAR_ACCELERATION	Rate of change of angular velocity.	DEGREE/SECOND^2
ANGULAR_VELOCITY	Rate of change of angular position.	DEGREE/SECOND
AMPERAGE	The measurement of electrical current	AMPERE
ALTERNATING	The measurement of alternating current. If not specified further in statistic, defaults to RMS current	AMPERE
DIRECT	The measurement of DC current	AMPERE
ACTUAL	The measured amperage being delivered from a power source.	AMPERE
TARGET	The desired or preset amperage to be delivered from a power source.	AMPERE
ANGLE	The measurement of angular position	DEGREE
ACTUAL	The actual angular position as read from the physical component.	DEGREE
COMMANDED	A calculated value for angular position computed by the Controller type component	DEGREE
AXIS_FEEDRATE	The feedrate of a linear axis.	MILLIMETER/SECOND
ACTUAL	The measured value of the feedrate of a linear axis.	MILLIMETER/SECOND

DataItem type/subType	Description	Units
COMMANDED	The feedrate of a linear axis as specified by the Controller type component. The COMMANDED feedrate is a calculated value that includes adjustments and overrides.	MILLIMETER/SECOND
JOG	The feedrate specified by a logic or motion program, by a pre-set value, or set by a switch as the feedrate for a linear axis when operating in a manual state or method (jogging).	MILLIMETER/SECOND
PROGRAMMED	The feedrate specified by a logic or motion program or set by a switch for a linear axis.	MILLIMETER/SECOND
RAPID	The feedrate specified by a logic or motion program, by a pre-set value, or set by a switch as the feedrate for a linear axis when operating in a rapid positioning mode.	MILLIMETER/SECOND
~~OVERRIDE~~	~~The operator's overridden value. Percent of commanded~~. **DEPRECATED** in *Version 1.3*. See EVENT category data items.	~~PERCENT~~
CLOCK_TIME	The value provided by a timing device at a specific point in time. CLOCK_TIME **MUST** be reported in W3C ISO 8601 format.	YYYY-MM-DDThh:mm:ss.ffff
CONCENTRATION	Percentage of one component within a mixture of components	PERCENT
CONDUCTIVITY	The ability of a material to conduct electricity	SIEMENS/METER
DISPLACEMENT	The change in position of an object	MILLIMETER
ELECTRICAL_ENERGY	The measurement of electrical energy consumption by a component	WATT_SECOND

DataItem type/subType	Description	Units
EQUIPMENT_TIMER	The measurement of the amount of time a piece of equipment or a sub-part of a piece of equipment has performed specific activities. Often used to determine when maintenance may be required for the equipment Multiple subTypes of EQUIPMENT_TIMER **MAY** be defined. A subType **MUST** always be specified.	SECOND
LOADED	Measurement of the time that the sub-parts of a piece of equipment are under load. Example: For traditional machine tools, this is a measurement of the time that the cutting tool is assumed to be engaged with the part.	SECOND
WORKING	Measurement of the time that a piece of equipment is performing any activity – the equipment is active and performing a function under load or not. Example: For traditional machine tools, this includes LOADED, plus rapid moves, tool changes, etc.	SECOND
OPERATING	Measurement of the time that the major sub-parts of a piece of equipment are powered or performing any activity whether producing a part or product or not. Example: For traditional machine tools, this includes WORKING, plus idle time.	SECOND
POWERED	The measurement of time that primary power is applied to the piece of equipment and, as a minimum, the controller or logic portion of the piece of equipment is powered and functioning or components that are required to remain on are powered. Example: Heaters for an extrusion machine that are required to be powered even when the equipment is turned off.	SECOND
DELAY	Measurement of the time that a piece of equipment is waiting for an event or an action to occur.	SECOND

DataItem type/subType	Description	Units
FILL_LEVEL	The measurement of the amount of a substance remaining compared to the planned maximum amount of that substance	PERCENT
FLOW	The rate of flow of a fluid	LITER/SECOND
FREQUENCY	The measurement of the number of occurrences of a repeating event per unit time	HERTZ
~~GLOBAL_POSITION~~	**DEPRECATED** in *Version 1.1*	
~~LEVEL~~	**DEPRECATED** in *Version 1.2*. See FILL_LEVEL	
LENGTH	The length of an object	MILLIMETER
STANDARD	The standard or original length of an object.	MILLIMETER
REMAINING	The remaining total length of an object.	MILLIMETER
USEABLE	The remaining useable length of an object.	MILLIMETER
LINEAR_FORCE	The measure of the push or pull introduced by an actuator or exerted on an object.	NEWTON
LOAD	The measurement of the actual versus the standard rating of a piece of equipment.	PERCENT
MASS	The measurement of the mass of an object(s) or an amount of material.	KILOGRAM
PATH_FEEDRATE	The feedrate for the axes, or a single axis, associated with a Path component– a vector.	MILLIMETER/SECOND
ACTUAL	The measured value of the feedrate of the axes, or a single axis, associated with a Path component.	MILLIMETER/SECOND
COMMANDED	The feedrate as specified by the Controller type component for the axes, or a single axis, associated with a Path component. The COMMANDED feedrate is a calculated value that includes adjustments and overrides.	MILLIMETER/SECOND

DataItem type/subType	Description	Units
JOG	The feedrate specified by a logic or motion program, by a pre-set value, or set by a switch as the feedrate for the axes, or a single axis, associated with a Path when operating in a manual state or method (jogging).	MILLIMETER/SECOND
PROGRAMMED	The feedrate specified by a logic or motion program or set by a switch as the feedrate for the axes, or a single axis, associated with a Path.	MILLIMETER/SECOND
RAPID	The feedrate specified by a logic or motion program, by a pre-set value, or set by a switch as the feedrate for the axes, or a single axis, associated with a Path when operating in a rapid positioning mode.	MILLIMETER/SECOND
~~OVERRIDE~~	~~The operator's overridden value. Percent of commanded.~~ **DEPRECATED** in *Version 1.3*. See EVENT category DataItems.	~~PERCENT~~
PATH_POSITION	A measured or calculated position of a control point associated with a CONTROLLER element, or PATH element if provided, of a piece of equipment. The control point **MUST** be reported as a set of space-delimited floating-point numbers representing a point in 3-D space. The position of the control point **MUST** be reported in units of MILLIMETER and listed in order of X, Y, and Z referenced to the coordinate system of the piece of equipment. Any control point representing a position in 1-D or 2-D space **MAY** be represented in terms of 3-D space by setting any undefined coordinate to zero (0). PATH_POSITION **SHOULD** be further defined with a coordinateSytem attribute. If a coordinateSystem attribute is not specified, the position of the control point **MUST** be reported in WORK coordinates.	MILLIMETER_3D
ACTUAL	The measured position of the current program control point as reported by the piece of equipment.	MILLIMETER_3D

