MTConnect® Standard
Part 2.0 – Devices Information Model
Version 1.5.0

Prepared for: MTConnect Institute
Prepared on: December 2, 2019

MTConnect Specification and Materials

Table of Contents

Table of Figures

List of Tables

1 Purpose of This Document

This document, *MTConnect Standard: 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 *Agent* from the equipment. The *Devices Information Model* also defines the structure for the XML document that is returned from an *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 *MTConnect Standard: Part 3.0 - Streams Information Model* of the MTConnect Standard for details on the XML documents that are returned from an *Agent* in response to a *Sample Request* or *Current Request*.

2 Terminology and Conventions

Refer to *Section 3* of *MTConnect Standard Part 1.0 - Overview and Fundamentals* for a dictionary of terms, reserved language, and document conventions used in the MTConnect Standard.

2.1 Glossary

CDATA

General meaning:

An abbreviation for Character Data.

CDATA is used to describe a value (text or data) published as part of an XML element.

For example, "`This is some text`" is the CDATA in the XML element:

```
<Message ...>This is some text</Message>
```

Appears in the documents in the following form: CDATA

HTTP

Hyper-Text Transport Protocol. The protocol used by all web browsers and web applications.

Note: HTTP is an IETF standard and is defined in RFC 7230.
See https://tools.ietf.org/html/rfc7230 for more information.

NMTOKEN

The data type for XML identifiers.

Note: The identifier must start with a letter, an underscore "_" or a colon. The next character must be a letter, a number, or one of the following ".", "-", "_", ":". The identifier must not have any spaces or special characters.

Appears in the documents in the following form: NMTOKEN.

XML

Stands for eXtensible Markup Language.

XML defines a set of rules for encoding documents that both a human-readable and machine-readable.

XML is the language used for all code examples in the MTConnect Standard.

Refer to http://www.w3.org/XML for more information about XML.

Agent

Refers to an MTConnect Agent.

Software that collects data published from one or more piece(s) of equipment, organizes that data in a structured manner, and responds to requests for data from client software systems by providing a structured response in the form of a *Response Document* that is constructed using the *semantic data models* defined in the Standard.

Appears in the documents in the following form: *Agent*.

Asset

General meaning:

Typically referred to as an *MTConnect Asset*.

An *MTConnect 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.

Used to identify a storage area in an *Agent*:

See description of *buffer*.

Used as an *Information Model*:

Used to describe an *Information Model* that contains the rules and terminology that describe information that may be included in electronic documents representing *MTConnect Assets*.

The *Asset Information Models* defines the structure for the *Assets Response Document*.

Individual *Information Models* describe the structure of the *Asset Documents* represent each type of *MTConnect Asset*. Appears in the documents in the following form: *Asset Information Models* or (asset type) *Information Model*.

Used when referring to an *MTConnect Asset*:

Refers to the information related to an *MTConnect Asset* or a group of *MTConnect Assets*.

Appears in the documents in the following form: *Asset* or *Assets*.

Used as an XML container or element:

- When used as an XML container that consists of one or more types of `Asset` XML elements.

  Appears in the documents in the following form: `Assets`.

- When used as an abstract XML element. It is replaced in the XML document by types of `Asset` elements representing individual *Asset* entities.

  Appears in the documents in the following form: `Asset`.

Used to describe information stored in an *Agent*:

Identifies an electronic document published by a data source and stored in the *assets buffer* of an *Agent*.

Appears in the documents in the following form: *Asset Document*.

Used as an XML representation of an *MTConnect Response Document*:

Identifies an electronic document encoded in XML and published by an *Agent* in response to a *Request* for information from a client software application relating to *MTConnect Assets*.

Appears in the documents in the following form: `MTConnectAssets`.

Used as an *MTConnect Request*:

Represents a specific type of communications request between a client software application and an *Agent* regarding *MTConnect Assets*.

Appears in the documents in the following form: *Asset Request*.

Used as part of an *HTTP Request*:

Used in the path portion of an *HTTP Request Line*, by a client software application, to initiate an *Asset Request* to an *Agent* to publish an `MTConnectAssets` document.

Appears in the documents in the following form: `asset`.

Asset Document

An electronic document published by an *Agent* in response to a *Request* for information from a client software application relating to Assets.

buffer

General meaning:

A section of an *Agent* that provides storage for information published from pieces of equipment.

Used relative to *Streaming Data*:

A section of an *Agent* that provides storage for information relating to individual pieces of *Streaming Data*.

Appears in the documents in the following form: *buffer*.

Used relative to *MTConnect Assets*:

A section of an *Agent* that provides storage for *Asset Documents*.

Appears in the documents in the following form: *assets buffer*.

Child Element

A portion of a data modeling structure that illustrates the relationship between an element and the higher-level *Parent Element* within which it is contained.

Appears in the documents in the following form: *Child Element*.

Current Request

An HTTP request to the *Agent* for returning latest known values for the `DataItem` as an `MTConnectStreams` XML document

Data Entity

A primary data modeling element that represents all elements that either describe data items that may be reported by an *Agent* or the data items that contain the actual data published by an *Agent*.

Appears in the documents in the following form: *Data Entity*.

Data Set

A set of *key-value pairs* where each entry is uniquely identified by the *key*.

Devices Information Model

A set of rules and terms that describes the physical and logical configuration for a piece of equipment and the data that may be reported by that equipment.

Appears in the documents in the following form: *Devices Information Model*.

Document

General meaning:

A piece of written, printed, or electronic matter that provides information.

Used to represent an *MTConnect Document*:

Refers to printed or electronic document(s) that represent a *Part*(s) of the MTConnect Standard.

Appears in the documents in the following form: *MTConnect Document*.

Used to represent a specific representation of an *MTConnect Document*:

Refers to electronic document(s) associated with an *Agent* that are encoded using XML; *Response Documents* or *Asset Documents*.

Appears in the documents in the following form: *MTConnect XML Document*.

Used to describe types of information stored in an *Agent*:

In an implementation, the electronic documents that are published from a data source and stored by an *Agent*.

Appears in the documents in the following form: *Asset Document*.

Used to describe information published by an *Agent*:

A document published by an *Agent* based upon one of the *semantic data models* defined in the MTConnect Standard in response to a request from a client.

Appears in the documents in the following form: *Response Document*.

Equipment Metadata

See *Metadata*

HTTP Request

In the MTConnect Standard, a communications command issued by a client software application to an *Agent* requesting information defined in the *HTTP Request Line*.

Appears in the documents in the following form: *HTTP Request*.

HTTP Request Line

In the MTConnect Standard, the first line of an *HTTP Request* describing a specific *Response Document* to be published by an *Agent*.

Appears in the documents in the following form: *HTTP Request Line*.

Information Model

The rules, relationships, and terminology that are used to define how information is structured.

For example, an information model is used to define the structure for each *MTConnect Response Document*; the definition of each piece of information within those documents and the relationship between pieces of information.

Appears in the documents in the following form: *Information Model*.

Interaction Model

The definition of information exchanged to support the interactions between pieces of equipment collaborating to complete a task.

Appears in the documents in the following form: *Interaction Model*.

Interface

General meaning:

The exchange of information between pieces of equipment and/or software systems.

Appears in the documents in the following form: interface.

Used as an *Interaction Model*:

An *Interaction Model* that describes a method for inter-operations between pieces of equipment.

Appears in the documents in the following form: *Interface*.

Used as an XML container or element:

- When used as an XML container that consists of one or more types of `Interface` XML elements.

Appears in the documents in the following form: `Interfaces`.

- When used as an abstract XML element. It is replaced in the XML document by types of `Interface` elements.

Appears in the documents in the following form: `Interface`

key

A unique identifier in a *key-value pair* association.

key-value pair

An association between an identifier referred to as the *key* and a value which taken together create a *key-value pair*. When used in a set of *key-value pairs* each *key* is unique and will only have one value associated with it at any point in time.

Lower Level

A nested element that is below a higher level element.

Metadata

Data that provides information about other data.

For example, *Equipment Metadata* defines both the *Structural Elements* that represent the physical and logical parts and sub-parts of each piece of equipment, the relationships between those parts and sub-parts, and the definitions of the *Data Entities* associated with that piece of equipment.

Appears in the documents in the following form: *Metadata* or *Equipment Metadata*.

MTConnect Document

See *Document*.

MTConnect Request

A communication request for information issued from a client software application to an *Agent*.

Appears in the documents in the following form: *MTConnect Request*.

MTConnect XML Document

See *Document*.

Parent Element

An XML element used to organize *Lower Level* child elements that share a common relationship to the *Parent Element*.

Appears in the documents in the following form: *Parent Element*.

Request

A communications method where a client software application transmits a message to an *Agent*. That message instructs the *Agent* to respond with specific information.

Appears in the documents in the following form: *Request*.

Response Document

See *Document*.

Sample Request

A request from the *Agent* for a stream of time series data.

semantic data model

A methodology for defining the structure and meaning for data in a specific logical way.

It provides the rules for encoding electronic information such that it can be interpreted by a software system.

Appears in the documents in the following form: *semantic data model*.

Streaming Data

The values published by a piece of equipment for the *Data Entities* defined by the *Equipment Metadata*.

Appears in the documents in the following form: *Streaming Data*.

Streams Information Model

The rules and terminology (*semantic data model*) that describes the *Streaming Data* returned by an *Agent* from a piece of equipment in response to a *Sample Request* or a *Current Request*.

Appears in the documents in the following form: *Streams Information Model*.

Structural Element

General meaning:

An XML element that organizes information that represents the physical and logical parts and sub-parts of a piece of equipment.

Appears in the documents in the following form: *Structural Element*.

Used to indicate hierarchy of Components:

When used to describe a primary physical or logical construct within a piece of equipment.

Appears in the documents in the following form: *Top Level Structural Element*.

When used to indicate a *Child Element* which provides additional detail describing the physical or logical structure of a *Top Level Structural Element*.

Appears in the documents in the following form: *Lower Level Structural Element*.

Top Level

Structural Elements that represent the most significant physical or logical functions of a piece of equipment.

Valid Data Value

One or more acceptable values or constrained values that can be reported for a *Data Entity*.

Appears in the documents in the following form: *Valid Data Value*(s).

2.2 Acronyms

AMT

The Association for Manufacturing Technology

2.3 MTConnect References

[MTConnect Part 1.0] *MTConnect Standard Part 1.0 - Overview and Fundamentals*. Version 1.5.0.

[MTConnect Part 2.0] *MTConnect Standard: Part 2.0 - Devices Information Model*. Version 1.5.0.

[MTConnect Part 3.0] *MTConnect Standard: Part 3.0 - Streams Information Model*. Version 1.5.0.

[MTConnect Part 4.0] *MTConnect Standard: Part 4.0 - Assets Information Model*. Version 1.5.0.

[MTConnect Part 5.0] *MTConnect Standard: Part 5.0 - Interfaces*. Version 1.5.0.

3 Devices Information Model

The *Devices Information Model* provides a representation of 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 *MTConnect Standard Part 1.0 - Overview and Fundamentals Section 8.1.1*) to an *Agent* associated with a piece of equipment. An *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 *MTConnect Standard Part 1.0 - Overview and Fundamentals Section 6.5.1*.

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 - Structural Elements for MTConnectDevices* 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 - Data Entities for Device* and *Section 8 - Listing of Data Items*).

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 *MTConnect*

Standard: 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 *Section 4 - Structural Elements for MTConnectDevices*, *Section 7 - Data Entities for Device*, and *Section 9 - Sensor*.

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 *MTConnect Standard: Part 4.0 - Assets Information Model* 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* 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 an *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 - Component Structural Elements*. Each Component type is also a container type element. As a container, the Component type element is used to organize infor-

mation 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.

Example 1 illustrates the relationship between a parent `Component` and *Lower Level* child components:

Example 1: Component Levels

```
1 <Devices>
2   <Device>
3     <Components>
4       <Axes>   Parent Component
5         <Components>
6           <Rotary>   Child component of Axes and Parent component of Lower Level components
7             <Components>
8               <Chuck>   Child Component of Rotary
```

Figure 1 demonstrates the various *Structural Elements* provided to describe a piece of equipment and the relationship between these elements.

