MTConnect® Standard

Part 2 – Components and Data Items

Version 1.2.0 – Final

Prepared for: MTConnect Institute
Prepared by: John Turner
Prepared on: February 17, 2012

MTConnect® Specification and Materials

Table of Contents

Table of Figures

1 Overview

MTConnect® is a standard based on an open protocol for data integration. MTConnect® is not intended to replace the functionality of existing products, but it strives to enhance the data acquisition capabilities of devices and applications and move toward a plug-and-play environment to reduce the cost of integration.

MTConnect® is built upon the most prevalent standards in the manufacturing and software industries, maximizing the number of tools available for its implementation and providing the highest level of interoperability with other standards and tools in these industries.

To facilitate this level of interoperability, a number of objectives are being met. Foremost is the ability to transfer data via a standard protocol which includes:

- A device identity (i.e. model number, serial number, calibration data, etc.).
- The identity of all the independent components of the device.
- Possibly a device's design characteristics (i.e. axis length, maximum speeds, device thresholds, etc.).
- Most importantly, data captured in real or near-real-time (i.e. current speed, position data, temperature data, program block, etc.) by a device that can be utilized by other devices or applications (e.g. utilized by maintenance diagnostic systems, management production information systems, CAM products, etc.).

The types of data that may need to be addressed in MTConnect® could include:

- Physical and actual device design data
- Measurement or calibration data
- Near-real-time data from the device

To accommodate the vast amount of different types of devices and information that may come into play, MTConnect® will provide a common high-level vocabulary and structure.

The first version of MTConnect® focused on a limited set of the characteristics that were selected based on the fact that they could have an immediate effect on the efficiency of operations. Subsequent versions of the standard have and will continue to add additional functionality to more completely define the manufacturing environment.

1.1 MTConnect® Document Structure

The MTConnect® specification is subdivided using the following scheme:

Part 1: Overview and Protocol

Part 2: Components and Data Items

Part 3: Streams, Events, Samples, and Condition

Part 4: Assets

These four documents are considered the basis of the MTConnect Standard. Information applicable to basic machine and device types will be included in these documents. Additional parts to the standard will be added to provide information and extensions to the standard focused on specific devices, components, or technologies considered requiring separate emphasis. All information specific to the topic of each additional part **MUST** be included within that document even when it is subject matter of one of the base parts of the standard.

Documents will be named (file name convention) as follows:

MTC_Part_<Number>_<Description>.doc.

For example, the file name for Part 2 of the standard is MTC_Part_2_Components.doc.

All documents will be developed in Microsoft® Word format and released in Adobe® PDF format.

2 Purpose of This Document

The four base MTConnect® documents are intended to:

- define the MTConnect® standard;
- specify the requirements for compliance with the MTConnect® standard;
- provide engineers with sufficient information to implement *Agents* for their devices;
- provide developers with the necessary guidelines to use the standard to develop applications.

Part 1 of the MTConnect Standard provides an overview of the MTConnect Architecture and the Protocol; including communications, fault tolerance, connectivity, and error handling requirements.

Part 2 of the MTConnect® standard focuses on the data model and description of the information that is available from the device. The descriptive data defines how a piece of equipment should be modeled, the structure of the component hierarchy, the names for each component (if restricted), and allowable data items for each of the components.

Part 3 of the MTConnect standard focuses on the data returned from a `current` or `sample` request (for more information on these requests, see Part 1). This section covers the data representing the state of the machine.

Part 4 of the MTConnect® standard provides a semantic model for entities that are used in the manufacturing process, but are not considered to be a device nor a component. These entities are defined as MTConnect® Assets. These assets may be removed from a device without detriment to the function of the device, and can be associated with other devices during their lifecycle. The data associated with these assets will be retrieved from multiple sources that are responsible for providing their knowledge of the asset. The first type of asset to be addressed is Tooling.

2.1 Terminology

Adapter An optional software component that connects the Agent to the Device.

Agent A process that implements the MTConnect® HTTP protocol, XML generation, and MTConnect protocol.

Alarm An alarm indicates an event that requires attention and indicates a deviation from normal operation. Alarms are reported in MTConnect as `Condition`.

Application A process or set of processes that access the MTConnect® *Agent* to perform some task.

Attribute A part of an XML element that provides additional information about that XML element. For example, the `name` XML element of the `Device` is given as <Device name="mill-1">...</Device>

CDATA The text in a simple content element. For example, *This is some text*, in <Message ...>This is some text</Message>.

Component A part of a device that can have sub-components and data items. A component is a basic building block of a device.

Controlled Vocabulary The value of an element or attribute is limited to a restricted set of possibilities. Examples of controlled vocabularies are country codes: US, JP, CA, FR, DE, etc…

Current A snapshot request to the *Agent* to retrieve the current values of all the data items specified in the path parameter. If no path parameter is given, then the values for all components are provided.

Data Item A data item provides the descriptive information regarding something that can be collected by the *Agent*.

Device A piece of equipment capable of performing an operation. A device may be composed of a set of components that provide data to the application. The device is a separate entity with at least one component or data item providing information about the device.

Discovery Discovery is a service that allows the application to locate *Agents* for devices in the manufacturing environment. The discovery service is also referred to as the *Name Service.*

Event An event represents a change in state that occurs at a point in time. Note: An event does not occur at predefined frequencies.

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

Instance When used in software engineering, the word *instance* is used to define a single physical example of that type. In object-oriented models, there is the class that describes the thing and the instance that is an example of that thing.

LDAP Lightweight Directory Access Protocol, better known as Active Directory in Microsoft Windows. This protocol provides resource location and contact information in a hierarchal structure.

MIME Multipurpose Internet Mail Extensions. A format used for encoding multipart mail and http content with separate sections separated by a fixed boundary.

Probe A request to determine the configuration and reporting capabilities of the device.

REST REpresentational State Transfer. A software architecture where the client and server move through a series of state transitions based solely on the request from the client and the response from the server.

Results A general term for the `Samples`, `Events`, and `Condition` contained in a `ComponentStream` as a response from a `sample` or `current` request.