DataItem type/subType	Description	Units
COMMANDED	The position computed by the Controller type component.	MILLIMETER_3D
TARGET	The desired end position for a movement or a series of movements. Multiple discrete movements may need to be completed to achieve the final TARGET position.	MILLIMETER_3D
PROBE	The position provided by a measurement probe.	MILLIMETER_3D
PH	The measure of the acidity or alkalinity.	PH
POSITION	A calculated or measured position related to a Component element. POSITION **SHOULD** be further defined with a coordinateSytem attribute. If a coordinateSystem attribute is not specified, the position of the control point **MUST** be reported in MACHINE coordinates.	MILLIMETER
ACTUAL	The physical measured position of the control point for a Component.	MILLIMETER
COMMANDED	A position calculated by the Controller type component for a discrete movement.	MILLIMETER
PROGRAMMED	The position of the control point for a Component specified by a logic or motion program	MILLIMETER
TARGET	The desired end position of the control point for a Component resulting from a movement or a series of movements. Multiple discrete movements may need to be completed to achieve the final TARGET position.	MILLIMETER
POWER_FACTOR	The measurement of the ratio of real power flowing to a load to the apparent power in that AC circuit.	PERCENT
PRESSURE	The force per unit area exerted by a gas or liquid	PASCAL

DataItem type/subType	Description	Units
PROCESS_TIMER	The measurement of the amount of time a piece of equipment has performed different types of activities associated with the process being performed at that piece of equipment. Multiple subtypes of PROCESS_TIMER may be defined. Typically, PROCESS_TIMER **SHOULD** be modeled as a data item for the Device element, but **MAY** be modeled for either a Controller or Path *Structural Element* in the XML document. A subType **MUST** always be specified.	SECOND
PROCESS	The measurement of the time from the beginning of production of a part or product on a piece of equipment until the time that production is complete for that part or product on that piece of equipment. This includes the time that the piece of equipment is running, producing parts or products, or in the process of producing parts.	SECOND
DELAY	Measurement of the time that a process is waiting and unable to perform its intended function.	SECOND
RESISTANCE	The degree to which a substance opposes the passage of an electric current.	OHM
ROTARY_VELOCITY	The rotational speed of a rotary axis.	REVOLUTION/MINUTE
ACTUAL	The measured value of rotational speed that the rotary axis is spinning.	REVOLUTION/MINUTE
COMMANDED	The rotational speed as specified by the Controller type component. The COMMANDED velocity is a calculated value that includes adjustments and overrides.	REVOLUTION/MINUTE
PROGRAMMED	The rotational velocity specified by a logic or motion program or set by a switch	REVOLUTION/MINUTE
~~OVERRIDE~~	~~The operator's overridden value. Percent of commanded.~~ **DEPRECATED** in *Version 1.3*. See EVENT category DataItems.	~~PERCENT~~

DataItem type/subType	Description	Units
SOUND_LEVEL	Measurement of a sound level or sound pressure level relative to atmospheric pressure	DECIBEL
NO_SCALE	No weighting factor on the frequency scale	DECIBEL
A_SCALE	A Scale weighting factor. This is the default weighting factor if no factor is specified	DECIBEL
B_SCALE	B Scale weighting factor	DECIBEL
C_SCALE	C Scale weighting factor	DECIBEL
D_SCALE	D Scale weighting factor	DECIBEL
~~SPINDLE_SPEED~~	**DEPRECATED** in *Version 1.2*. Replaced by ROTARY_VELOCITY	
~~ACTUAL~~	~~The rotational speed of a rotary axis. ROTARY_MODE **MUST** be SPINDLE.~~	~~REVOLUTION/MINUTE~~
~~COMMANDED~~	~~The rotational speed the as specified by the Controller type Component.~~	~~REVOLUTION/MINUTE~~
~~OVERRIDE~~	~~The operator's overridden value. Percent of commanded.~~	~~PERCENT~~
STRAIN	The amount of deformation per unit length of an object when a load is applied.	PERCENT
TEMPERATURE	The measurement of temperature	CELSIUS
TENSION	A measurement of a force that stretches or elongates an object	NEWTON
TILT	A measurement of angular displacement	MICRO_RADIAN
TORQUE	The turning force exerted on an object or by an object	NEWTON_METER
VOLT_AMPERE	The measure of the apparent power in an electrical circuit, equal to the product of root-mean-square (RMS) voltage and RMS current (commonly referred to as VA)	VOLT_AMPERE
VOLT_AMPERE_REACTIVE	The measurement of reactive power in an AC electrical circuit (commonly referred to as VAR)	VOLT_AMPERE_REACTIVE

DataItem type/subType	Description	Units
VELOCITY	The rate of change of position.	MILLIMETER/SECOND
VISCOSITY	A measurement of a fluid's resistance to flow	PASCAL_SECOND
VOLTAGE	The measurement of electrical potential between two points	VOLT
ALTERNATING	The measurement of alternating voltage. If not specified further in statistic, defaults to RMS voltage	VOLT
DIRECT	The measurement of DC voltage	VOLT
ACTUAL	The measured voltage being delivered from a power source.	VOLT
TARGET	The desired or preset voltage to be delivered from a power source.	VOLT
WATTAGE	The measurement of power flowing through or dissipated by an electrical circuit or piece of equipment.	WATT
ACTUAL	The measured wattage being delivered from a power source.	WATT
TARGET	The desired or preset wattage to be delivered from a power source.	WATT

8.2 Data Items in category EVENT

DataItem types in the EVENT category represent a discrete piece of information from a piece of equipment. EVENT does not have intermediate values that vary over time.

An EVENT is information that, when provided at any specific point in time, represents the current state of the piece of equipment.

There are two types of EVENT: those representing state, with two or more discrete values, and those representing messages that contain plain text data.

The table below defines the DataItem types and subtypes defined for the EVENT category. The subtypes are indented below their associated types.

DataItem type/subType	Description
ACTUATOR_STATE	Represents the operational state of an apparatus for moving or controlling a mechanism or system. The valid data value **MUST** be ACTIVE or INACTIVE.
~~ALARM~~	**DEPRECATED in *Version 1.1*.** Replaced with CONDITION category.
ACTIVE_AXES	The set of axes currently associated with a Path or Controller *Structural Element*. If this DataItem is not provided, it will be assumed that all axes are currently associated with the Controller *Structural Element* and with an individual Path. The valid data value for ACTIVE_AXES **SHOULD** be a space-delimited set of axes reported as the value of the name attribute for each axis. If name is not available, the piece of equipment **MUST** report the value of the nativeName attribute for each axis.
AVAILABILITY	Represents the *Agent's* ability to communicate with the data source. This **MUST** be provided for a Device Element and **MAY** be provided for any other *Structural Element*. The valid data value **MUST** be AVAILABLE or UNAVAILABLE.
AXIS_COUPLING	Describes the way the axes will be associated to each other. This is used in conjunction with COUPLED_AXES to indicate the way they are interacting. The valid data value **MUST** be TANDEM, SYNCHRONOUS, MASTER, and SLAVE. The coupling **MUST** be viewed from the perspective of a specific axis. Therefore, a MASTER coupling indicates that this axis is the master for the COUPLED_AXES.
AXIS_FEEDRATE_OVERRIDE	The value of a signal or calculation issued to adjust the feedrate of an individual linear type axis. The value provided for AXIS_FEEDRATE_OVERRIDE is expressed as a percentage of the designated feedrate for the axis. When AXIS_FEEDRATE_OVERRIDE is applied, the resulting commanded feedrate for the axis is limited to the value of the original feedrate multiplied by the value of the AXIS_FEEDRATE_OVERRIDE. There **MAY** be different subtypes of AXIS_FEEDRATE_OVERRIDE; each representing an override value for a designated subtype of feedrate depending on the state of operation of the axis. The subtypes of operation of an axis are currently defined as PROGRAMMED, JOG, and RAPID.