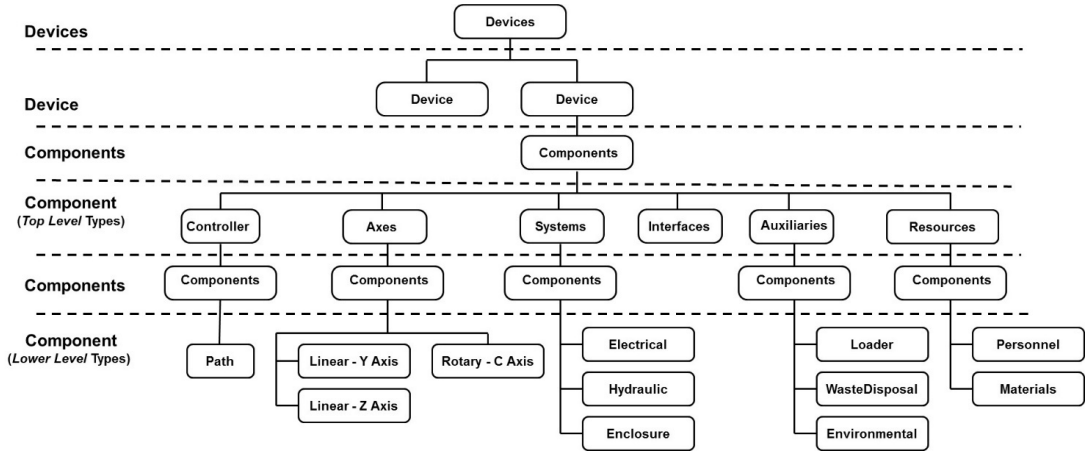

Figure 1: Example Device Structural Elements

`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 - Composition Type Structural Elements*.

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`.

Example 2 represents an XML document structure that demonstrates the relationship between a parent `Component` and its `Composition` elements.

Example 2: Component levels with Composition

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

Figure 2 demonstrates this relationship between a `Component` and some of its potential `Composition` elements.

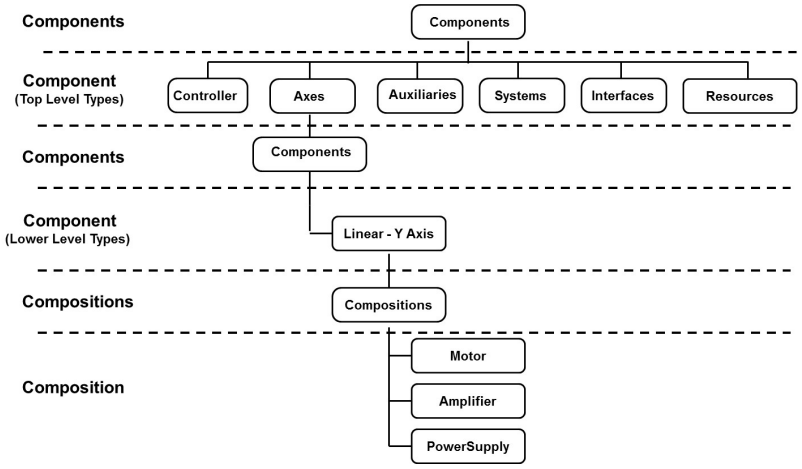

Figure 2: Example Composition Structural Elements

4.1 Devices

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

Table 1: MTConnect Devices Element

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

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 - Component*). 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 the 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.

Table 2: MTConnect Device Element

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..*

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 an *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

Figure 3 represents the structure of the `Device` XML element showing the attributes defined for `Device` and the elements that may be associated with `Device`.

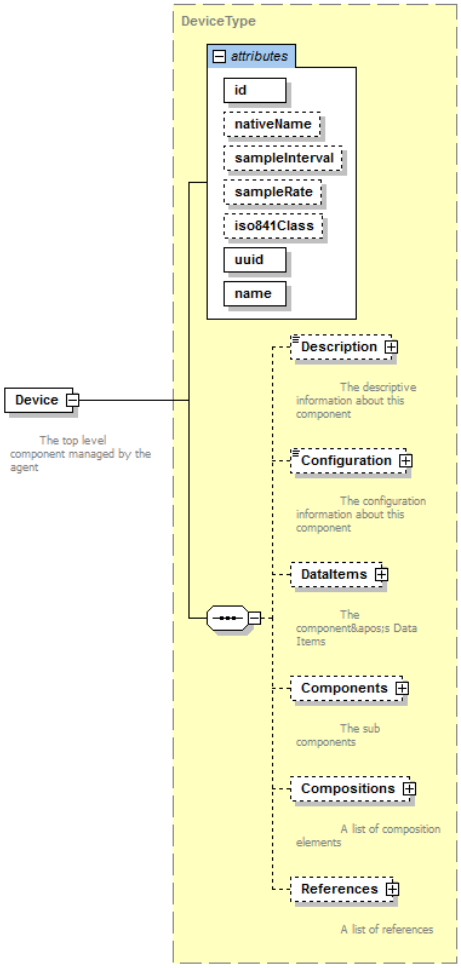

Figure 3: Device Diagram

4.2.2 Attribute for Device

Table 3 defines the attributes that may be used to provide additional information for a `Device` type element.

Table 3: Attributes for Device

Attribute	Description	Occurrence
`id`	The unique identifier for this 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 †††

Continuation of Table 3		
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

Table 4 lists the elements defined to provide additional information for a `Device` element. These elements are organized in the `Device` container.

Table 4: Elements for Device

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 - Data Entities for Device* and *Section 8 - Listing of Data Items* for more detail) 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

Figure 4 shows 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) **MAY** be added to extend the schema for `Description`.

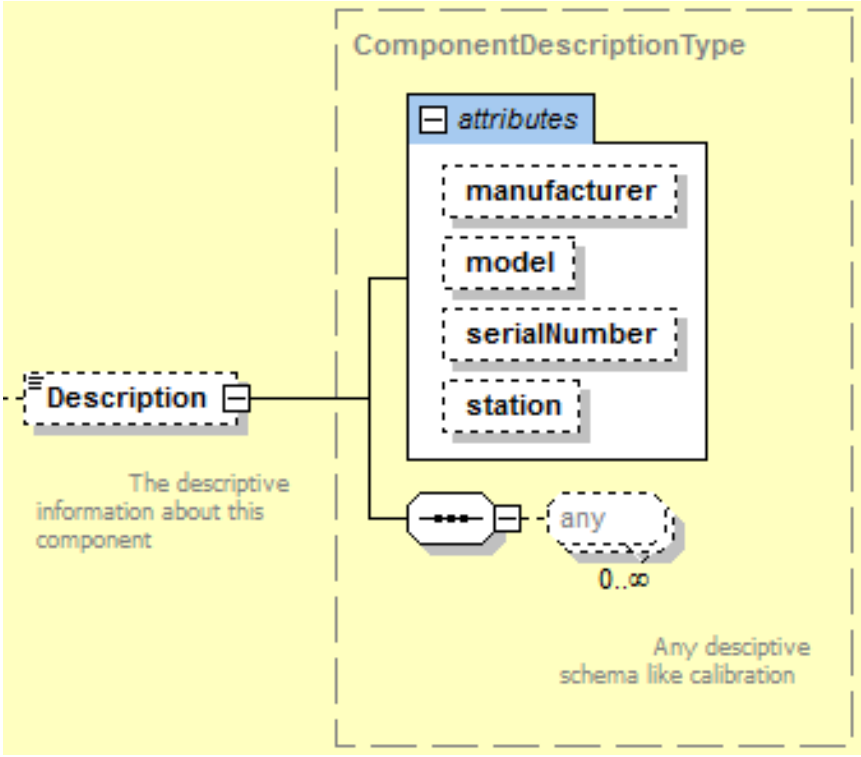

Figure 4: Description Diagram

453 *Table 5* lists the attributes defined for the `Description` XML element.

Table 5: Attributes for Description

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

Continuation of Table 5		
Attribute	Description	Occurrence
`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

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

Example 3: Example of Description

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

4.2.3.2 Configuration for Device

The `Configuration` XML element contains technical information about a piece of equipment. `Configuration` **MAY** include any information describing the physical layout or functional characteristics of the piece of equipment, such as capabilities, testing, installation, operation, calibration, or maintenance. `Configuration` **MAY** also include information representing the inter-relationships between pieces of equipment.

Table 6: MTConnect Configuration Element

Element	Description	Occurrence
`Configuration`	An XML element that contains technical information about a piece of equipment describing its physical layout, functional characteristics, and relationships with other pieces of equipment.	0..1

Configuration data for `Device` is structured in the `MTConnectDevices` XML document as shown in *Figure 5* . `AbstractConfiguration` is an abstract type XML element. It will never appear in the XML document representing a piece of equipment.

When `Configuration` is provided for a piece of equipment, that type of `Configuration` will appear in the XML document.

`SensorConfiguration` is described in detail in *Section 9.3 - Sensor Configuration*.

`Relationships` is described in detail in *Section 4.9 - Relationships*.

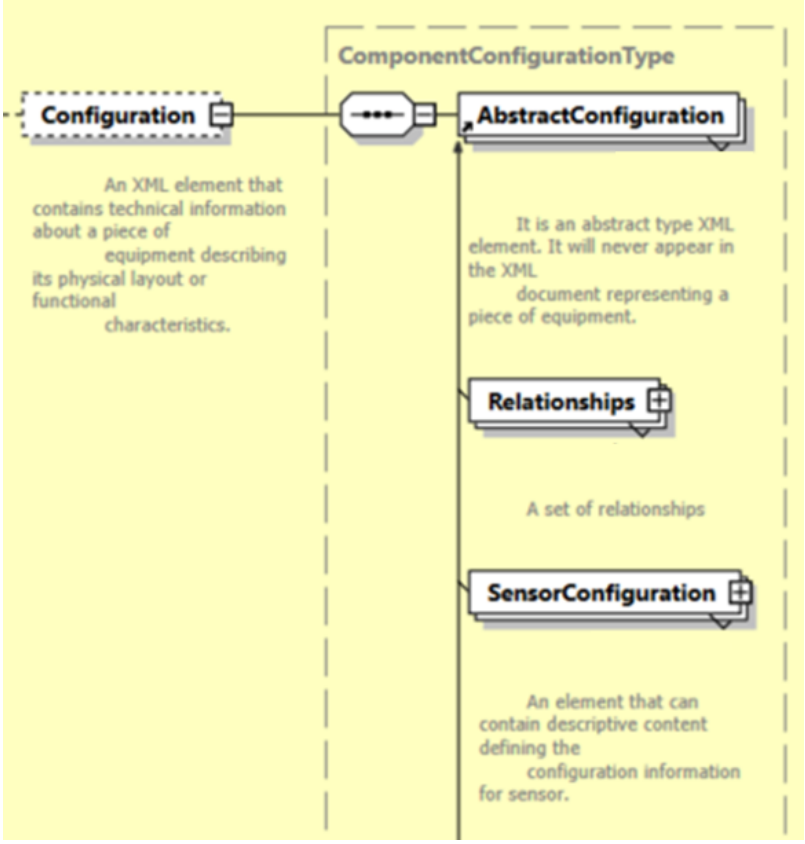

Figure 5: Configuration Diagram

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 *Section 7 - Data Entities for Device* and *Section 8 - Listing of Data Items* 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 - Compositions* for details on `Compositions`.

4.2.3.6 References for Device

`References` is an XML container used to organize `References` elements associated with a `Device` element. See *Section 4.7 - References* 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.

Table 7: MTConnect Components Element

Element	Description	Occurrence
`Components`	An 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 - Component Structural Elements* and include elements such as Axes, Controller, and Systems.

Table 8: MTConnect Component Element

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..*

4.4.1 XML Schema Structure for Component

Figure 6 represents the structure of a Component XML element showing the attributes defined for Component and the elements that **MAY** be associated with Component.