Sample A sample is a data point from within a continuous series of data points. An example of a Sample is the position of an axis.

Socket When used concerning inter-process communication, it refers to a connection between two end-points (usually processes). Socket communication most often uses TCP/IP as the underlying protocol.

Stream A collection of `Events, Samples,` and `Condition` organized by devices and components.

Service An application that provides necessary functionality.

Tag Used to reference an instance of an XML element.

TCP/IP TCP/IP is the most prevalent stream-based protocol for inter-process communication. It is based on the IP stack (Internet Protocol) and provides the flow-control and reliable transmission layer on top of the IP routing infrastructure.

URI Universal Resource Identifier. This is the official name for a web address as seen in the address bar of a browser.

UUID Universally unique identifier.

XPath XPath is a language for addressing parts of an XML Document. See the XPath specification for more information. http://www.w3.org/TR/xpath

XML Extensible Markup Language. http://www.w3.org/XML/

XML Schema The definition of the XML structure and vocabularies used in the XML Document.

XML Document An instance of an XML Schema which has a single root XML element and conforms to the XML specification and schema.

XML Element An element is the central building block of any XML Document. For example, in MTConnect® the Device XML element is specified as **`<Device >...</Device>`**

XML `NMTOKEN` The data type for XML identifiers. It **MUST** start with a letter, an underscore "_" or a colon ":" and then it **MUST** be followed by a letter, a number, or one of the following ".", "-", "_", ":". An `NMTOKEN` cannot have any spaces or special characters.

2.2 Terminology and Conventions

Please refer to *Section 2 of Part 1 "Overview and Protocol"* for XML Terminology and Documentation conventions.

3 Devices and Components

MTConnect organizes information and data from a data source (typically a machine) into an information model that defines the relationship between each piece of data and the source of that data. This information model allows an application to interpret the data received from a data source and correlate that data to its original definition, value, and context.

The basic MTConnect information model contains three primary containers: `Devices`, `Device`, and `Components`. These containers are the building blocks used to organize information about a piece of equipment. They also define how the various parts of a piece of equipment relate to each other.

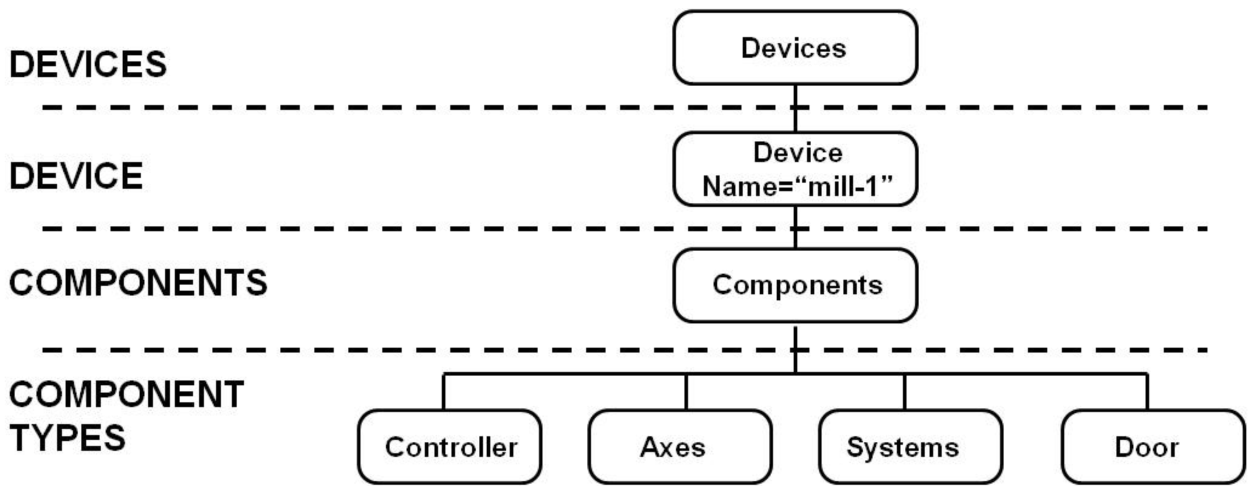

Figure 1: Example Container Structure

The first, or highest, level container in the MTConnect data structure is the `Devices` container. The `Devices` container is comprised of one or more `Device` XML Element(s). The `Devices` container provides a mechanism for grouping data from multiple `Device` elements that are providing their data through a common MTConnect *`Agent`*. `Devices` has no attributes and is only used to group data from `Device` elements together.

The next level container is `Device`. A `Device` typically represents a single piece of equipment or a machine. However, it can also represent any logical grouping of components that operate together to perform a function. Every `Device` in MTConnect® **MUST** have an `Availability` data item. `Availability` represents the device's ability to provide information about itself. The `Device` container is compromised of one or more `Components` XML Elements.

The third container in the MTConnect Data Structure is the `Components` container(s). `Components` provides a mechanism for grouping sub-elements of a `Device` into logical groups that are associated with each other. `Components` has no attributes and is only used to group `Component` elements together. The `Components` container is compromised of one or more `Component` XML Elements.

3.1 Devices

The Devices XML Element is the top level container for every Device. It may contain multiple Device elements. Devices may only contain elements of type Device.

Elements	Description	Occurrence
Device	The root of each device. Device is contained within the top level Devices container. There can be multiple Device elements.	1..INF

3.2 Device

A Device is a XML Element that holds all the Components associated with a piece of equipment. This can be a logical grouping of Component XML Elements that perform a particular function. The Device **MUST** have an Availability data item that indicates if this device is available to provide information.

In the MTConnect® schema, a Device is actually a unique type of Component (defined below). A Device supports all of the functions and capabilities defined for a Component. However, it MUST be uniquely identified throughout the MTConnect® Standard and schema as a Device to clearly define the difference between a logical collection of components that function together as a Device and the identification of each Component that forms the structure within a Device.