DataItem type/subType	Description
JOG	The value of a signal or calculation issued to adjust the feedrate of an individual linear type axis when that axis is being operated in a manual state or method (jogging). When the JOG subtype of AXIS_FEEDRATE_OVERRIDE is applied, the resulting commanded feedrate for the axis is limited to the value of the original JOG subtype of the AXIS_FEEDRATE multiplied by the value of the JOG subtype of AXIS_FEEDRATE_OVERRIDE.
PROGRAMMED	The value of a signal or calculation issued to adjust the feedrate of an individual linear type axis that has been specified by a logic or motion program or set by a switch. When the PROGRAMMED subtype of AXIS_FEEDRATE_OVERRIDE is applied, the resulting commanded feedrate for the axis is limited to the value of the original PROGRAMMED subtype of the AXIS_FEEDRATE multiplied by the value of the PROGRAMMED subtype of AXIS_FEEDRATE_OVERRIDE.
RAPID	The value of a signal or calculation issued to adjust the feedrate of an individual linear type axis that is operating in a rapid positioning mode. When the RAPID subtype of AXIS_FEEDRATE_OVERRIDE is applied, the resulting commanded feedrate for the axis is limited to the value of the original RAPID subtype of the AXIS_FEEDRATE multiplied by the value of the RAPID subtype of AXIS_FEEDRATE_OVERRIDE.
AXIS_INTERLOCK	An indicator of the state of the axis lockout function when power has been removed and the axis is allowed to move freely. The valid data value **MUST** be ACTIVE or INACTIVE.
AXIS_STATE	An indicator of the controlled state of a LINEAR or ROTARY component representing an axis. The valid data value **MUST** be HOME, TRAVEL, PARKED, or STOPPED.
BLOCK	The line of code or command being executed by a Controller *Structural Element*. The value reported for Block **MUST** include the entire expression for a line of program code, including all parameters.
BLOCK_COUNT	The total count of the number of blocks of program code that have been executed since execution started. BLOCK_COUNT counts blocks of program code executed regardless of program structure (e.g., looping or branching within the program). The starting value for BLOCK_COUNT **MAY** be established by an initial value provided in the Constraint element defined for the data item.

DataItem type/subType	Description
CHUCK_INTERLOCK	An indication of the state of an interlock function or control logic state intended to prevent the associated CHUCK component from being operated. The valid data value **MUST** be ACTIVE or INACTIVE.
MANUAL_UNCLAMP	An indication of the state of an operator controlled interlock that can inhibit the ability to initiate an unclamp action of an electronically controlled chuck. The valid data value **MUST** be ACTIVE or INACTIVE. When MANUAL_UNCLAMP is ACTIVE, it is expected that a chuck cannot be unclamped until MANUAL_UNCLAMP is set to INACTIVE.
CHUCK_STATE	An indication of the operating state of a mechanism that holds a part or stock material during a manufacturing process. It may also represent a mechanism that holds any other mechanism in place within a piece of equipment. The valid data value **MUST** be OPEN, CLOSED, or UNLATCHED.
~~CODE~~	**DEPRECATED** in *Version 1.1*.
COMPOSITION_STATE	An indication of the operating condition of a mechanism represented by a Composition type element. A subType **MUST** always be specified. A compositionId **MUST** always be specified.
ACTION	An indication of the operating state of a mechanism represented by a Composition type component. The operating state indicates whether the Composition element is activated or disabled. The valid data value **MUST** be ACTIVE or INACTIVE.
LATERAL	An indication of the position of a mechanism that may move in a lateral direction. The mechanism is represented by a Composition type component. The position information indicates whether the Composition element is positioned to the right, to the left, or is in transition. The valid data value **MUST** be RIGHT, LEFT, or TRANSITIONING.
MOTION	An indication of the open or closed state of a mechanism. The mechanism is represented by a Composition type component. The operating state indicates whether the state of the Composition element is open, closed, or unlatched. The valid data value **MUST** be OPEN, UNLATCHED, or CLOSED.

DataItem type/subType	Description
SWITCHED	An indication of the activation state of a mechanism represented by a Composition type component. The activation state indicates whether the Composition element is activated or not. The valid data value **MUST** be ON or OFF.
VERTICAL	An indication of the position of a mechanism that may move in a vertical direction. The mechanism is represented by a Composition type component. The position information indicates whether the Composition element is positioned to the top, to the bottom, or is in transition. The valid data value **MUST** be UP, DOWN, or TRANSITIONING.
CONTROLLER_MODE	The current mode of the Controller component. The valid data value **MUST** be AUTOMATIC, MANUAL, MANUAL_DATA_INPUT, SEMI_AUTOMATIC, or EDIT.
CONTROLLER_MODE_OVERRIDE	A setting or operator selection that changes the behavior of a piece of equipment. A subType **MUST** always be specified.
DRY_RUN	A setting or operator selection used to execute a test mode to confirm the execution of machine functions. The valid data value **MUST** be ON or OFF. When DRY_RUN is ON, the equipment performs all of its normal functions, except no part or product is produced. If the equipment has a spindle, spindle operation is suspended.
SINGLE_BLOCK	A setting or operator selection that changes the behavior of the controller on a piece of equipment. The valid data value **MUST** be ON or OFF. Program execution is paused after each BLOCK of code is executed when SINGLE_BLOCK is ON. When SINGLE_BLOCK is ON, EXECUTION **MUST** change to INTERRUPTED after completion of each BLOCK of code.
MACHINE_AXIS_LOCK	A setting or operator selection that changes the behavior of the controller on a piece of equipment. The valid data value **MUST** be ON or OFF. When MACHINE_AXIS_LOCK is ON, program execution continues normally, but no equipment motion occurs

DataItem type/subType	Description
OPTIONAL_STOP	A setting or operator selection that changes the behavior of the controller on a piece of equipment. The valid data value **MUST** be ON or OFF. The program execution is stopped after a specific program block is executed when OPTIONAL_STOP is ON. In the case of a G-Code program, a program BLOCK containing a M01 code designates the command for an OPTIONAL_STOP. EXECUTION **MUST** change to OPTIONAL_STOP after a program block specifying an optional stop is executed and the OPTIONAL_STOP selection is ON.
TOOL_CHANGE_STOP	A setting or operator selection that changes the behavior of the controller on a piece of equipment. The valid data value **MUST** be ON or OFF. Program execution is paused when a command is executed requesting a cutting tool to be changed. EXECUTION **MUST** change to INTERRUPTED after completion of the command requesting a cutting tool to be changed and TOOL_CHANGE_STOP is ON.
COUPLED_AXES	Refers to the set of associated axes. The valid data value for COUPLED_AXES **SHOULD** be a space-delimited set of axes reported as the value of the name attribute for each axis. If name is not available, the piece of equipment **MUST** report the value of the nativeName attribute for each axis.
DIRECTION	The direction of motion. A subType **MUST** always be specified.
ROTARY	The rotational direction of a rotary motion using the right hand rule convention. The valid data value **MUST** be CLOCKWISE or COUNTER_CLOCKWISE.
LINEAR	The direction of motion of a linear motion. The valid data value **MUST** be POSTIVE or NEGATIVE.
DOOR_STATE	The opened or closed state of the door. The valid data value **MUST** be OPEN, UNLATCHED, or CLOSED.
END_OF_BAR	An indication of whether the end of a piece of bar stock being feed by a bar feeder has been reached. The valid data value **MUST** be expressed as a Boolean expression of YES or NO.