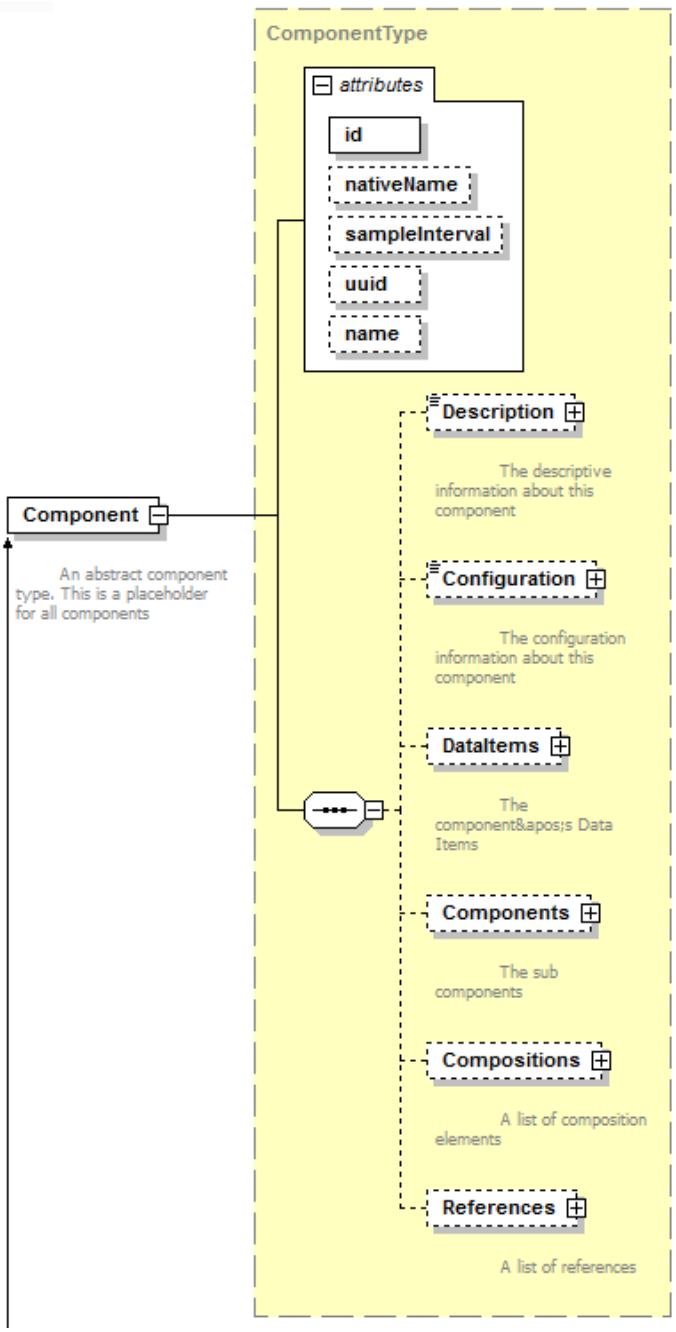

Figure 6: Component Diagram

4.4.2 Attribute for Component

Table 9 defines the attributes that may be used to provide additional information for a `Component` type XML element.

Table 9: Attributes for Component

Attribute	Description	Occurrence
`id`	The unique identifier for this 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

Continuation of Table 9		
Attribute	Description	Occurrence
`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 †††

Continuation of Table 9		
Attribute	Description	Occurrence
`uuid`	A unique identifier for this XML element. `uuid` is an optional attribute. The value provided for 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 `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` **MUST** 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

Table 10 lists the elements defined to provide additional information for a `Component` type XML element.

Table 10: Elements for Component

Element	Description	Occurrence
`Description`	An 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* (defined in *Section 8 - Listing of Data Items*) 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 - Composition Type Structural Elements*) associated with this `Component` element.	0..1
`References`	A container for the `Reference` elements associated with this `Component` element.	0..1 †

Note: †At least one of `Components`, `DataItems`, or `References` **MUST** be provided.

4.4.3.1 Description for Component

Figure 7 illustrates 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) **MAY** be added to extend the schema for `Description`.

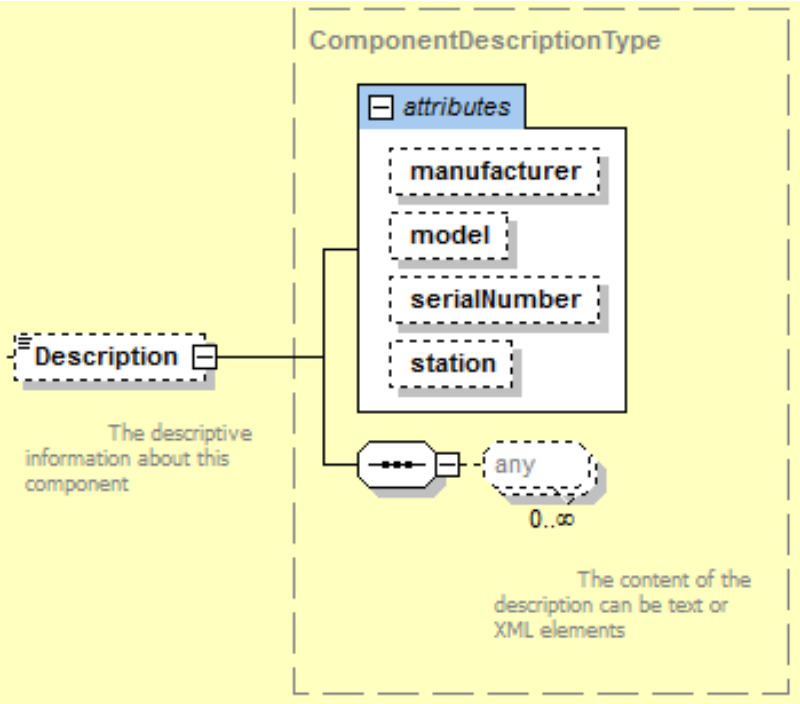

Figure 7: Description of Component Diagram

530 *Table 11* lists the attributes defined for the `Description` XML element.

Table 11: Attributes for Description for Component

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

Continuation of Table 11		
Attribute	Description	Occurrence
`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.

Example 4: Example of Description

```
1 <Description manufacturer="Example Co"
2     serialNumber="EXCO-TT-099PP-XXXX"> Advanced Pulse
3     watt-hour transducer with pulse output
4 </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. `Configuration` **MAY** also include information representing the inter-relationships between components within a piece of equipment.

Table 12: MTConnect Configuration Element for Component

Element	Description	Occurrence
`Configuration`	An XML element that contains technical information about a component describing its physical layout, functional characteristics, and relationships with other components within a piece of equipment.	0..1

Configuration data for `Component` is structured in the `MTConnectDevices` XML

document as shown in *Figure 8* . `AbstractConfiguration` is an abstract type XML element. It will never appear in the XML document representing a piece of equipment. When `Configuration` is provided for a component, that type of `Configuration` will appear in the XML document.

`SensorConfiguration` is described in detail in *Section 9.3 - Sensor Configuration*.

`Relationships` is described in detail in *Section 4.9 - Relationships*.

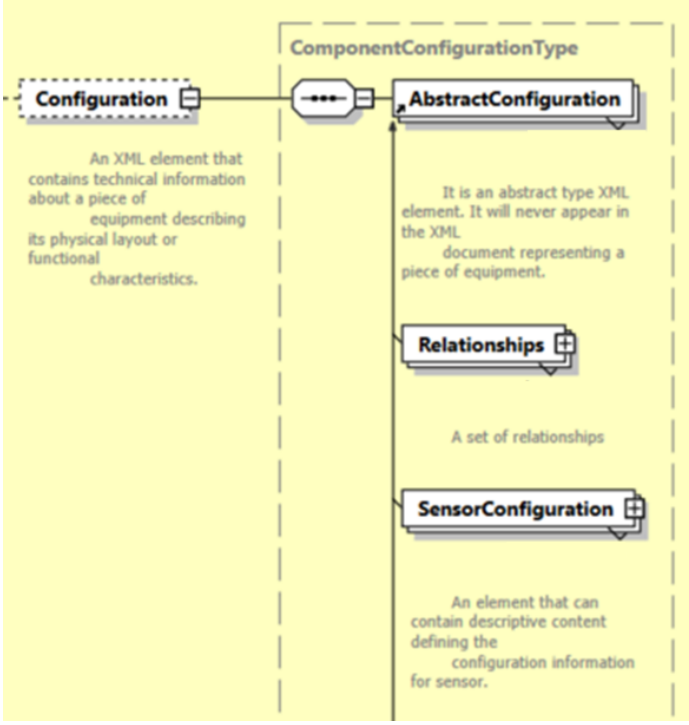

Figure 8: Component Configuration Diagram

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 - Data Entities for Device* 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 - XML Schema Structure for Component*.

Example 5: Example of parent Component and Child Elements

```
<Devices>
  <Device>
    <Components>
      <Axes> (Component)
        <Components>
          <Linear> (Component)
            <Components>
              <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 - References* 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.

Table 13: MTConnect Compositions Element

Element	Description	Occurrence
`Compositions`	An 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 - Composition Type Structural Elements* and include elements describing such basic building blocks as motors, amplifiers, filters, and pumps.

Example 6: Example of parent Component and child Composition elements

```
1 <Devices>
2   <Device>
3     <Components>
```

```
608  4        <Axes> (Component)
609  5          <Components>
610  6            <Linear> (Component)
611  7              <Compositions>
612  8                <Composition>
613  9                <Composition>
614 10                <Composition>
```

Table 14: MTConnect Composition Element

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..*

615 4.6.1 XML Schema Structure for Composition

616 *Figure 9* illustrates a `Composition` XML element showing the attributes defined for
617 `Composition` and the elements that may be associated with `Composition` type XML
618 elements.

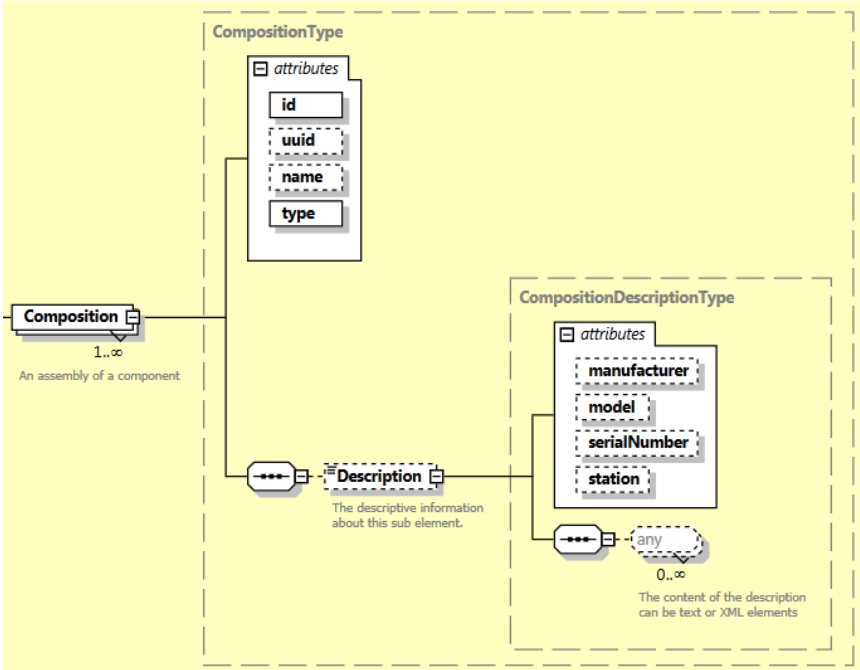

Figure 9: Composition Diagram

619 4.6.2 Attributes for Composition

620 *Table 15* defines the attributes that may be used to provide additional information for a
621 `Composition` type XML element.

Table 15: Attributes for Composition

Attribute	Description	Occurrence
`id`	The unique identifier for this element. `id` is a required attribute. An `id` **MUST** be unique across all the `id` attributes in the document. An XML ID-type.	1

Continuation of Table 15		
Attribute	Description	Occurrence
`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 - Composition Type Structural Elements* for a list of currently defined types.	1

4.6.3 Elements of Composition

Table 16 lists the elements defined to provide additional information for a `Composition` type XML element.

Table 16: Elements for Composition

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

4.6.3.1 Description for Composition

Figure 10 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) **MAY** be added to extend the schema for `Description`.

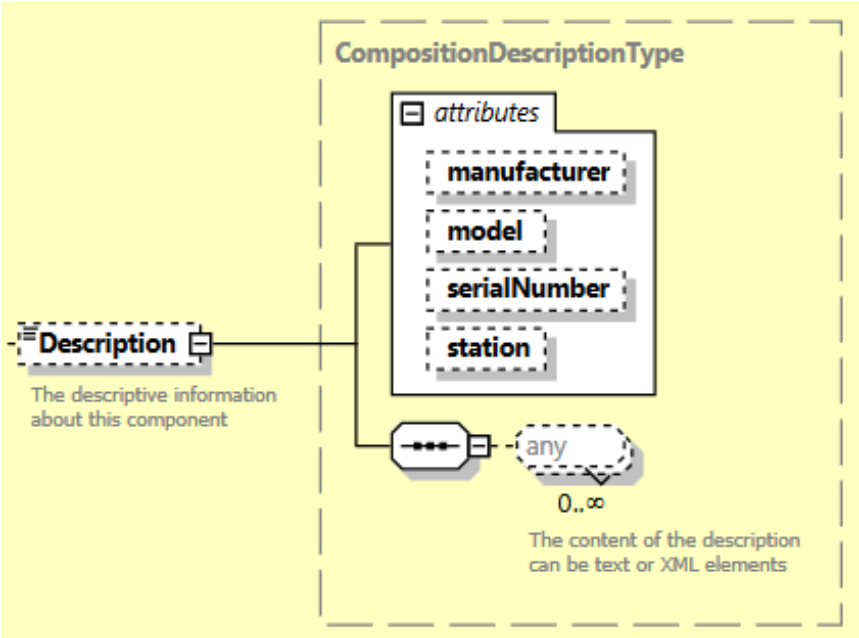

Figure 10: Description of Composition Diagram

Table 17 lists the attributes defined for the `Description` XML element.