Note: Some components may not be integral to a parent device or another component. These components may function independently or produce data that is not relevant to a parent device. An example would be a temperature sensor installed in a plant to monitor the ambient air temperature. In this case, the Component **MAY** be modeled in the MTConnect schema as a Device. When modeled as a Device, the component **MUST** provide all of the data and capabilities defined for a Device. It is also possible for these components to be defined as a Component of a parent device and simultaneously as an independent Device; communicating data associated with the parent Device incorporated into that device's data stream and independently communicating additional data in a separate data stream using its own uuid.

3.2.1.1 Device Structure

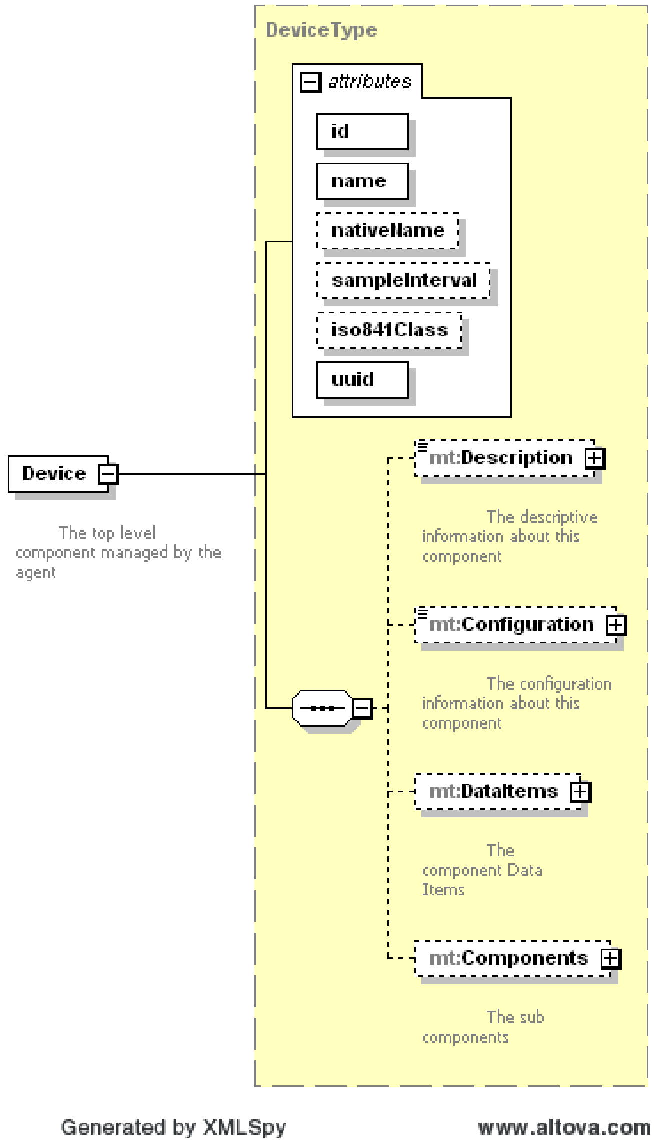

Figure 2: Device Schema Diagram

3.2.1.2 Device Attributes

Attribute	Description	Occurrence
iso841Class	**DEPRECATED in Release 1.1.0**	
uuid	A unique identifier that will only refer to this Device. For example, this may be the manufacturer's code and the serial number. The uuid should be alphanumeric and not exceeding 255 characters. An NMTOKEN XML type.	0..1*
name	The name of the Device. This name should be unique within the machine to allow for easier data integration. An NMTOKEN XML type.	1
nativeName	The name the device manufacturer assigned to this Device. If the native name is not provided, it **MUST** be the name.	0..1
id	The unique identifier for this Device in the document. An id **MUST** be unique across all the id attributes in the document. An XML ID-type.	1
sampleRate	**DEPRECATED IN REL. 1.2 (REPLACED BY sampleInterval**	
sampleInterval	The interval in milliseconds between the completion of the reading of one sample of data from a device until the beginning of the next sampling of that data. This is the number of milliseconds between data captures. If the sample interval is smaller than one millisecond, the number can be represented as a floating point number. For example, an interval of 100 microseconds would be 0.1.	0..1**

Notes: * The uuid **MUST** be provided for the Device. It is optional for all other Component types.

** The sampleInterval is used to aid an application in interpolating values. This is the desired sample interval and may vary depending on the capabilities of the device.

3.2.1.3 Device Elements

Element	Description	Occurrence
Description	An XML element that can contain any descriptive content. This can contain configuration information and manufacturer specific details.	0..1
Configuration	An XML element that can contain descriptive content defining the configuration information for a Device.	0..1
Components	A container for lower level Component XML Elements associated with this Device.	0..INF*
DataItems	The data items (defined below) provided by this Device. The data items define the measured values to be reported by this Device.	0..INF*

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

3.3 Components

Components is a container that provides structure for sub-elements of a device. Components contains one or more Component XML Elements.

Elements	Description	Occurrence
Component	Types of Component XML Elements. There can be multiple Component XML Elements.	1..INF

3.4 Component Types

A Component XML Element defines physical or logical sub-element of a device. Component is an abstract type and will never appear in the MTConnect XML document. Component elements are represented as XML Element sub-types such as Axes, Controller, Door, etc.

Component elements contain information and data defining the element's operational state, the environment in which it is functioning, and its health or status. This information and the measured values associated with a component are defined as DataItems and will be discussed in *Section 3.5* of this document.

Component can be further sub-divided into smaller Components XML Elements to provide additional detail on the structure and configuration of a Component. These sub-elements have all the characteristics and capabilities of the parent component.

While these sub-elements are by definition Components, they **SHALL** be called *subcomponents* within the MTConnect Standard to provide clarity on the relationship between the parent component and its associated sub-elements (*subcomponents)*. Additionally, *subcomponents* may be further subdivided into additional Components, as required, to provide a complete description of a device and its measured values (DataItems).

Components and related *subcomponents* are represented in the XML schema as follows:

```
<Devices>
 <Device>
  <Components>
   <Axes(Component Type Subcomponent)>
    <Components>
     <Linear (Component Type Subcomponent) >
      < Components>
        <Etc.  >
```

Figure 3 below describes the relationship between `Component` and *subcomponents*.