DataItem type/subType	Description
PRIMARY	Specific applications **MAY** reference one or more locations on a piece of bar stock as the indication for the END_OF_BAR. The main or most important location **MUST** be designated as the PRIMARY indication for the END_OF_BAR. If no subType is specified, PRIMARY **MUST** be the default END_OF_BAR indication.
AUXILIARY	When multiple locations on a piece of bar stock are referenced as the indication for the END_OF_BAR, the additional location(s) **MUST** be designated as AUXILIARY indication(s) for the END_OF_BAR.
EMERGENCY_STOP	The current state of the emergency stop signal. The valid data value **MUST** be ARMED (the circuit is complete and the device is allowed to operate) or TRIGGERED (the circuit is open and the device must cease operation).
EQUIPMENT_MODE	An indication that a piece of equipment, or a sub-part of a piece of equipment, is performing specific types of activities. EQUIPMENT_MODE **MAY** have more than one subtype defined. A subType **MUST** always be specified.
LOADED	An indication that the sub-parts of a piece of equipment are under load. Example: For traditional machine tools, this is an indication that the cutting tool is assumed to be engaged with the part. The valid data value **MUST** be ON or OFF.
WORKING	An indication that a piece of equipment is performing any activity – the equipment is active and performing a function under load or not. Example: For traditional machine tools, this includes when the piece of equipment is LOADED, making rapid moves, executing a tool change, etc. The valid data value **MUST** be ON or OFF.
OPERATING	An indication that the major sub-parts of a piece of equipment are powered or performing any activity whether producing a part or product or not. Example: For traditional machine tools, this includes when the piece of equipment is WORKING or it is idle. The valid data value **MUST** be ON or OFF.

DataItem type/subType	Description
POWERED	An indication that primary power is applied to the piece of equipment and, as a minimum, the controller or logic portion of the piece of equipment is powered and functioning or components that are required to remain on are powered. Example: Heaters for an extrusion machine that required to be powered even when the equipment is turned off. The valid data value **MUST** be ON or OFF.
DELAY	An indication that a piece of equipment is waiting for an event or an action to occur.
EXECUTION	The execution status of the Controller. The valid data value **MUST** be READY, ACTIVE, INTERRUPTED, FEED_HOLD, STOPPED, OPTIONAL_STOP, PROGRAM_STOPPED, or PROGRAM_COMPLETED.
FUNCTIONAL_MODE	The current intended production status of the device or component. Typically, the FUNCTIONAL_MODE **SHOULD** be modeled as a data item for the Device element, but **MAY** be modeled for any *Structural Element* in the XML document. The valid data value **MUST** be PRODUCTION, SETUP, TEARDOWN, MAINTENANCE, or PROCESS_DEVELOPMENT.
HARDNESS	The measurement of the hardness of a material. The measurement does not provide a unit. A subType **MUST** always be specified to designate the hardness scale associated with the measurement.
ROCKWELL	A scale to measure the resistance to deformation of a surface.
VICKERS	A scale to measure the resistance to deformation of a surface.
SHORE	A scale to measure the resistance to deformation of a surface.
BRINELL	A scale to measure the resistance to deformation of a surface.
LEEB	A scale to measure the elasticity of a surface.
MOHS	A scale to measure the resistance to scratching of a surface.
INTERFACE_STATE	The current functional or operational state of an Interface type element indicating whether the interface is active or is not currently functioning. The valid data value **MUST** be ENABLED or DISABLED.

DataItem type/subType	Description
~~LINE~~	~~The current line of code being executed.~~ ~~The data will be an alpha numeric value representing the line number of the current line of code being executed.~~ **DEPRECATED** in *Version 1.4*
~~MAXIMUM~~	~~The maximum line number of the code being executed.~~
~~MINIMUM~~	~~The minimum line number of the code being executed.~~
LINE_LABEL	An optional identifier for a BLOCK of code in a PROGRAM.
LINE_NUMBER	A reference to the position of a block of program code within a control program. The line number **MAY** represent either an absolute position starting with the first line of the program or an incremental position relative to the occurrence of the last LINE_LABEL. LINE_NUMBER does not change subject to any looping or branching in a control program. A subType **MUST** be defined.
ABSOLUTE	The position of a block of program code relative to the beginning of the control program.
INCREMENTAL	The position of a block of program code relative to the occurrence of the last LINE_LABEL encountered in the control program.
MATERIAL	The identifier of a material used or consumed in the manufacturing process. The valid data value **MUST** be a text string.
MESSAGE	Any text string of information to be transferred from a piece of equipment to a client software application.
OPERATOR_ID	The identifier of the person currently responsible for operating the piece of equipment. **DEPRECATION WARNING:** May be deprecated in the future. See USER below.
PALLET_ID	The identifier for a pallet. The valid data value **MUST** be a text string.
PART_COUNT	The current count of parts produced as represented by the Controller. The valid data value **MUST** be an integer value.
ALL	The count of all the parts produced. If the subtype is not given, this is the default.

DataItem type/subType	Description
GOOD	Indicates the count of correct parts made.
BAD	Indicates the count of incorrect parts produced.
TARGET	Indicates the number of parts that are projected or planned to be produced.
REMAINING	The number of parts remaining in stock or to be produced.
PART_ID	An identifier of a part in a manufacturing operation. The valid data value **MUST** be a text string.
PART_NUMBER	An identifier of a part or product moving through the manufacturing process. The valid data value **MUST** be a text string.
PATH_FEEDRATE_OVERRIDE	The value of a signal or calculation issued to adjust the feedrate for the axes associated with a Path component that may represent a single axis or the coordinated movement of multiple axes. The value provided for PATH_FEEDRATE_OVERRIDE is expressed as a percentage of the designated feedrate for the path. When PATH_FEEDRATE_OVERRIDE is applied, the resulting commanded feedrate for the path is limited to the value of the original feedrate multiplied by the value of the PATH_FEEDRATE_OVERRIDE. There **MAY** be different subtypes of PATH_FEEDRATE_OVERRIDE; each representing an override value for a designated subtype of feedrate depending on the state of operation of the path. The states of operation of a path are currently defined as PROGRAMMED, JOG, and RAPID.
JOG	The value of a signal or calculation issued to adjust the feedrate of the axes associated with a Path component when the axes, or a single axis, are being operated in a manual mode or method (jogging). When the JOG subtype of PATH_FEEDRATE_OVERRIDE is applied, the resulting commanded feedrate for the axes, or a single axis, associated with the path are limited to the value of the original JOG subtype of the PATH_FEEDRATE multiplied by the value of the JOG subtype of PATH_FEEDRATE_OVERRIDE.