Table 17: Attributes for Description for Composition

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

Continuation of Table 17		
Attribute	Description	Occurrence
`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

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

Example 7: Example of Description

```
1 <Description manufacturer="Example Co"
2     serialNumber="A124FFF" station="2"> Spindle motor
3     associated with Path 2.
4 </Description>
```

4.7 References

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

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

`References` contains one or more `Reference` XML elements.

Table 18: MTConnect References Element

Element	Description	Occurrence
References	An 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

4.8 Reference

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

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

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

Figure 11 represents the structure of the Reference XML element.

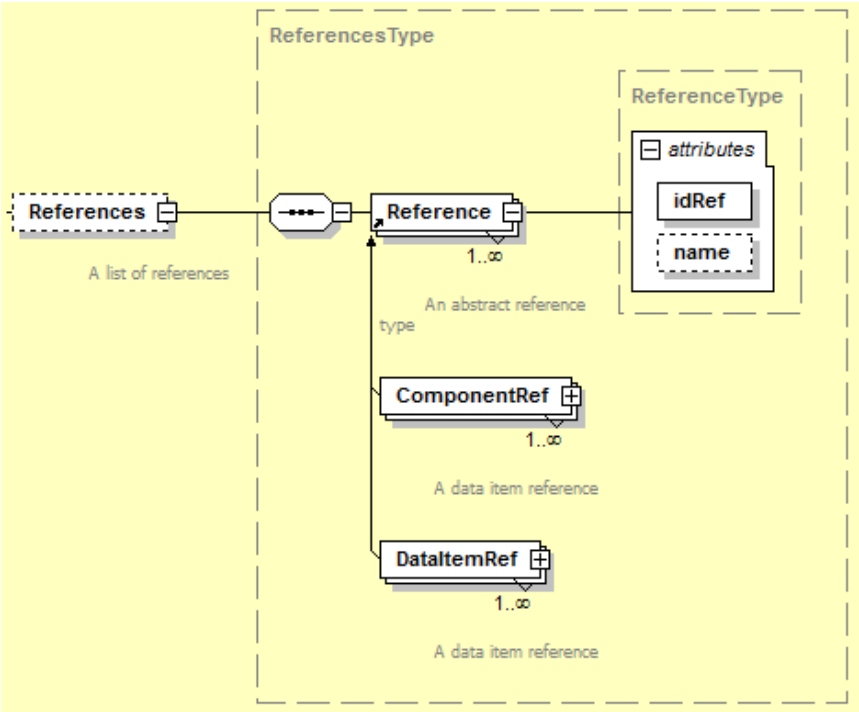

Figure 11: Reference Diagram

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.

Figure 12 represents the structure of a `ComponentRef` XML element showing the attributes defined for `ComponentRef`.

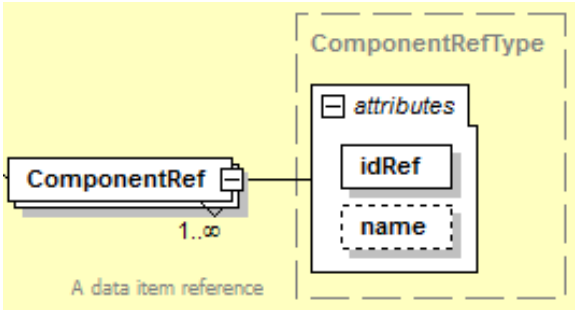

Figure 12: ComponentRef Diagram

668 *Table 19* lists the attributes defined for the `ComponentRef` element.

Table 19: Attributes for ComponentRef

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

669 4.8.2 DataItemRef

670 `DataItemRef` XML element is a pointer to a *Data Entity* associated with another *Struc-*
671 *tural Element* defined elsewhere in the XML document for a piece of equipment. `DataItem-`
672 `Ref` allows the data associated with a data item defined in another *Structural Element* to
673 be directly associated with this XML element.

674 *Figure 13* represents the structure of a `DataItemRef` XML element showing the at-
675 tributes defined for `DataItemRef`.

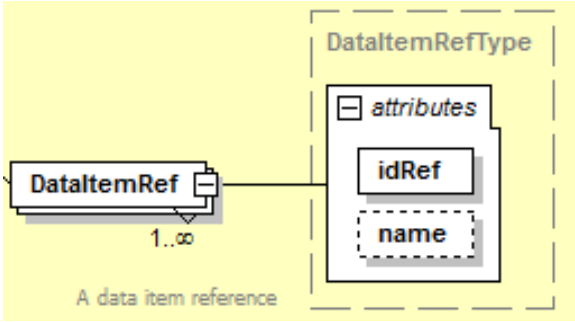

Figure 13: DataItemRef Diagram

676 *Table 20* lists the attributes defined for the `DataItemRef` element.

Table 20: Attributes for DataItemRef

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

4.9 Relationships

`Relationships` is an XML container that organizes information defining the association between pieces of equipment that function independently but together perform a manufacturing operation. `Relationships` may also define the association between components within a piece of equipment.

`Relationships` may be modeled as part of a `Device` or a `Component` *Structural Element*.

`Relationships` contains one or more `Relationship` XML elements.

Table 21: MTConnect Relationships Element

Element	Description	Occurrence
`Relationships`	XML container consisting of one or more `Relationship` XML elements. Only one `Relationships` container **MUST** appear for a `Device` or a `Component` element.	0..1

4.10 Relationship

`Relationship` is an XML element that describes the association between two pieces of equipment that function independently but together perform a manufacturing operation. `Relationship` may also be used to define the association between two components within a piece of equipment.

`Relationship` is an abstract type XML element, `Relationship` will be replaced in the XML document by specific `Relationship` types. XML elements representing `Relationship` are described in *Section 4.10.1 - DeviceRelationship* and *Section 4.10.2 - ComponentRelationship*.

A separate `Relationship` type element **MAY** be defined to describe each pair of associations with a piece of equipment or between `Component` elements within a piece of equipment.

Pieces of equipment may only be associated with other pieces of equipment and `Component` elements may only be associated with other `Component` elements within a specific piece of equipment.

The XML schema diagram in *Figure 14* represents the structure of the `Relationship` XML element.

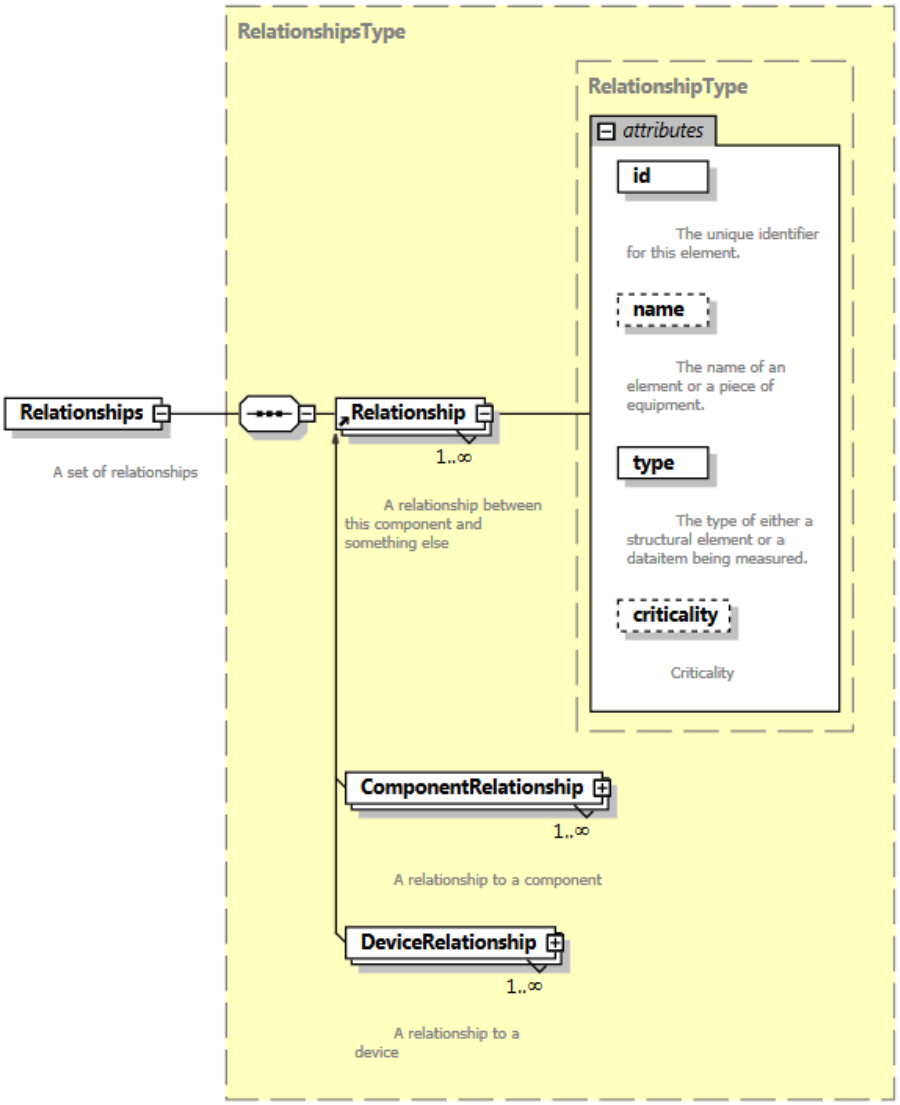

Figure 14: Relationship Diagram

4.10.1 DeviceRelationship

`DeviceRelationship` describes the association between two pieces of equipment that function independently but together perform a manufacturing operation.

The XML schema diagram in *Figure 15* represents the structure of a `DeviceRelationship` XML element showing the attributes defined for `DeviceRelationship`.

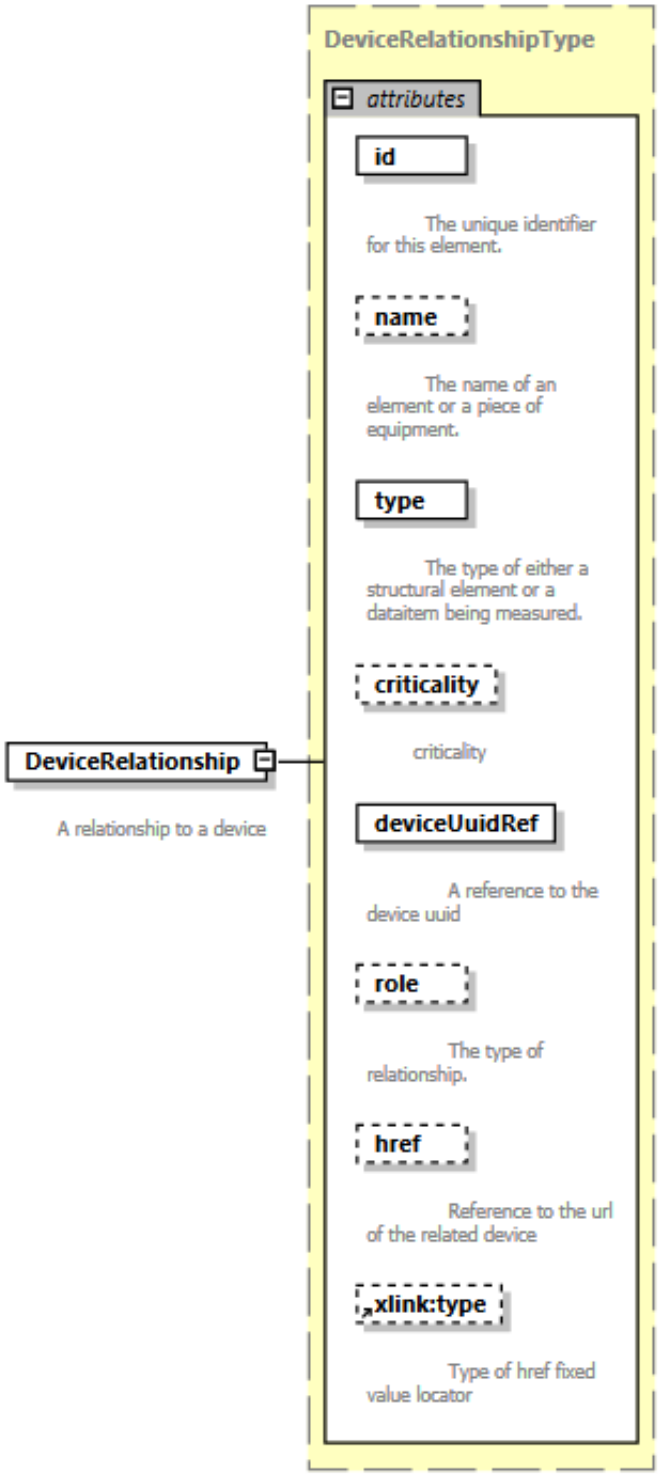

Figure 15: DeviceRelationship Diagram

707 The *Table 22* lists the attributes defined for the DeviceRelationship element.

Table 22: Attributes for DeviceRelationship

Attribute	Description	Occurrence
id	The unique identifier for this DeviceRelationship. id is a required attribute. The id attribute **MUST** be unique within the MTConnectDevices document. An XML ID-type.	1
name	The name associated with this DeviceRelationship. name is provided as an additional human readable identifier for this DeviceRelationship. name is an optional attribute. An NMTOKEN XML type.	0..1
type	Defines the authority that this piece of equipment has relative to the associated piece of equipment. type is a required attribute. The value provided for type **MUST** be one of the following values: PARENT: This piece of equipment functions as a parent in the relationship with the associated piece of equipment. CHILD: This piece of equipment functions as a child in the relationship with the associated piece of equipment. PEER: This piece of equipment functions as a peer which provides equal functionality and capabilities in the relationship with the associated piece of equipment.	1