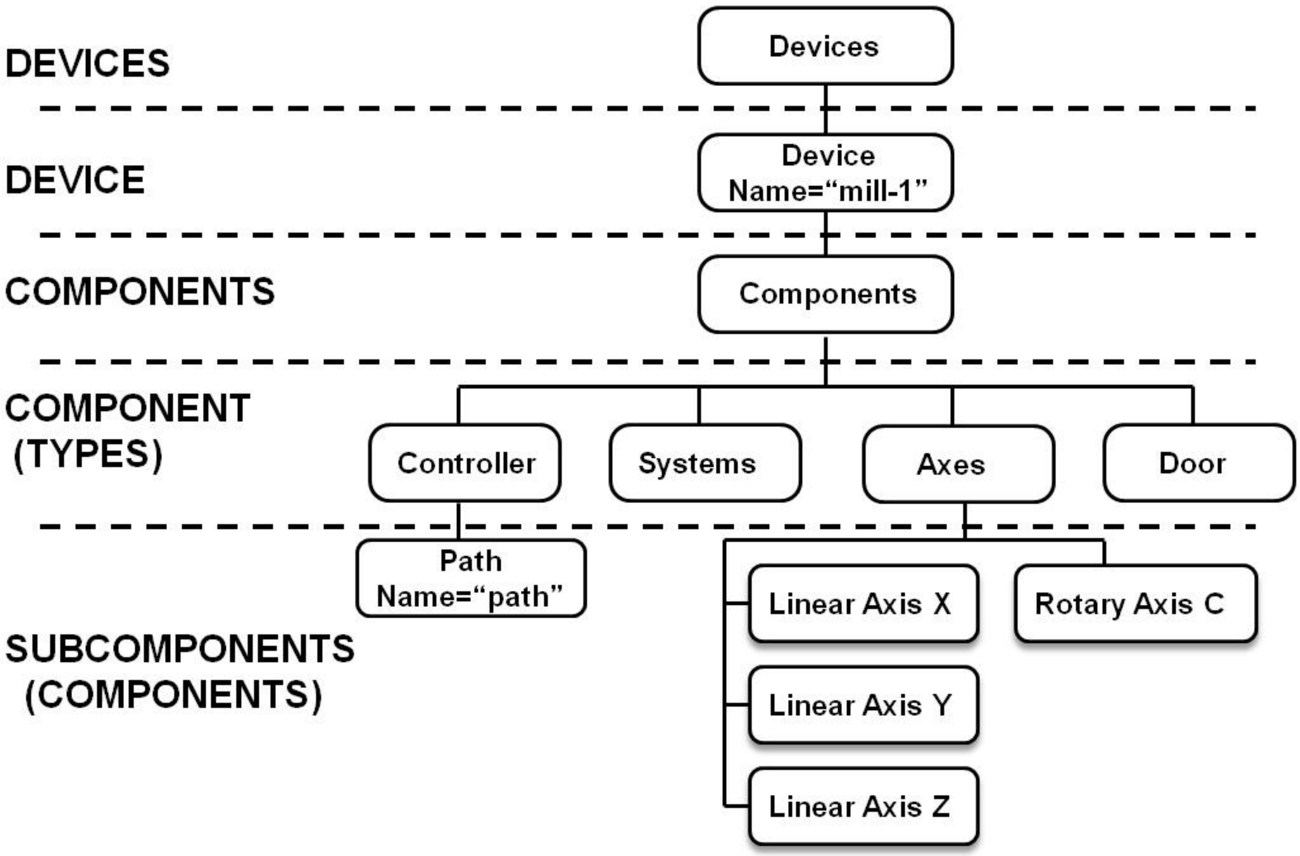

Figure 3: Component/Subcomponent Diagram

3.4.1 Component Schema

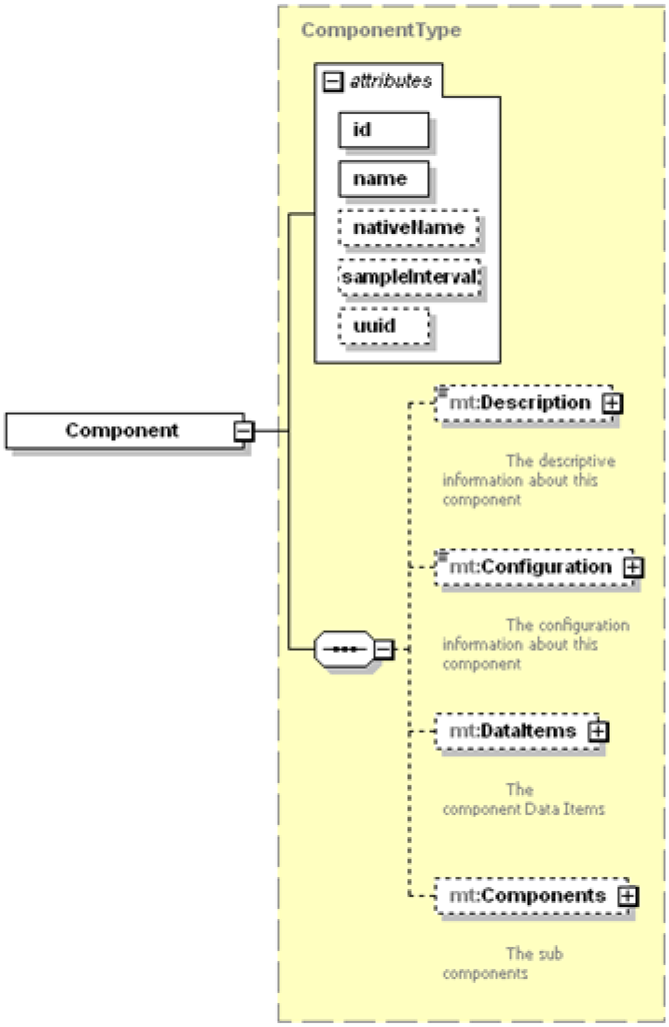

Figure 4: `Component` Schema

3.4.2 `Component` Attributes

Every component has the following composition:

Attribute	Description	Occurrence
`uuid`	A unique identifier that will only refer to this `Component`. For example, this can be the manufacturer's code and the serial number. The uuid should be alphanumeric and not exceeding 255 characters. An NMTOKEN XML type.	0..1*
`name`	The name of the `Component`. This name should be unique within the machine to allow for easier data integration. An NMTOKEN XML type.	1
`nativeName`	The name the device manufacturer assigned to the `Component`. If the native name is not provided it **MUST** be the `name`.	0..1

Attribute	Description	Occurrence
id	The unique identifier for this Component in the document. An id **MUST** be unique across all the id attributes in the document. An XML ID-type.	1
sampleRate	**DEPRECATED IN REL. 1.2 (REPLACED BY sampleInterval)**	
sampleInterval	The interval in milliseconds between the completion of the reading of one sample of data from a component until the beginning of the next sampling of that data. This is the number of milliseconds between data captures. If the sample interval is smaller than one millisecond, the number can be represented as a floating point number. For example, an interval of 100 microseconds would be 0.1.	0..1**

Notes: * The uuid **MUST** be provided for the Device. It is optional for all other Component types.
** The sampleInterval is used to aid the application in interpolating values. This is the desired sample Interval and may vary depending on the capabilities of the component.

3.4.3 Component Elements

Element	Description	Occurrence
Description	An element that can contain any descriptive content. This can contain information about the Component and manufacturer specific details.	0..1
Components	A container for lower level Component XML Elements associated with this Component.	0..INF*
Configuration	An element that can contain descriptive content defining the configuration information for a Component.	0..1
DataItems	The data items this component provides. The data items define the measured values to be reported by this Component.	0..INF*

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

3.4.3.1 Component Description

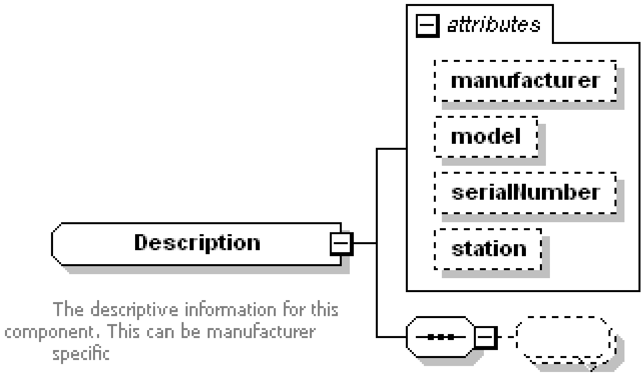

Figure 5: Component Schema

Attribute	Description	Occurrence
manufacturer	The name of the manufacturer of the Component	0..1
model	The model description of the Component	0..1
serialNumber	The component's serial number	0..1
station	The station where the Component is located when a component is part of a manufacturing unit or cell with multiple stations that share the same physical controller.	0..1

The CDATA of Description is any additional descriptive information the implementer chooses to include regarding the Component. An example of a Description is as follows:

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

The information can be provided for any component. For example, an electrical power sensor can be defined as follows:

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

3.4.3.2 Component `Components`

Element	Description	Occurrence
`Components`	One or more subcomponents. This can also include the subtypes of `Component` like `Axes`, `Linear`, `Path`, etc...	1..INF

3.4.3.3 Component `DataItems`

Element	Description	Occurrence
`DataItems`	Only XML elements of types `DataItem` can be specified	1..INF

3.4.3.4 Component `Configuration`

Element	Description	Occurrence
`Configuration`	An XML element that can contain descriptive content defining the configuration information for a `Component`. Not all `Component` *types* support `Configuration`. When `Configuration` is supported, details on the schema for `Configuration` will be included in the applicable sections of the MTConnect standard.	1..INF

Configuration data is structured in the MTConnect schema as shown below:

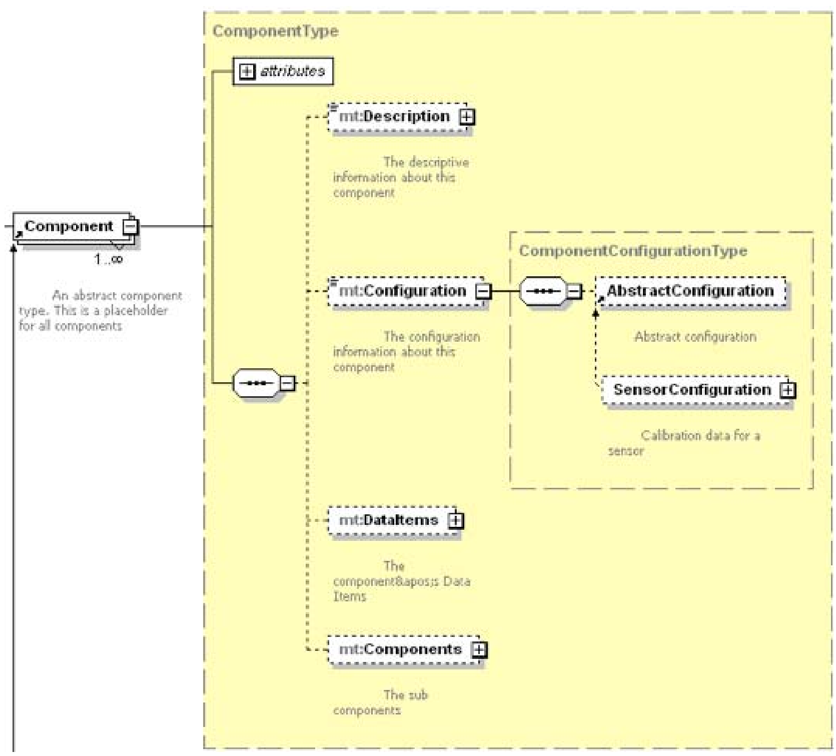

Figure 5: Component Configuration Schema

3.5 DataItem

DataItem is a piece of information that can be collected from a Device, Component, or *subcomponent*. A DataItem **MAY** report both a numeric value (a numeric quantity reported as either a Sample or Event category) and a health status (reported as a Condition category). A DataItem specifies the type of data being collected and an array of optional attributes that further defines that data. The value of the data is provided in the Streams response.