DataItem type/subType	Description
PROGRAMMED	The value of a signal or calculation issued to adjust the feedrate of the axes associated with a Path component when the axes, or a single axis, are operating as specified by a logic or motion program or set by a switch. When the PROGRAMMED subtype of PATH_FEEDRATE_OVERRIDE is applied, the resulting commanded feedrate for the axes, or a single axis, associated with the path are limited to the value of the original PROGRAMMED subtype of the PATH_FEEDRATE multiplied by the value of the PROGRAMMED subtype of PATH_FEEDRATE_OVERRIDE.
RAPID	The value of a signal or calculation issued to adjust the feedrate of the axes associated with a Path component when the axes, or a single axis, are being operated in a rapid positioning mode or method (rapid). When the RAPID subtype of PATH_FEEDRATE_OVERRIDE is applied, the resulting commanded feedrate for the axes, or a single axis, associated with the path are limited to the value of the original RAPID subtype of the PATH_FEEDRATE multiplied by the value of the RAPID subtype of PATH_FEEDRATE_OVERRIDE.
PATH_MODE	Describes the operational relationship between a PATH *Structural Element* and another PATH *Structural Element* for pieces of equipment comprised of multiple logical groupings of controlled axes or other logical operations. The valid data value **MUST** be INDEPENDENT, MASTER, SYNCHRONOUS, or MIRROR. The default value **MUST** be INDEPENDENT if PATH_MODE is not specified.
POWER_STATE	The indication of the status of the source of energy for a *Structural Element* to allow it to perform its intended function or the state of an enabling signal providing permission for the *Structural Element* to perform its functions. The valid data value **MUST** be ON or OFF. **DEPRECATION WARNING:** May be deprecated in the future.
LINE	The state of the power source for the *Structural Element*.
CONTROL	The state of the enabling signal or control logic that enables or disables the function or operation of the *Structural Element*.
~~POWER_STATUS~~	**DEPRECATED** in *Version 1.1*.
PROGRAM	The name of the logic or motion program being executed by the Controller component. The valid data value **MUST** be a text string.

DataItem type/subType	Description
PROGRAM_EDIT	An indication of the Controller component's program editing mode. On many controls, a program can be edited while another program is currently being executed. The valid data value **MUST** be: ACTIVE: The controller is in the program edit mode. READY: The controller is capable of entering the program edit mode and no function is inhibiting a change of mode. NOT_READY: A function is inhibiting the controller from entering the program edit mode.
PROGRAM_EDIT_NAME	The name of the program being edited. This is used in conjunction with PROGRAM_EDIT when in ACTIVE state. The valid data value **MUST** be a text string.
PROGRAM_COMMENT	A comment or non-executable statement in the control program. The valid data value **MUST** be a text string.
PROGRAM_HEADER	The non-executable header section of the control program. The valid data value **MUST** be a text string.
ROTARY_MODE	The mode for a Rotary type axis. The valid data value **MUST** be SPINDLE, INDEX, or CONTOUR.
ROTARY_VELOCITY_OVERRIDE	A command issued to adjust the programmed velocity for a Rotary type axis. This command represents a percentage change to the velocity calculated by a logic or motion program or set by a switch for a Rotary type axis. ROTARY_VELOCITY_OVERRIDE is expressed as a percentage of the programmed ROTARY_VELOCITY.
SERIAL_NUMBER	The serial number associated with a Component, Asset, or Device. The valid data value **MUST** be a text string.
SPINDLE INTERLOCK	An indication of the status of the spindle for a piece of equipment when power has been removed and it is free to rotate. The valid data value **MUST** be: • ACTIVE if power has been removed and the spindle cannot be operated. • INACTIVE if power to the spindle has not been deactivated.
~~TOOL_ID~~	**DEPRECATED** in *Version 1.2*. See TOOL_ASSET_ID. ~~The identifier of the tool currently in use for a given Path~~

DataItem type/subType	Description
TOOL_ASSET_ID	The identifier of an individual tool asset. The valid data value **MUST** be a text string.
TOOL_NUMBER	The identifier of a tool provided by the piece of equipment controller. The valid data value **MUST** be a text string.
TOOL_OFFSET	A reference to the tool offset variables applied to the active cutting tool associated with a `Path` in a `Controller` type component. The valid data value **MUST** be a text string. The reported value returned for TOOL_OFFSET identifies the location in a table or list where the actual tool offset values are stored. A `subType` **MUST** always be specified.
RADIAL	A reference to a radial type tool offset variable.
LENGTH	A reference to a length type tool offset variable.
USER	The identifier of the person currently responsible for operating the piece of equipment. A `subType` **MUST** always be specified.
OPERATOR	The identifier of the person currently responsible for operating the piece of equipment.
MAINTENANCE	The identifier of the person currently responsible for performing maintenance on the piece of equipment.
SET_UP	The identifier of the person currently responsible for preparing a piece of equipment for production or restoring the piece of equipment to a neutral state after production.
WIRE	The identifier for the type of wire used as the cutting mechanism in Electrical Discharge Machining or similar processes. The valid data value **MUST** be a text string.
WORKHOLDING_ID	The identifier for the workholding currently in use. The valid data value **MUST** be a text string.
WORK_OFFSET	A reference to the offset variables for a work piece or part associated with a `Path` in a `Controller` type component. The valid data value **MUST** be a text string. The reported value returned for `WORK_OFFSET` identifies the location in a table or list where the actual tool offset values are stored.

1096

8.3 Data Items in category CONDITION

CONDITION category data items report data representing a *Structural Element's* status regarding its ability to operate or it provides an indication whether the data reported for the *Structural Element* is within an expected range.

CONDITION is reported differently than SAMPLE or EVENT. CONDITION **MUST** be reported as NORMAL, WARNING, or FAULT.

All DataItem types in the SAMPLE category **MAY** have associated CONDITION states. CONDITION states indicate whether the value for the data is within an expected range and **MUST** be reported as NORMAL, or the value is unexpected or out of tolerance for the data and a WARNING or FAULT **MUST** be provided.

Some DataItem types in the EVENT category **MAY** have associated CONDITION states.

Additional CONDITION types are provided to represent the health and fault status of *Structural Elements*. The table below defines these additional DataItem types.

CONDITION type data items are unlike other data item types since they **MAY** have multiple concurrently active values at any point in time.