Continuation of Table 22		
Attribute	Description	Occurrence
criticality	Defines whether the services or functions provided by the associated piece of equipment is required for the operation of this piece of equipment. criticality is an optional attribute. The value provided for criticality **MUST** be one of the following values: CRITICAL: The services or functions provided by the associated piece of equipment is required for the operation of this piece of equipment. NONCRITICAL: The services or functions provided by the associated piece of equipment is not required for the operation of this piece of equipment.	0..1
deviceUuidRef	A reference to the associated piece of equipment. The value provided for deviceUuidRef **MUST** be the value provided for the uuid attribute of the Device element of the associated piece of equipment. deviceUuidRef is a required attribute. An NMTOKEN XML type.	1

Continuation of Table 22		
Attribute	Description	Occurrence
`role`	Defines the services or capabilities that the referenced piece of equipment provides relative to this piece of equipment. `role` is an optional attribute. The value provided for `role` **MUST** be one of the following values: `SYSTEM`: The associated piece of equipment performs the functions of a `System` for this piece of equipment. In MTConnect, `System` provides utility type services to support the operation of a piece of equipment and these services are required for the operation of a piece of equipment. `AUXILIARY`: The associated piece of equipment performs the functions as an `Auxiliary` for this piece of equipment. In MTConnect, `Auxiliary` extends the capabilities of a piece of equipment, but is not required for the equipment to function.	0..1
`href`	A URI identifying the *Agent* that is publishing information for the associated piece of equipment. `href` **MUST** also include the UUID for that specific piece of equipment. `href` is of type `xlink:href` from the W3C XLink specification: (https://www.w3.org/TR/xlink11/). `href` is an optional attribute.	0..1
`xlink:type`	The XLink `type` attribute **MUST** have a fixed value of `locator` as defined in W3C XLink 1.1 https://www.w3.org/TR/xlink11/ *section 5.4 Locator Attribute (`href`).* If the `href` attribute is provided, it **MUST** conform to the URI syntactic rules as defined in IETF RFC 3986 for Uniform Resource Identifiers. (https://www.ietf.org/rfc/rfc3986.txt)	0..1

4.10.2 ComponentRelationship

ComponentRelationship describes the association between two components within a piece of equipment that function independently but together perform a capability or service within a piece of equipment.

The XML schema in *Figure 16* represents the structure of a ComponentRelationship XML element showing the attributes defined for ComponentRelationship.

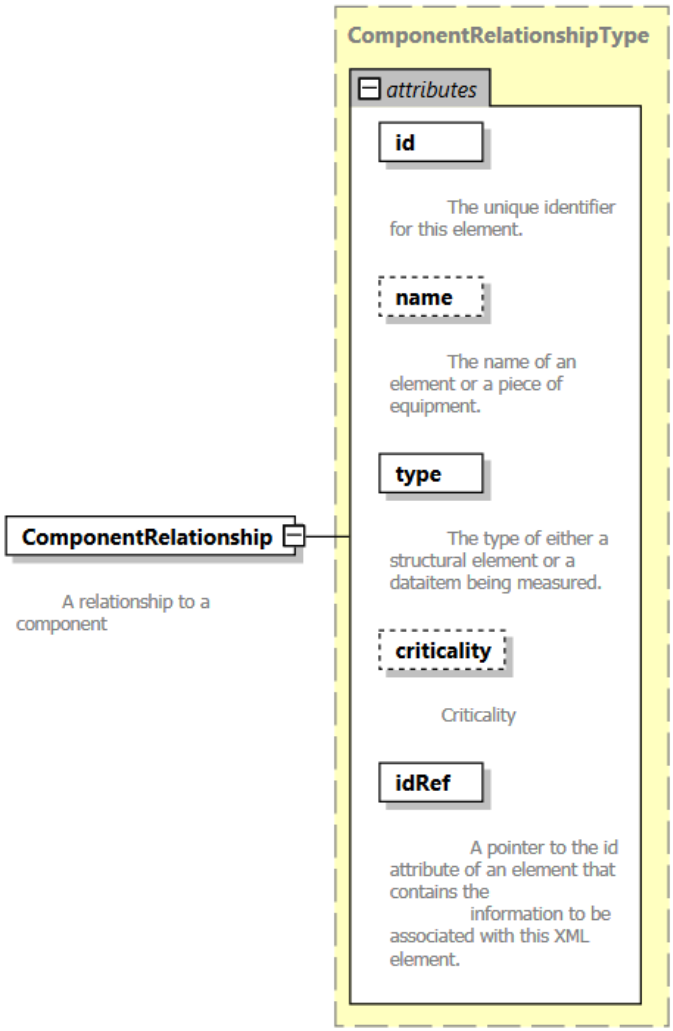

Figure 16: ComponentRelationship Diagram

The *Table 23* lists the attributes defined for the ComponentRelationship element.

Table 23: Attributes for ComponentRelationship

Attribute	Description	Occurrence
`id`	The unique identifier for this `ComponentRelationship`. `id` is a required attribute. The `id` attribute **MUST** be unique within the `MTConnectDevices` document. An XML ID-type.	1
`name`	The name associated with this `ComponentRelationship`. `name` is provided as an additional human readable identifier for this `ComponentRelationship`. `name` is an optional attribute. An NMTOKEN XML type.	0..1
`type`	Defines the authority that this component element has relative to the associated component element. `type` is a required attribute. The value provided for `type` **MUST** be one of the following values: `PARENT`: This component functions as a parent in the relationship with the associated component element. `CHILD`: This component functions as a child in the relationship with the associated component element. `PEER`: This component functions as a peer which provides equal functionality and capabilities in the relationship with the associated component element.	1

Continuation of Table 23		
Attribute	Description	Occurrence
`criticality`	Defines whether the services or functions provided by the associated component element is required for the operation of this piece of equipment. `criticality` is an optional attribute. The value provided for `criticality` **MUST** be one of the following values: `CRITICAL`: The services or functions provided by the associated component element is required for the operation of this piece of equipment. `NONCRITICAL`: The services or functions provided by the associated component element is not required for the operation of this piece of equipment.	0..1
`idRef`	A reference to the associated component element. The value provided for `idRef` **MUST** be the value provided for the `id` attribute of the associated `Component` element. `idRef` is a required attribute. An NMTOKEN XML type.	1

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 *Table 24*. 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 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 - Structural Elements for MTConnectDevices*, `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.

Table 24 defines the *Top Level* `Component` elements available to describe a piece of equipment.

Table 24: Top Level Component Elements

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.

Continuation of Table 24	
Top Level Component Element ††	Description
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.
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 - Sensor* 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.

Figure 17 defines the relationship between the `Axes` container and the individual axis type *Structural Elements*.

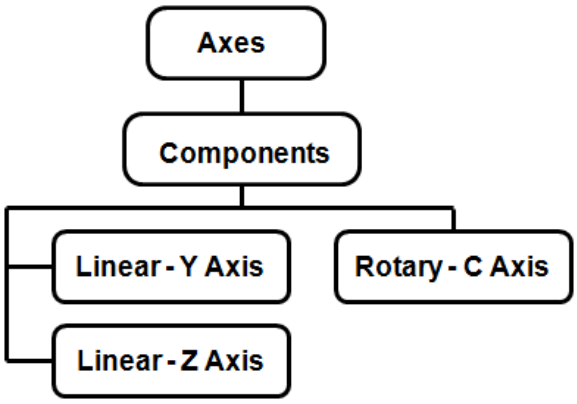

Figure 17: Axes Example with Two Linear Axes and One Rotary Axis

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` **MUST** 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` is a component that organizes and 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.

5.3.11 EndEffector System

`EndEffector` is an XML container that represents the information for those functions that form the last link segment of a piece of equipment. It is the part of a piece of equipment that interacts with the manufacturing 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 - Sensor Unit* 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 - Sensor*.

5.4.7 Deposition System

`Deposition` is an XML container that represents the information for a system that manages the addition of material or state change of material being performed in an additive manufacturing process. For example, this could describe the portion of a piece of equipment that manages a material extrusion process or a vat polymerization process.

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.

5.6 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 *MTConnect Standard: Part 5.0 - Interfaces* 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 - Sensor* 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`.

Table 25 defines `Composition` type elements that are currently available to describe sub-parts of a `Component` element.

Table 25: Composition type Elements

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.

Continuation of Table 25	
Element Type	Description
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.
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.
CHOPPER	A mechanism used to break material into smaller pieces.
CHUCK	A mechanism that holds a part, stock material, or any other item in place.
CHUTE	An inclined channel for conveying material.
CIRCUIT_BREAKER	A mechanism for interrupting an electric circuit.
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.
EXPOSURE_UNIT	A mechanism for emitting a type of radiation
EXTRUSION_UNIT	A mechanism for dispensing liquid or powered materials
FAN	Any mechanism for producing a current of air.

Continuation of Table 25	
Element Type	Description
FILTER	Any substance or structure through which liquids or gases are passed to remove suspended impurities or to recover solids.
GALVANOMOTOR	An electromechanical actuator that produces deflection of a beam of light or energy in response to electric current through its coil in a magnetic field.
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.
LINEAR_POSITION_FEEDBACK	A mechanism that measures linear motion or position.
MOTOR	A mechanism that converts electrical, pneumatic, or hydraulic energy into mechanical energy.
OIL	A viscous liquid.
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.
PUMP	An apparatus raising, driving, exhausting, or compressing fluids or gases by means of a piston, plunger, or set of rotating vanes.
REEL	A rotary storage unit for material
SENSING_ELEMENT	A mechanism that provides a signal or measured value.
SPREADER	A mechanism for flattening or spreading materials

Continuation of Table 25	
Element Type	Description
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.
TABLE	A surface for holding an object or material
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.
VAT	A container for liquid or powdered materials
WATER	A fluid.
WIRE	A string like piece or filament of relatively rigid or flexible material provided in a variety of diameters.

994 Note: As the MTConnect Standard evolves, more Composition types will be
995 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 *MTConnect Standard: Part 3.0 - Streams Information Model* for detail 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 Entity*. 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 describe that data. The different types of DataItem elements are defined in *Section 8 - Listing of Data Items*.

Figure 18 demonstrates the relationship between *Data Entities* (DataItem) and the various *Structural Elements* in the MTConnectDevices XML document.

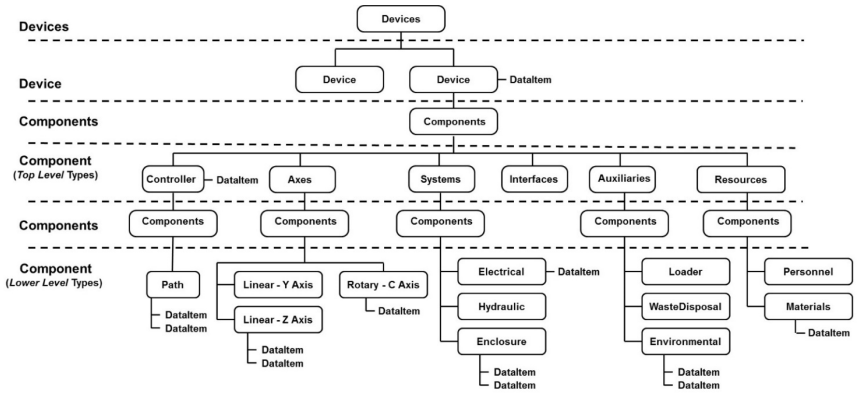

Figure 18: Example Data Entities for Device (DataItem)

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.

Table 26: MTConnect DataItems Element

Element	Description	Occurrence
DataItems	An 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 an *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.

Table 27: MTConnect DataItem Element

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

7.2.1 XML Schema Structure for DataItem

Figure 19 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.