The *Agent* transmits data items associated with each Component to an application. The actual values for those data items are delivered in Streams and will be discussed in detail in the MTConnect Standard *Part 3: Streams, Samples, and Events*.

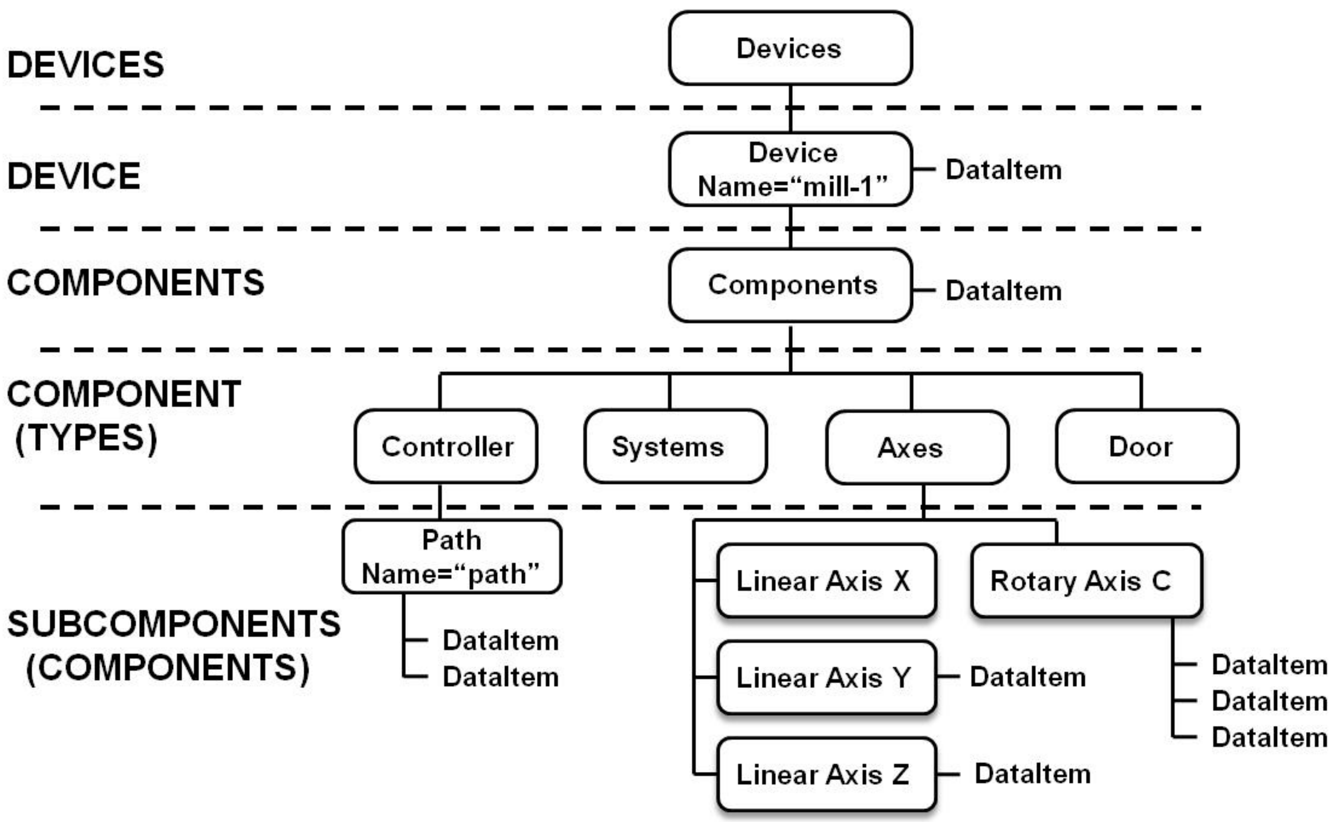

Figure 6: Example DataItem Structure

3.5.1 DataItem Schema

A `DataItem` **MUST** specify the type of data being collected, the `id` of the `DataItem`, and the `category` of the item. Since many data item types provide both a value (reported as either a `Sample` or `Event` category) and a health status (reported as a `Condition` category), each `DataItem` **MUST** report a category for each data type to aid the application in determining the specific meaning of the data. The `DataItem` **MAY** specify a `Source` sub-element to identify `where` the physical connection to the data source originates; ex. data relative to a servo motor may actually originate from a measurement made in the controller.