DataItem Type	Description
ACTUATOR	An indication of a fault associated with an actuator.
CHUCK_INTERLOCK	An indication of the operational condition of the interlock function for an electronically controller chuck.
COMMUNICATIONS	An indication that the piece of equipment has experienced a communications failure.
DATA_RANGE	An indication that the value of the data associated with a measured value or a calculation is outside of an expected range.
DIRECTION	An indication of a fault associated with the direction of motion of a *Structural Element.*
END_OF_BAR	An indication that the end of a piece of bar stock has been reached.
HARDWARE	An indication of a fault associated with the hardware subsystem of the *Structural Element*.
INTERFACE_STATE	An indication of the operation condition of an Interface component.
LOGIC_PROGRAM	An indication that an error occurred in the logic program or programmable logic controller (PLC) associated with a piece of equipment.

DataItem Type	Description
MOTION_PROGRAM	An indication that an error occurred in the motion program associated with a piece of equipment
SYSTEM	A general purpose indication of a fault associated with a piece of equipment that is classified elsewhere.

9 *Sensor*

Sensor is a unique type of a piece of equipment. A *Sensor* is typically comprised of two major components: a *sensor unit* that provides signal processing, conversion, and communications and the *sensing elements* that provides a signal or measured value.

In MTConnect, the *sensor unit* is modeled as a *Lower Level* `Component` called `Sensor`. The *sensing element* may be modeled as a `Composition` element of a `Sensor` element and the measured value would be modeled as a `DataItem` (See *Section 8* of this document for more information on `DataItem` elements). Each *sensor unit* may have multiple *sensing elements*; each representing the data for a variety of measured values.

Example: A pressure transducer could be modeled as a `Sensor` (`Component`) with a `name` = *Pressure Transducer B* and its measured value could be modeled as a `PRESSURE` type `DataItem`.

While a *Sensor* may be modeled in the XML document in different ways, it will always be modeled to associate the information measured by each *sensor element* with the *Structural Element* to which the measured value is most closely associated.

9.1 Sensor Data

The most basic implementation of a sensor occurs when the *sensing element* itself is not identified in the data model, but the data that is measured by the *sensing element* is provided as a data item associated with a `Component`. An example would be the measured value of the temperature of a spindle motor. This would be represented as a `DataItem` called `TEMPERATURE` that is associated with the `Rotary` type axis element called “C” as follows:

```
1.  <Components>
2.    <Axes
3.      <Components>
4.        <Rotary id="c" name="C">
5.          <DataItems>
6.            <DataItem type="TEMPERATURE" id="ctemp" category="SAMPLE"
7.              name="Stemp" units="DEGREE"/>
8.          </DataItems>
9.        </Rotary>
10.     </Components>
11.   </Axes>
12. </Components>
```

A sensor may measure values associated with any `Component` or `Device` element. Some examples of how sensor data may be modeled are represented in *Figure 12* below:

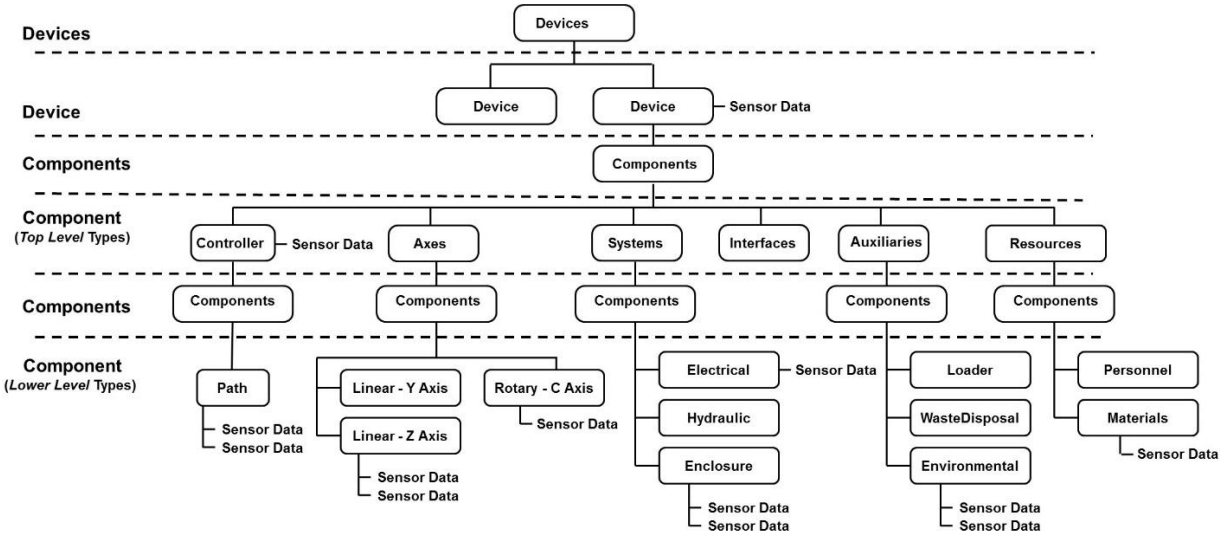


Figure 20: Sensor Data Associations

9.2 Sensor Unit

A *sensor unit* is an intelligent piece of equipment that manages the functions of one or more *sensing elements*.

Typical functions of the *sensor unit* include:

- convert low level signals from the *sensing elements* into data that can be used by other pieces of equipment. (Example: Convert a non-linear millivolt signal from a temperature sensor into a scaled temperature value that can be transmitted to another piece of equipment.)
- process *sensing element* data into calculated values. (Example: temperature sensor data is converted into calculated values of average temperature, maximum temperature, minimum temperature, etc.)
- provide calibration and configuration information associated with each *sensing element*
- monitor the health and integrity of the *sensing elements* and the *sensor unit*. (Example: The *sensor unit* may provide diagnostics on each *sensing element* (e.g., open wire detection) and itself (e.g., measure internal temperature of the *sensor unit*).

Depending on how the *sensor unit* is used, it may be considered as either an independent piece of equipment and modeled in the XML document as a `Device`, or it may be modeled as a *Lower Level* `Component` called `Sensor` if it is integral to a piece of equipment.

A `Sensor` **MAY** have its own `uuid` so it can be tracked throughout its lifetime.

The following examples demonstrate how a *Sensor* may be modeled in the XML document differently based on how the *Sensor* functions within the overall piece of equipment.

Example#1: If the *Sensor* provides vibration measurement data for the spindle on a piece of equipment, it could be modeled as a `Sensor` for rotary axis named C.

```
<Components>
  <Axes>
    <Components>
      <Rotary id="c" name="C">
        <Components>
          <Sensor id="spdlm" name="Spindlemonitor">
            <DataItems>
              <DataItem type="DISPLACEMENT" id="cvib"
                category="SAMPLE" name="Svib" units="MILLIMETER"/>
            </DataItems>
          </Sensor >
        <Components>
      </Rotary>
    </Components>
  </Axes>
</Components>
```

Example#2: If a *Sensor* provides measurement data for multiple `Component` elements within a piece of equipment and is not associated with any particular `Component` element, it **MAY** be modeled in the XML document as an independent *Lower Level* `Component` and the data associated with measurements are associated with their associated `Component` elements.