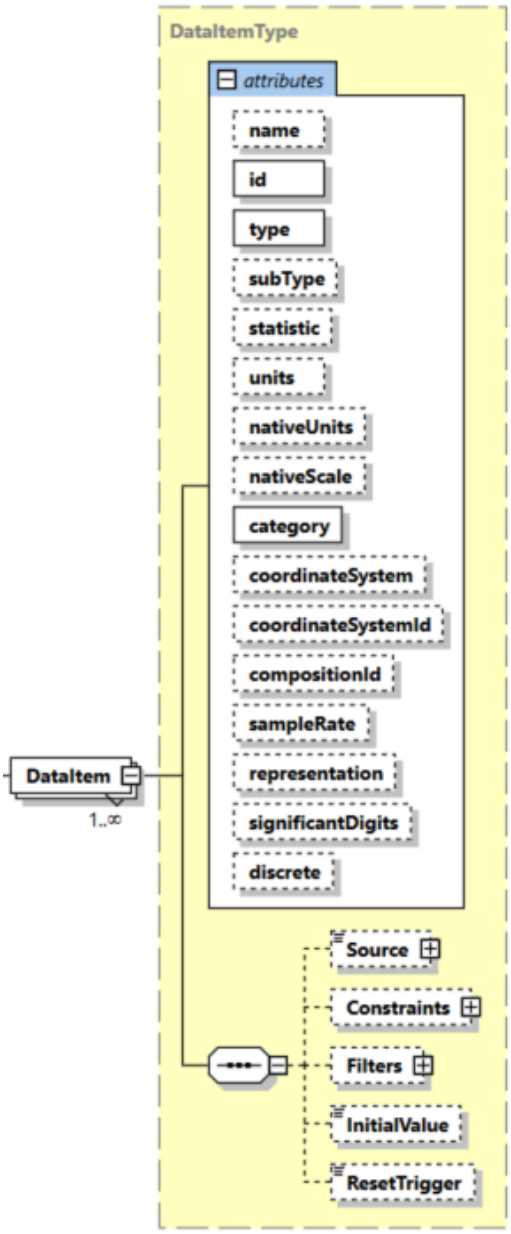

Figure 19: DataItem Diagram

7.2.2 Attributes for DataItem

Table 28 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.

Table 28: Attributes for 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 element. 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

Continuation of Table 28		
Attribute	Description	Occurrence
`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
`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 7.2.2.5 - units Attribute for DataItem* 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 7.2.2.6 - nativeUnits Attribute for DataItem* for a list of available native units identified in the MTConnect Standard.	0..1

Continuation of Table 28		
Attribute	Description	Occurrence
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/MINUTE. 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

Continuation of Table 28		
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 as a single value. `representation` is an optional attribute. `representation` defines the unique format for each set of data. `representation` for `TIME_SERIES`, `DISCRETE` (**DEPRECATED** in *Version 1.5*), `DATA_SET`, and `VALUE` are defined in *Section 7.2.2.12 - representation Attribute for DataItem*. 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

Continuation of Table 28		
Attribute	Description	Occurrence
`discrete`	An indication signifying whether each value reported for the *Data Entity* is significant and whether duplicate values are to be suppressed. The value defined **MUST** be either `true` or `false` - an XML boolean type. `true` indicates that each update to the *Data Entity*'s value is significant and duplicate values **MUST NOT** be suppressed. `false` indicates that duplicated values **MUST** be suppressed. If a value is not defined for `discrete`, the default value **MUST** be `false`.	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.

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

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

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

Section 8 - Listing of Data Items provides a detailed listing of the data item `type` and `subType` elements defined for each `category` of data item available for a piece of equipment: `SAMPLE`, `EVENT`, and `CONDITION`.

7.2.2.4 statistic Attribute for DataItem

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

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

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

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

Table 29 shows the `statistic` calculations that can be defined for a `DataItem`.

Table 29: DataItem attribute statistic type

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.
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.

7.2.2.5 units Attribute for DataItem

Table 30 lists the units that are defined as the standard unit of measure for each type of DataItem. All SAMPLE type data items **MUST** report data values in standard units.

Table 30: DataItem attribute units type

Units	Description
AMPERE	Amps
CELSIUS	Degrees Celsius
COUNT	A count of something.
CUBIC_MILLIMETER	Geometric volume in millimeters
CUBIC_MILLIMETER/SECOND	Change of geometric volume per second
CUBIC_MILLIMETER/SECOND^2	Change in geometric volume per second squared
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.
KILOGRAM	Kilograms
LITER	Measurement of volume of a fluid
LITER/SECOND	Liters per second
MICRO_RADIAN	Measurement of Tilt
MILLIGRAM	Milligram
MILLIGRAM/CUBIC_MILLIMETER	Milligram per cubic millimeter
MILLILITER	Milliliter
MILLIMETER	Millimeters
MILLIMETER/REVOLUTION	Millimeters per revolution.
MILLIMETER/SECOND	Millimeters per second

Continuation of Table 30	
Units	Description
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

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.

Table 31 defines the `nativeUnits` currently supported by the `MTConnectDevices` XML document:

Table 31: DataItem attribute nativeunits type

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.
`HOUR`	A measurement of time in hours
`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.
`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

Continuation of Table 31	
Native Units	Description
MINUTE	A measurement of time in minutes
OTHER	Unsupported units
POUND	US pounds
POUND/INCH^2	Pressure in pounds per square inch (PSI).
RADIAN	Angle in radians
RADIAN/MINUTE	Velocity in radians per minute.
RADIAN/SECOND	Rotational acceleration in radian per second squared
RADIAN/SECOND^2	Rotational acceleration in radian per second squared
REVOLUTION/SECOND	Rotational velocity in revolution per second

7.2.2.7 nativeScale Attribute for DataItem

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

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

7.2.2.8 category Attribute for DataItem

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

Each *Data Entity* provided by a piece of equipment **MUST** be identified with one of the following: SAMPLE, EVENT, CONDITION.

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 a linear axis called X.

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.

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, CLOSED, or UNLATCHED. (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.

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.

Table 32 defines the types of `coordinateSystem` currently supported by the `MTConnectDevices` XML document:

Table 32: DataItem attribute coordinateSystem type

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 *Table 33*.

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

Table 33: DataItem attribute representation type

Representation	Description
DATA_SET	The reported value(s) are represented as a set of *key-value pairs*. Each reported value in the *Data Set* **MUST** have a unique key.

Continuation of Table 33	
Representation	Description
DISCRETE (**DEPRECATED** in *Version 1.5*)	**DEPRECATED** as a representation in MTConnect Version. 1.5. Replaced by the discrete attribute for a *Data Entity – Section 7.2.2.14 - discrete Attribute for DataItem*. ~~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.~~
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.
VALUE	The measured value of the sample data. If no representation is specified for a data item, the representation **MUST** be determined to be VALUE.

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.

7.2.2.14 discrete Attribute for DataItem

An indication signifying whether each value reported for the *Data Entity* is significant and whether duplicate values are to be suppressed.

The value defined **MUST** be either `true` or `false` - an XML boolean type.

`true` indicates that each update to the *Data Entity*'s value is significant and duplicate values **MUST NOT** be suppressed.

`false` indicates that duplicated values **MUST** be suppressed.

If a value is not defined for `discrete`, the default value **MUST** be `false`.

7.2.3 Elements for DataItem

Table 34 lists the elements defined to provide additional information for a `DataItem` type XML element.

Table 34: Elements for DataItem

Element	Description	Occurrence
`Source`	`Source` is an optional XML element that identifies the `Component`, `DataItem`, or `Composition` representing the area of the piece of equipment from which a measured value originates. Additionally, `Source` **MAY** provide information relating to the identity of a measured value. This information is reported as CDATA for `Source`. (example, a PLC tag)	0..1

Continuation of Table 34		
Element	Description	Occurrence
`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 may be used to identify the physical part of a piece of equipment where the data represented by `DataItem` originated and/or it may be used to identify a complex name or an alternate name used to identify the data where it originated (e.g. a PLC tag name).

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. Additionally, a second or alternate name may be required to describe a piece of data. An example of this case would be the identity of the bit address in a PLC that represents this piece of data (PLC address I0015.4). When these cases occur, the alternate name can be provided as the value for the CDATA for `Source`.

The XML schema in *Figure 20* represents the structure of the `Source` XML element showing the attributes defined for `Source`.

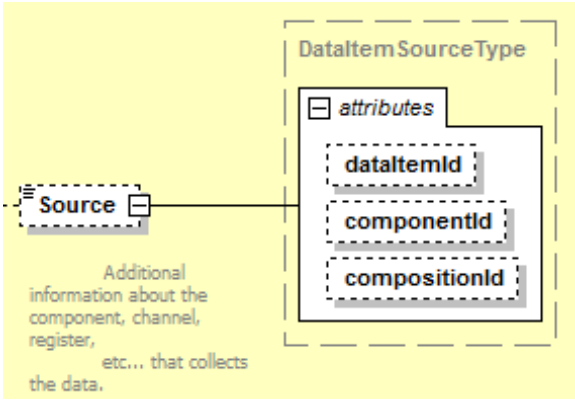

Figure 20: Source Diagram

7.2.3.1.1 Attributes for Source

Table 35 identifies the attributes available to identify `Source` for a measured value:

Table 35: Attributes for Source

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

Continuation of Table 35		
Attribute	Description	Occurrence
`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 Request* 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

1218 The XML schema in *Figure 21* represents the structure of the `Constraints` XML
1219 element and the elements defined for `Constraints`.

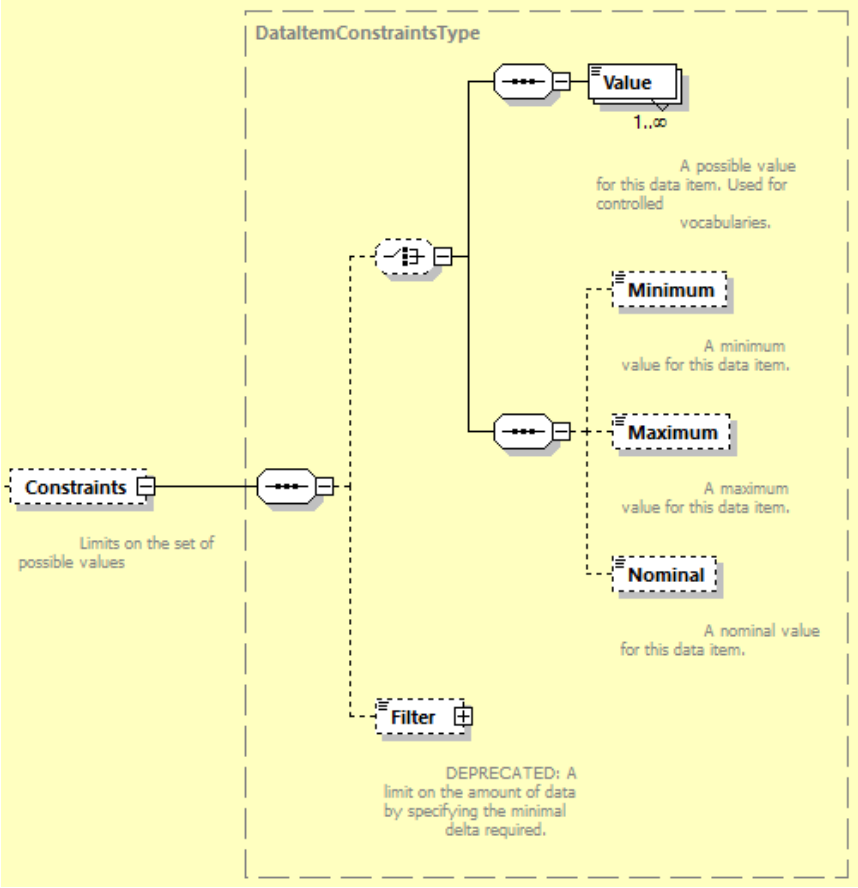

Figure 21: Constraints Diagram

1220 *Table 36* identifies the elements available to identify `Constraints` for a measured value:

Table 36: Elements for Constraints

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 Request* or *Sample Request* request **MUST** be the data value defined for `Value` element. `Value` **MUST NOT** be used in conjunction with any other `Constraint` elements.	0..*
`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 a 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 a 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 a value using the same units as the reported data.	0..1

Continuation of Table 36		
Element	Description	Occurrence
~~`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 †

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.

Table 37: MTConnect Filters Element

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 an *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.

Figure 22 represents the structure for `Filter` XML element.