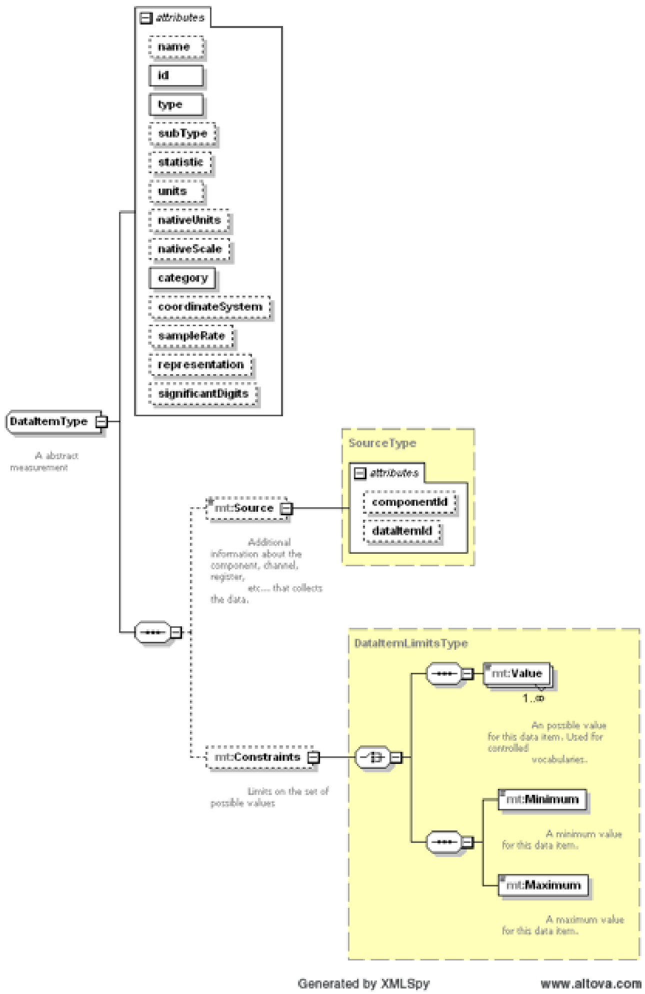

Figure 7: `DataItem` Schema Diagram

A `DataItem` **MAY** also specify a *subtype* to further qualify the type of data being provided. *Subtypes* are required for certain data items. For example, the `Position` has two *subtypes*: `ACTUAL` and `COMMANDED`. These are two separate `DataItem(s)` that can be reported independently. See the sections below addressing `Sample`, `Event`, and `Condition` for a complete list of *type/subtype* relations.

For information on the transformation between DataItem name as returned in a Probe request and the corresponding data returned in a Stream element, see the MTConnect Standard *Part 3, Section 3.5.*

3.5.2 DataItem Attributes

Attribute	Description	Occurrence
id	The unique identifier for this DataItem. The id attribute **MUST** be unique across the entire document including the ids for components. An XML ID-type.	1
name	The name of the DataItem. A name is provided as an additional human readable identifier for this DataItem in addition to the id. It is not required and will be implementation dependent. An NMTOKEN XML type.	0..1
type	The type of data being measured. Examples of types are POSITION, VELOCITY, ANGLE, BLOCK, ROTARY_VELOCITY, etc.	1
subType	A sub-categorization of the data item type. For example, the subtypes of POSITION are ACTUAL and COMMANDED. Not all types have subtypes and this can be left off.	0..1
category	This is how the meaning of the data item will be determined. The available options are SAMPLE, EVENT, or CONDITION.	1
statistic	Data calculated specific to a DataItem. Examples of statistic are AVERAGE, MINIMUM, MAXIMUM, ROOT_MEAN_SQUARE, RANGE, MEDIAN, MODE, AND STANDARD_DEVIATION.	0..1
representation	Data consisting of multiple data points or samples or a file presented as a single DataItem. Each representation will have a unique format defined for each representation. Examples of representation are VALUE, TIME_SERIES, MP3, WAV, etc. Initially, the representation for TIME_SERIES and VALUE are defined. If a representation is not specified, it **MUST** be determined to be VALUE.	0..1
nativeUnits	The native units used by the Component. These units will be converted before they are delivered to the application.	0..1
units	Units **MUST** be present for all samples. If the data represented by a DataItem is a numeric value, except for line number and count, the units **MUST** be specified.	0..1
nativeScale	The multiplier for the native units. The received data **MAY** be divided by this value before conversion. If provided, the value **MUST** be numeric.	0..1

Attribute	Description	Occurrence
significantDigits	The number of significant digits in the reported value. This is used by applications to determine accuracy of values. This **SHOULD** be specified for all numeric values.	0..1
sampleRate	The rate at which successive samples of a DataItem are recorded. 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	0..1**
coordinateSystem	The coordinate system being used. The available values for coordinateSystem is WORK and MACHINE	0..1

3.5.3 Data Item Elements

Element	Description	Occurrence
Source	Source is an optional XML element that identifies the Component, subcomponent, or DataItem where the physical connection to the data source originates. The CDATA of the Source element **MAY** also contain the long name of the data item if it is too complex for the name attribute. For example, if we want to name the data item for X axis actual position "Xact", but the X axis position is delivered from the device as Channel.0.position, Source is used to provide the necessary mapping. If the source is not specified, it will be assumed to be the same as the name.	0..1
Constraints	The set of possible values that can be assigned to this DataItem. Constraints provide a way to specify the capabilities for this Component by limiting the choices for the value that is reported in the *Streams* response. For example, for ROTARY_MODE the axis can be limited to SPINDLE for an axis that can only spin.	0..1

3.5.3.1 Source Attributes

Source identifies the physical device or data source where the data represented by the DataItem is generated:

Attribute	Description	Occurrence
componentID	A unique identifier that references a specific Component from which the data represented by the DataItem originates. This **MUST** be the unique identifier defined for the component in its id attribute and **MUST** occur elsewhere in the XML document. It is an XML xs:IDREF type.	0..1

Attribute	Description	Occurrence
`dataItemID`	A unique identifier that references a specific `DataItem` from which the data represented by this `DataItem` is generated. This **MUST** be the unique identifier defined for the `DataItem` `id` attribute and **MUST** occur elsewhere in the XML document. It is an `XML xs:IDREF` type.	0..1