This example represents a *sensor unit* with two *sensing elements*, one measures spindle vibration and the other measures the temperature for the X axis. The *sensor unit* also has a *sensing element* measuring the internal temperature of the *sensor unit*.

```
<Device id="d1" uuid="HM1" name="HMC_3Axis">
  <Description>3 Axis Mill</Description>
  <Components>
    <Axes>
      <Components>
        <Sensor id="sens1" name="Sensorunit">
          <DataItems>
            <DataItem type="TEMPERATURE" id="sentemp"
                category="SAMPLE" name="Sensortemp" units="DEGREE"/>
          </DataItems>
        </Sensor>
        <Rotary id="c" name="C">
          <DataItems>
            <DataItem type="DISPLACEMENT" id="cvib" category="SAMPLE"
                name="Svib" units="MILLIMETER">
              <Source componentid="sens1"/>
            <DataItem/>
          </DataItems>
        </Rotary>
        <Linear id="x" name="X">
          <DataItems>
            <DataItem type="TEMPERATURE" id="xt" category="SAMPLE"
                name="Xtemp" units="DEGREE">
              <Source componentid="sens1"/>
            <DataItem/>
          </DataItems>
        </Linear>
      </Components>
    </Axes>
  </Components>
</Device>
```

9.3 `Sensor` Configuration

When a *Sensor unit* is modeled in the XML document as a `Component` or as a separate piece of equipment, it may provide additional configuration information for the *sensor elements* and the *sensor unit* itself.

Configuration data provides information required for maintenance and support of the sensor.

Configuration data is *only* available when the *Sensor unit* is modeled as a `Component` or a separate piece of equipment. For details on the modeling of configuration data in the XML document, see *Section 4.4.3.2 `Configuration` for `Component`*. Details specific to `SensorConfiguration` are provided below.

When `Sensor` represents the *sensor unit* for multiple *sensing element(s)*, each *sensing element* is represented by a `Channel`. The sensor unit itself and each `Channel` representing one *sensing element* **MAY** have its own configuration data.

`SensorConfiguration` can contain any descriptive content for a *sensor unit*. This element is defined to contain mixed content and additional XML elements (indicated by the `any` element in the schema below) **MAY** be added to extend the schema for `SensorConfiguration`.

The following XML schema represents the structure of the `SensorConfiguration` XML element showing the attributes defined for `SensorConfiguration`.

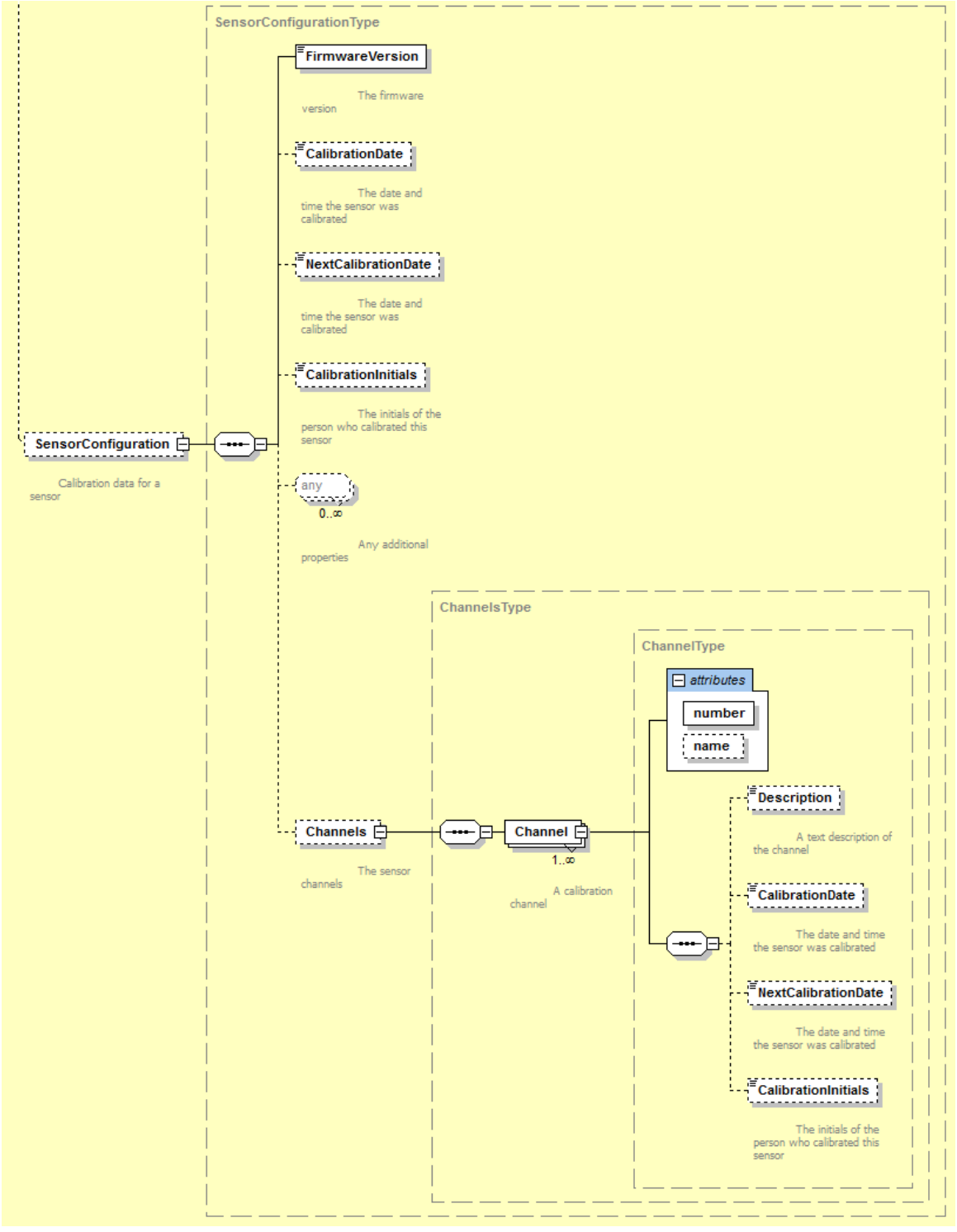


Figure 21: `SensorConfiguration` Schema Diagram

Element	Description	Occurrence
SensorConfiguration	An element that can contain descriptive content defining the configuration information for Sensor. For Sensor, the valid configuration is SensorConfiguration which provides data from a subset of items commonly found in a transducer electronic data sheet for sensors and actuators called TEDS. TEDS formats are defined in IEEE 1451.0 and 1451.4 transducer interface standards (ref 15 and 16, respectively). MTConnect does not support all of the data represented in the TEDS data, nor does it duplicate the function of the TEDS data sheets.	0..1