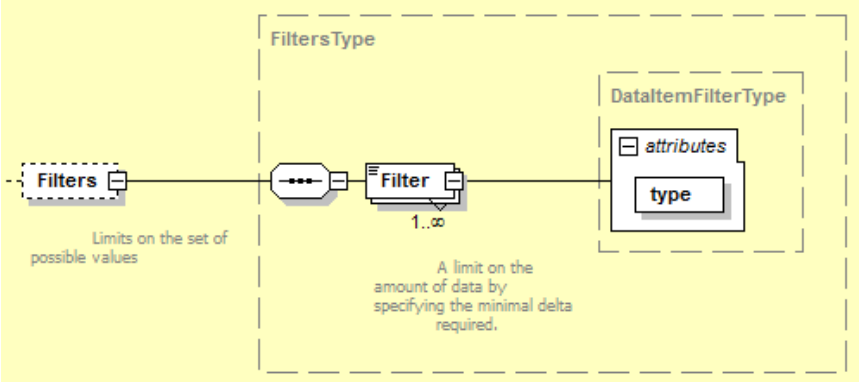

Figure 22: Filter Diagram

Table 38 describes the types of `Filter` defined for a `DataItem` element and the expected behavior of an *Agent* when a `Filter` is applied to `DataItem` element.

Table 38: DataItem Element Filter type

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 †

Continuation of Table 38		
type	Description	Occurrence
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.

Table 39: MTConnect ResetTrigger Element

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 *Table 40*.

Table 40: DataItem Element ResetTrigger type

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 Entity* is not reset and accumulates for the entire life of the piece of equipment.
`MAINTENANCE`	The value of the *Data Entity* 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.

Continuation of Table 40	
Reset Actions	Description
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.

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.

Table 41 defines the types and subtypes of DataItem elements defined for the SAMPLE category. The subtypes are indented below their associated types.

Table 41: DataItem type subType for category SAMPLE

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 EQUIPMENT_TIMER.	SECOND
AMPERAGE	The measurement of electrical current.	AMPERE
ACTUAL	The measured amperage being delivered from a power source.	AMPERE
ALTERNATING	The measurement of alternating current. If not specified further in statistic, defaults to RMS voltage.	AMPERE
DIRECT	The measurement of DC current.	AMPERE

Continuation of Table 41: DataItem type subType for category SAMPLE		
DataItem type/subType	Description	Units
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
ANGULAR_ACCELERATION	Rate of change of angular velocity.	DEGREE/SECOND^2
ANGULAR_VELOCITY	Rate of change of angular position.	DEGREE/SECOND
AXIS_FEEDRATE	The feedrate of a linear axis.	MILLIMETER/SECOND
ACTUAL	The measured value of the feedrate of a linear axis.	MILLIMETER/SECOND
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

Continuation of Table 41: DataItem type subType for category SAMPLE		
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 a linear axis when operating in a manual state or method (jogging).	MILLIMETER/SECOND
~~OVERRIDE~~	~~The operator's overridden value. Percent of commanded.~~ **DEPRECATED** in Version 1.3. See EVENT category data items.	~~PERCENT~~
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
CAPACITY_FLUID	The fluid capacity of an object or container.	MILLILITER
CAPACITY_SPATIAL	The geometric capacity of an object or container.	CUBIC_MILLIMETER
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

Continuation of Table 41: DataItem type subType for category SAMPLE		
DataItem type/subType	Description	Units
CONCENTRATION	Percentage of one component within a mixture of components.	PERCENT
CONDUCTIVITY	The ability of a material to conduct electricity.	SIEMENS/METER
CUTTING_SPEED	The speed difference (relative velocity) between the cutting mechanism and the surface of the workpiece it is operating on.	MILLIMETER/SECOND
ACTUAL	The measured value between the cutting mechanism and the surface of the workpiece it is operating on.	MILLIMETER/SECOND
COMMANDED	The commanded value between the cutting mechanism and the surface of the workpiece it is operating on.	MILLIMETER/SECOND
PROGRAMMED	The programmed value between the cutting mechanism and the surface of the workpiece it is operating on.	MILLIMETER/SECOND
DENSITY	The volumetric mass of a material per unit volume of that material.	MILLIGRAM/CUBIC_-MILLIMETER
DEPOSITION_-ACCELERATION_-VOLUMETRIC	The rate of change in spatial volume of material deposited in an additive manufacturing process.	CUBIC_-MILLIMETER/SECOND^2

Continuation of Table 41: DataItem type subType for category SAMPLE		
DataItem type/subType	Description	Units
ACTUAL	The measured rate of change in spatial volume of material deposited in an additive manufacturing process.	CUBIC_-MILLIMETER/SECOND^2
COMMANDED	The commanded rate of change in spatial volume of material to be deposited in an additive manufacturing process.	CUBIC_-MILLIMETER/SECOND^2
DEPOSITION_DENSITY	The density of the material deposited in an additive manufacturing process per unit of volume.	MILLIGRAM/CUBIC_-MILLIMETER
ACTUAL	The measured density of the material deposited in an additive manufacturing process.	MILLIGRAM/CUBIC_-MILLIMETER
COMMANDED	The commanded density of material to be deposited in an additive manufacturing process.	MILLIGRAM/CUBIC_-MILLIMETER
DEPOSITION_MASS	The mass of the material deposited in an additive manufacturing process.	MILLIGRAM
ACTUAL	The measured mass of the material deposited in an additive manufacturing process.	MILLIGRAM
COMMANDED	The commanded mass of the material to be deposited in an additive manufacturing process.	MILLIGRAM

Continuation of Table 41: DataItem type subType for category SAMPLE		
DataItem type/subType	Description	Units
DEPOSITION_RATE_-VOLUMETRIC	The rate at which a spatial volume of material is deposited in an additive manufacturing process.	CUBIC_-MILLIMETER/SECOND
ACTUAL	The measured rate at which a spatial volume of material is deposited in an additive manufacturing process.	CUBIC_-MILLIMETER/SECOND
COMMANDED	The programmed rate at which a spatial volume of material is to be deposited in an additive manufacturing process.	CUBIC_-MILLIMETER/SECOND
DEPOSITION_VOLUME	The spatial volume of material to be deposited in an additive manufacturing process.	CUBIC_MILLIMETER
ACTUAL	The measured spatial volume of material deposited.	CUBIC_MILLIMETER
COMMANDED	The target spatial volume of material to be deposited.	CUBIC_MILLIMETER
DISPLACEMENT	The change in position of an object.	MILLIMETER
ELECTRICAL_ENERGY	The measurement of electrical energy consumption by a component.	WATT_SECOND

Continuation of Table 41: DataItem type subType for category SAMPLE		
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
DELAY	Measurement of the time that a piece of equipment is waiting for an event or an action to occur.	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

Continuation of Table 41: DataItem type subType for category SAMPLE		
DataItem type/subType	Description	Units
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
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

Continuation of Table 41: DataItem type subType for category SAMPLE		
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	None
LENGTH	The length of an object.	MILLIMETER
REMAINING	The remaining total length of an object.	MILLIMETER
STANDARD	The standard or original length of an object.	MILLIMETER
USEABLE	The remaining useable length of an object.	MILLIMETER
~~LEVEL~~	**DEPRECATED** in Version 1.2. See FILL_LEVEL	None
LINEAR_FORCE	The measurement 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

Continuation of Table 41: DataItem type subType for category SAMPLE		
DataItem type/subType	Description	Units
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
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
~~OVERRIDE~~	~~The operator's overridden value. Percent of commanded.~~ **DEPRECATED** in Version 1.3. See EVENT category data items.	~~PERCENT~~

Continuation of Table 41: DataItem type subType for category SAMPLE		
DataItem type/subType	Description	Units
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
PATH_FEEDRATE_-PER_REVOLUTION	The feedrate for the axes, or a single axis.	MILLIMETER/REVO-LUTION
ACTUAL	The measured value of the feedrate of the axes, or a single axis.	MILLIMETER/REVO-LUTION
COMMANDED	The feedrate as specified by the Controller for the axes, or a single axis. The COMMANDED feedrate is a calculated value that includes adjustments and overrides.	MILLIMETER/REVO-LUTION
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.	MILLIMETER/REVO-LUTION

Continuation of Table 41: DataItem type subType for category SAMPLE		
DataItem type/subType	Description	Units
PATH_POSITION	A measured or calculated position of a control point associated with 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 coordinateSystem 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

Continuation of Table 41: DataItem type subType for category SAMPLE		
DataItem type/subType	Description	Units
PROGRAMMED	The position of the control point specified by a logic or motion program.	MILLIMETER_3D
COMMANDED	The position computed by the Controller type component.	MILLIMETER_3D
PROBE	The position provided by a measurement probe.	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
PH	The measurement of the acidity or alkalinity.	PH
POSITION	A measured or calculated position of a Component element as reported by a piece of equipment. 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

Continuation of Table 41: DataItem type subType for category SAMPLE		
DataItem type/subType	Description	Units
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

Continuation of Table 41: DataItem type subType for category SAMPLE		
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
DELAY	Measurement of the time that a process is waiting and unable to perform its intended function.	SECOND

Continuation of Table 41: DataItem type subType for category SAMPLE		
DataItem type/subType	Description	Units
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
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
~~OVERRIDE~~	~~The operator's overridden value. Percent of commanded.~~ **DEPRECATED** in Version 1.3. See EVENT category data items.	~~PERCENT~~

Continuation of Table 41: DataItem type subType for category SAMPLE		
DataItem type/subType	Description	Units
PROGRAMMED	The rotational velocity specified by a logic or motion program or set by a switch.	REVOLUTION/MINUTE
SOUND_LEVEL	The measurement of a sound level or sound pressure level relative to atmospheric pressure.	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
NO_SCALE	No weighting factor on the frequency scale	DECIBEL
~~SPINDLE_SPEED~~	**DEPRECATED** in Version 1.2. Replaced by ROTARY_VELOCITY	~~REVOLUTION/MINUTE~~
~~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

Continuation of Table 41: DataItem type subType for category SAMPLE		
DataItem type/subType	Description	Units
TEMPERATURE	The measurement of temperature.	CELSIUS
TENSION	The measurement of a force that stretches or elongates an object.	NEWTON
TILT	The measurement of angular displacement.	MICRO_RADIAN
TORQUE	The turning force exerted on an object or by an object.	NEWTON_METER
VELOCITY	The rate of change of position.	MILLIMETER/SECOND
VISCOSITY	The measurement of a fluids resistance to flow.	PASCAL_SECOND
VOLTAGE	The measurement of electrical potential between two points.	VOLT
ACTUAL	The measured voltage being delivered from a power source.	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
TARGET	The desired or preset voltage to be delivered from a power source.	VOLT

Continuation of Table 41: DataItem type subType for category SAMPLE		
DataItem type/subType	Description	Units
VOLT_AMPERE	The measurement 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
VOLUME_FLUID	The fluid volume of an object or container.	MILLILITER
ACTUAL	The amount of fluid currently present in an object or container.	MILLILITER
CONSUMED	The amount of fluid material consumed from an object or container during a manufacturing process.	MILLILITER
VOLUME_SPATIAL	The geometric volume of an object or container.	CUBIC_MILLIMETER
ACTUAL	The amount of bulk material currently present in an object or container.	CUBIC_MILLIMETER
CONSUMED	The amount of bulk material consumed from an object or container during a manufacturing process.	CUBIC_MILLIMETER

Continuation of Table 41: DataItem type subType for category SAMPLE		
DataItem type/subType	Description	Units
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.

Table 42 defines the DataItem types and subtypes defined for the EVENT category. The subtypes are indented below their associated types.

Table 42: DataItem type subType for category EVENT

DataItem type subType	Description
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.
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.
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.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
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.

Continuation of Table 42: DataItem type subType for category EVENT	
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.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
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.
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.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
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.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
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.
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.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
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.
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

Continuation of Table 42: DataItem type subType for category EVENT	
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.
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.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
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.
DATE_CODE	The time and date code associated with a material or other physical item. DATE_CODE **MUST** be reported in ISO 8601 format.
MANUFACTURE	The time and date code relating to the production of a material or other physical item.
EXPIRATION	The time and date code relating to the expiration or end of useful life for a material or other physical item.
FIRST_USE	The time and date code relating the first use of a material or other physical item.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
DEVICE_UUID	The identifier of another piece of equipment that is temporarily associated with a component of this piece of equipment to perform a particular function. The *Valid Data Value* **MUST** be a NMTOKEN XML type.
DIRECTION	The direction of motion. A subType **MUST** always be specified.
LINEAR	The direction of motion of a linear motion. The *Valid Data Value* **MUST** be POSITIVE or NEGATIVE.
ROTARY	The rotational direction of a rotary motion using the right hand rule convention. The *Valid Data Value* **MUST** be CLOCKWISE or COUNTER_CLOCKWISE.
DOOR_STATE	The operational state of a DOOR type component or composition element. The *Valid Data Value* **MUST** be OPEN, UNLATCHED, or CLOSED.
EMERGENCY_STOP	The current state of the emergency stop signal for a piece of equipment, controller path, or any other component or subsystem of a piece of equipment. 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).
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.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
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.
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.
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.
DELAY	An indication that a piece of equipment is waiting for an event or an action to occur.
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.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
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.
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.
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.
EXECUTION	The execution status of the Controller. The *Valid Data Value* **MUST** be READY, ACTIVE, INTERRUPTED, WAIT, FEED_HOLD, STOPPED, OPTIONAL_STOP, PROGRAM_STOPPED, or PROGRAM_COMPLETED.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
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.
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.
ROCKWELL	A scale to measure the resistance to deformation of a surface.
SHORE	A scale to measure the resistance to deformation of a surface.
VICKERS	A scale to measure the resistance to deformation 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.

Continuation of Table 42: DataItem type subType for category EVENT	
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 1.4.0.
~~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.
MATERIAL_LAYER	Identifies the layers of material applied to a part or product as part of an additive manufacturing process. The *Valid Data Value* **MUST** be an integer.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
ACTUAL	The current number of layers of material applied to a part or product during an additive manufacturing process.
TARGET	The target or planned number layers of material applied to a part or product during an additive manufacturing process.
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 component. 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.
BAD	Indicates the count of incorrect parts produced.
GOOD	Indicates the count of correct parts made.
REMAINING	The number of parts remaining in stock or to be produced.
TARGET	Indicates the number of parts that are projected or planned to be produced.
PART_DETECT	An indication designating whether a part or work piece has been detected or is present. The *Valid Data Value* **MUST** be PRESENT or NOT_PRESENT.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
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. **DEPRECATION WARNING** : May be deprecated in the future.
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.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
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.
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.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
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.
CONTROL	The state of the enabling signal or control logic that enables or disables the function or operation of the *Structural Element*.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
LINE	The state of the power source for the *Structural Element*.
~~POWER_STATUS~~	**DEPRECATED** in Version 1.1.0.
PROCESS_TIME	The time and date associated with an activity or event. PROCESS_TIME **MUST** be reported in ISO 8601 format.
START	The time and date associated with the beginning of an activity or event.
COMPLETE	The time and date associated with the completion of an activity or event.
TARGET_COMPLETION	The projected time and date associated with the end or completion of an activity or event.
PROGRAM	The identity of the logic or motion program being executed by the piece of equipment. The *Valid Data Value* **MUST** be a text string.
SCHEDULE	The identity of a control program that is used to specify the order of execution of other programs.
MAIN	The identity of the primary logic or motion program currently being executed. It is the starting nest level in a call structure and may contain calls to sub programs.
ACTIVE	The identity of the logic or motion program currently executing.
PROGRAM_COMMENT	A comment or non-executable statement in the control program. The *Valid Data Value* **MUST** be a text string.
SCHEDULE	The identity of a control program that is used to specify the order of execution of other programs.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
MAIN	The identity of the primary logic or motion program currently being executed. It is the starting nest level in a call structure and may contain calls to sub programs.
ACTIVE	The identity of the logic or motion program currently executing.
PROGRAM_EDIT	An indication of the status of the Controller components 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_HEADER	The non-executable header section of the control program. The *Valid Data Value* **MUST** be a text string.
PROGRAM_LOCATION	The Uniform Resource Identifier (URI) for the source file associated with PROGRAM.
SCHEDULE	An identity of a control program that is used to specify the order of execution of other programs.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
MAIN	The identity of the primary logic or motion program currently being executed. It is the starting nest level in a call structure and may contain calls to sub programs.
ACTIVE	The identity of the logic or motion program currently executing.
PROGRAM_LOCATION_TYPE	Defines whether the logic or motion program defined by PROGRAM is being executed from the local memory of the controller or from an outside source. The *Valid Data Value* **MUST** be LOCAL or EXTERNAL.
SCHEDULE	An identity of a control program that is used to specify the order of execution of other programs.
MAIN	The identity of the primary logic or motion program currently being executed. It is the starting nest level in a call structure and may contain calls to sub programs.
ACTIVE	The identity of the logic or motion program currently executing.
PROGRAM_NEST_LEVEL	An indication of the nesting level within a control program that is associated with the code or instructions that is currently being executed. If an initial value is not defined, the nesting level associated with the highest or initial nesting level of the program **MUST** default to zero (0). The value reported for PROGRAM_NEST_LEVEL **MUST** be an integer.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
ROTARY_MODE	The current operating mode for a Rotary type axis. The *Valid Data Value* **MUST** be SPINDLE, INDEX, or CONTOUR.
ROTARY_VELOCITY_OVERRIDE	The value of 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_ASSET_ID	The identifier of an individual tool asset.The *Valid Data Value* **MUST** be a text string.
TOOL_GROUP	An identifier for the tool group associated with a specific tool. Commonly used to designate spare tools.
~~TOOL_ID~~	**DEPRECATED** in Version 1.2.0. See TOOL_ASSET_ID. ~~The identifier of the tool currently in use for a given Path.~~

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
TOOL_NUMBER	The identifier assigned by the Controller component to a cutting tool when in use by a piece of equipment. The *Valid Data Value* **MUST** be a text string.
TOOL_OFFSET	A reference to the tool offset variables applied to the active cutting tool. 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. **DEPRECATED** in V1.5 ~~A subType **MUST** always be specified.~~
LENGTH	A reference to a length type tool offset.
RADIAL	A reference to a radial type tool offset.
USER	The identifier of the person currently responsible for operating the piece of equipment. A subType **MUST** always be specified.
MAINTENANCE	The identifier of the person currently responsible for performing maintenance on the piece of equipment.
OPERATOR	The identifier of the person currently responsible for operating 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.
VARIABLE	A data value whose meaning may change over time due to changes in the operation of a piece of equipment or the process being executed on that piece of equipment.

Continuation of Table 42: DataItem type subType for category EVENT	
DataItem type subType	Description
WAIT_STATE	An indication of the reason that EXECUTION is reporting a value of WAIT. The *Valid Data Value* **MUST** be POWERING_UP, POWERING_DOWN, PART_LOAD, PART_UNLOAD, TOOL_LOAD, TOOL_UNLOAD, MATERIAL_LOAD, MATERIAL_UNLOAD, SECONDARY_PROCESS, PAUSING, or RESUMING.
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 current workholding or part clamp in use by a piece of equipment. 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.

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*. *Table 43* 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.

Table 43: DataItem type for category CONDITION

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*.

Continuation of Table 43	
DataItem type	Description
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.
MOTION_PROGRAM	An indication that an error occurred in the motion program associated with a piece of equipment.
SYSTEM	An indication of a fault associated with a piece of equipment or component that cannot be classified as a specific type.

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.

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 - Listing of Data Items* 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 shown in *Example 8*:

Example 8: Example of Sensing Element provided as data item associated with a Component

```
<Components>
    <Axes
        <Components>
            <Rotary id="c" name="C">
                <DataItems>
                    <DataItem type="TEMPERATURE"
                        id="ctemp" category="SAMPLE"
                        name="Stemp" units="DEGREE"/>
                </DataItems>
            </Rotary>
        </Components>
    </Axes>
```

```
13 </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 23* :

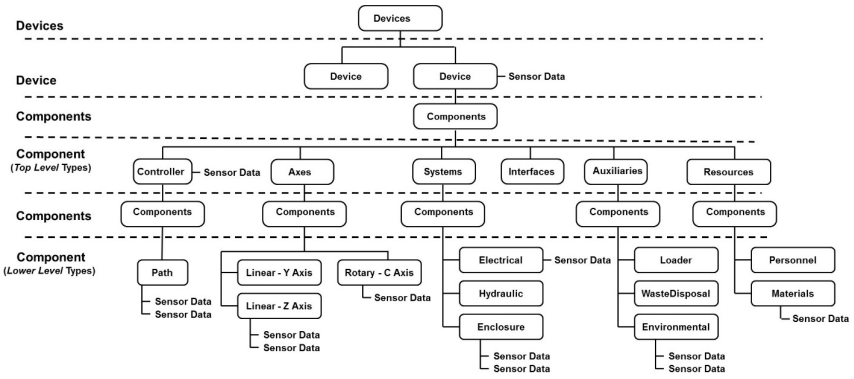

Figure 23: Sensor Data Associations

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 *Top 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.

Example 9: Example of Sensor for rotary axis

```
<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*.

Example 10: Example of Sensor Unit with Sensing Element

```
<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*.

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 *Figure 24*) **MAY** be added to extend the schema for `SensorConfiguration`.

Figure 24 represents the structure of the `SensorConfiguration` XML element showing the attributes defined for `SensorConfiguration`.

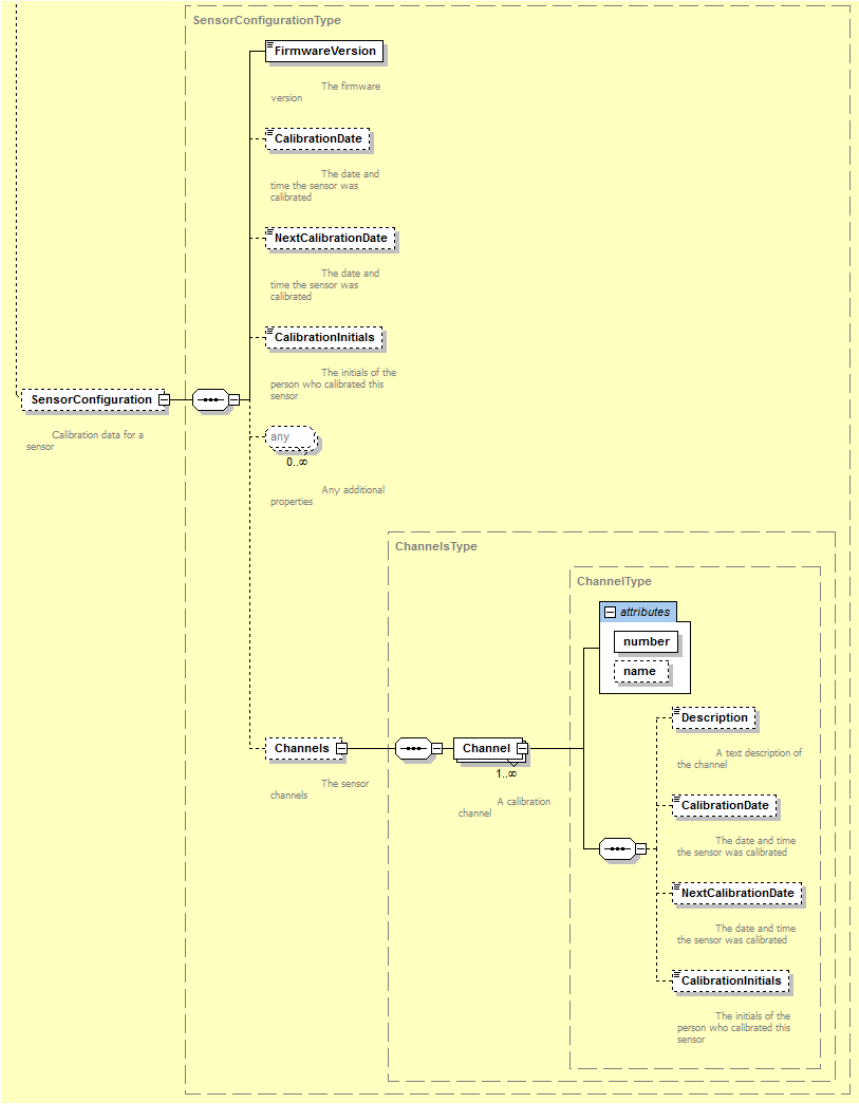

Figure 24: SensorConfiguration Diagram

Table 44: MTConnect SensorConfiguration Element

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

1443 9.3.1 Elements for SensorConfiguration

1444 *Table 45* defines the configuration elements available for `SensorConfiguration`:

Table 45: Elements for SensorConfiguration

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

Continuation of Table 45		
Element	Description	Occurrence
`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
`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

9.3.1.1 Attributes for Channel

`Channel` represents each *sensing element* connected to a *sensor unit*. *Table 46* defines the attributes for `Channel`:

Table 46: 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

1448 #### 9.3.1.2 Elements for Channel

1449 *Table 47* describes the elements provided for `Channel`.

Table 47: Elements 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

Continuation of Table 47		
Element	Description	Occurrence
`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

Example 11 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).

Example 11: Example of configuration data for Sensor

```
<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

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.

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.

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.

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.

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.

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.

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

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.

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.

H.L. Horton, F.D. Jones, and E. Oberg. *Machinery's Handbook*. Industrial Press, Inc.

New York, 1984.

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

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

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

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

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.*

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.