1261

1262 ### 9.3.1 Elements for SensorConfiguration

1263 The following table defines the configuration elements available for
1264 SensorConfiguration:

1265

Element	Description	Occurrence
FirmwareVersion	Version number for the *sensor unit* as specified by the manufacturer. FirmwareVersion is a required element if SensorConfiguration is used. The data value for FirmwareVersion is provided in the CDATA for this element and **MAY** be any numeric or text content.	1
CalibrationDate	Date upon which the *sensor unit* was last calibrated. The data value for CalibrationDate is provided in the CDATA for this element and **MUST** be represented in the W3C ISO 8601 format.	0..1
NextCalibrationDate	Date upon which the *sensor unit* is next scheduled to be calibrated. The data value for NextCalibrationDate is provided in the CDATA for this element and **MUST** be represented in the W3C ISO 8601 format.	0..1

Element	Description	Occurrence
CalibrationInitials	The initials of the person verifying the validity of the calibration data. The data value for CalibrationInitials is provided in the CDATA for this element and **MAY** be any numeric or text content.	0..1
Channels	When Sensor represents multiple *sensing elements*, each *sensing element* is represented by a Channel for the Sensor. Channels is an XML container used to organize information for the *sensing elements*.	0..1

1266

1267 #### 9.3.1.1 Attributes for Channel

1268 Channel represents each *sensing element* connected to a *sensor unit*. The table below defines
1269 the attributes for Channel:

Attribute	Description	Occurrence
number	A unique identifier that will only refer to a specific *sensing element*. number is a required attribute. For example, this can be the manufacturer code and the serial number. number **SHOULD** be alphanumeric and not exceeding 255 characters. An NMTOKEN XML type.	1
name	The name of the *sensing element*. name is an optional attribute. name **SHOULD** be unique within the *sensor unit* to allow for easier data integration. An NMTOKEN XML type.	0..1

1270

1271

9.3.1.2 Elements for `Channel`

The following table describes the elements provided for `Channel`.

Element	Description	Occurrence
`Description`	An XML element that can contain any descriptive content. The `CDATA` of `Description` **MAY** include any additional descriptive information the implementer chooses to include regarding a *sensor element*.	0..1
`CalibrationDate`	Date upon which the *sensor unit* was last calibrated to the *sensor element* The data value for `CalibrationDate` is provided in the CDATA for this element and **MUST** be represented in the W3C ISO 8601 format.	0..1
`NextCalibrationDate`	Date upon which the *sensor element* is next scheduled to be calibrated with the *sensor unit*. The data value for `NextCalibrationDate` is provided in the CDATA for this element and **MUST** be represented in the W3C ISO 8601 format.	0..1
`CalibrationInitials`	The initials of the person verifying the validity of the calibration data The data value for `CalibrationInitials` is provided in the CDATA for this element and **MAY** be any numeric or text content.	0..1

The following is an example of the configuration data for Sensor that is modeled as a Component. It has Configuration data for the *sensor unit*, one Channel named A/D:1, and two DataItems – Voltage (as a SAMPLE) and Voltage (as a CONDITION or alarm).

```
<Sensor id="sensor" name="sensor">
  <Configuration>
    <SensorConfiguration>
      <FirmwareVersion>2.02</FirmwareVersion>
      <CalibrationDate>2010-05-16</CalibrationDate>
      <NextCalibrationDate>2010-05-16</NextCalibrationDate>
      <CalibrationInitials>WS</CalibrationInitials>
      <Channels>
        <Channel number="1" name="A/D:1">
          <Description>A/D With Thermister</Description>
        </Channel>
      </Channels>
    </SensorConfiguration>
  </Configuration>
  <DataItems>
    <DataItem category="CONDITION" id="senvc" type="VOLTAGE" />
    <DataItem category="SAMPLE" id="senv" type="VOLTAGE" units="VOLT"
      subType="DIRECT" />
  </DataItems>
</Sensor>
```

Appendices

A. Bibliography

1. Engineering Industries Association. *EIA Standard - EIA-274-D*, Interchangeable Variable, Block Data Format for Positioning, Contouring, and Contouring/Positioning Numerically Controlled Machines. Washington, D.C. 1979.

2. ISO TC 184/SC4/WG3 N1089. *ISO/DIS 10303-238*: Industrial automation systems and integration Product data representation and exchange Part 238: Application Protocols: Application interpreted model for computerized numerical controllers. Geneva, Switzerland, 2004.

3. International Organization for Standardization. *ISO 14649*: Industrial automation systems and integration – Physical device control – Data model for computerized numerical controllers – Part 10: General process data. Geneva, Switzerland, 2004.

4. International Organization for Standardization. *ISO 14649*: Industrial automation systems and integration – Physical device control – Data model for computerized numerical controllers – Part 11: Process data for milling. Geneva, Switzerland, 2000.

5. International Organization for Standardization. *ISO 6983/1* – Numerical Control of machines – Program format and definition of address words – Part 1: Data format for positioning, line and contouring control systems. Geneva, Switzerland, 1982.

6. Electronic Industries Association. *ANSI/EIA-494-B-1992*, 32 Bit Binary CL (BCL) and 7 Bit ASCII CL (ACL) Exchange Input Format for Numerically Controlled Machines. Washington, D.C. 1992.

7. National Aerospace Standard. *Uniform Cutting Tests* - NAS Series: Metal Cutting Equipment Specifications. Washington, D.C. 1969.

8. International Organization for Standardization. *ISO 10303-11*: 1994, Industrial automation systems and integration Product data representation and exchange Part 11: Description methods: The EXPRESS language reference manual. Geneva, Switzerland, 1994.

9. International Organization for Standardization. *ISO 10303-21*: 1996, Industrial automation systems and integration -- Product data representation and exchange -- Part 21: Implementation methods: Clear text encoding of the exchange structure. Geneva, Switzerland, 1996.

10. H.L. Horton, F.D. Jones, and E. Oberg. *Machinery's handbook*. Industrial Press, Inc. New York, 1984.

11. International Organization for Standardization. *ISO 841-2001: Industrial automation systems and integration - Numerical control of machines - Coordinate systems and motion nomenclature.* Geneva, Switzerland, 2001.

12. ASME B5.57: *Methods for Performance Evaluation of Computer Numerically Controlled Lathes and Turning Centers,* 1998

13. ASME/ANSI B5.54: *Methods for Performance Evaluation of Computer Numerically Controlled Machining Centers. 2005.*

14. OPC Foundation. *OPC Unified Architecture Specification, Part 1: Concepts Version 1.00. July 28, 2006.*

15. IEEE STD 1451.0-2007*, Standard for a Smart Transducer Interface for Sensors and Actuators – Common Functions, Communication Protocols, and Transducer Electronic Data Sheet (TEDS) Formats, IEEE Instrumentation and Measurement Society, TC-9, The Institute of Electrical and Electronics Engineers, Inc., New York, N.Y. 10016, SH99684, October 5, 2007.*

16. IEEE STD 1451.4-1994, Standard for a Smart Transducer Interface for Sensors and Actuators – Mixed-Mode Communication Protocols and Transducer Electronic Data Sheet (TEDS) Formats, IEEE Instrumentation and Measurement Society, TC-9, The Institute of Electrical and Electronics Engineers, Inc., New York, N.Y. 10016, SH95225, December 15, 2004.