3.5.3.2 Constraints Elements

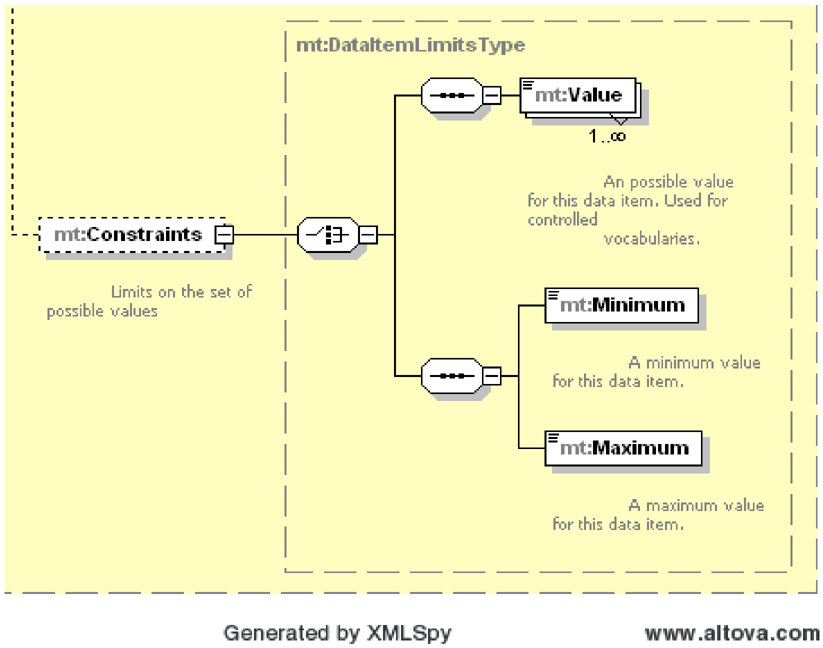

Figure 8: Constraints Schema

Element	Description	Occurrence
`Value`	A constraint on the possible values for this data item. If there is only one value listed here, the `DataItem` value will be constant. In the case of a constant `DataItem` value, the value is not required to be supplied in the streams document.	0..INF
`Maximum`	The maximum value for this `DataItem`. This will be the bounded upper range. This will only be relevant when the `DataItem` has a numeric type.	0..1
`Minimum`	The minimum value for this `DataItem`. This will be the bounded lower range. This will only be relevant when the `DataItem` has a numeric type.	0..1

3.5.4 Data Item attribute: `category`

MTConnect® provides three different categories of `DataItem - SAMPLE`, `EVENT`, and `CONDITION`. The `category` will indicate where the results will be reported in the XML

Document as a response to a `Sample` or `Current` request. See *Part 3 Section 3* on *Streams, Samples, and Events* for more information.

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

A `DataItem` of the category `Sample` that are continuous are always scalar floating point or integers that can have an infinite number of possible values. This is different from state or discrete type `DataItem` that has a limited number of possible values. A `DataItem` of category `Sample` **MUST** have units.

EVENT A `DataItem` of the category `Event` comprises discrete information from the device. 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 a `DoorStatus` that can be either `OPEN`, `UNLATCHED`, or `CLOSED`. An example of a message type `Event` is a `PROGRAM` that can be any valid string of characters. A `DataItem` of category `Event` does not have intermediate values that vary over time, as do `Samples`. An `Event` can be thought of as streaming information that if taken at any point in time represents the current state of the device.

CONDITION A `DataItem` that communicates the device's health and ability to function. A `DataItem` of category `Condition` can be one of `UNAVAILABLE`, `NORMAL`, `WARNING`, or `FAULT`. A `DataItem` of category `Condition` **MAY** report multiple active conditions at one time; whereas a `DataItem` of category `Sample` or `Event` can only have a single value at any one point in time.

3.5.5 Data Item attribute: `coordinateSystem`

A `DataItem` can specify an optional coordinate system that is being used. If not specified, the `Axes` coordinates **MUST** be `MACHINE` and the `Path` coordinates **MUST** be `WORK`. The possible values of coordinates are:

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 device, the `MACHINE` coordinates will be used.

3.5.6 Data Item attribute: `units`

Units	Description
AMPERE	Amps
CELSIUS	Degrees Celsius
COUNT	A counted event
DECIBEL	Sound Level
DEGREE	Angle in degrees
DEGREE/SECOND	Angular degrees per second
DEGREE/SECOND^2	Angular acceleration in degrees per second squared
HERTZ	Frequency measured in cycles per second
JOULE	A measurement of energy.
KILOGRAM	Kilograms
LITER	Liters
LITER/SECOND	Liters per second
MICRO_RADIAN	Measurement of Tilt
MILLIMETER	Millimeters
MILLIMETER/SECOND	Millimeters per second
MILLIMETER/SECOND^2	Acceleration in millimeters per second squared
MILLIMETER_3D	A point in space identified by X, Y, and Z positions and represented by a space delimited set of numbers each expressed in millimeters.
NEWTON	Force in Newtons
NEWTON_METER	Torque, a unit for force times distance.
OHM	Measure of Electrical Resistance
PASCAL	Pressure in Newtons per square meter
PASCAL_SECOND	Measurement of Viscosity
PERCENT	Percentage
PH	A measure of the acidity or alkalinity of a solution
REVOLUTION/MINUTE	Revolutions per minute
SECOND	A measurement of time.
SIEMENS/METER	A measurement of Electrical Conductivity
VOLT	Volts

Units	Description
VOLT_AMPERE	Volt-Ampere (VA)
VOLT_AMPERE_REACTIVE	Volt-Ampere Reactive (var)
WATT	Watts
WATT_SECOND	Measurement of electrical energy, equal to one Joule

`Units` **MUST** be specified for any `DataItem` with category `Sample`. The `nativeUnits` **MAY** also be specified if they apply to the type of data and if they differ from the `units`. The *Agent* is responsible for converting the `nativeUnits` to the `units` before sending them to the applications. In addition, `nativeUnits` **MAY** be scaled using the `nativeScale` attribute; for example, if the device measures velocity in 100 ft/min, MTConnect® would represent it with the following attributes: `nativeUnits="FEET/MINUTE"` and `nativeScale="100"`.

3.5.7 Data Item attribute: `statistic`

The `statistic` attribute indicates that the data has been processed using a statistical operation like average, mean, or root square. `statistic` may be reported for any `Sample` type `DataItem`. These values are calculated values generated by the `Component` or `Device` providing additional data regarding a `DataItem` sampled over a specified period of time. All `statistic` data is reported in the standard units of the `DataItem`.

The value of `statistic` is periodically reset. When `statistic` values are reported as a *Streams* value, the value of the `statistic` **MUST** include an attribute `Duration`. `Duration` defines the time elapsed since the statistic calculation was last reset.

Statistic	Description
AVERAGE	Mathematical Average value calculated for the `DataItem` 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 `DataItem` during the calculation period
MEDIAN	The middle number of a series of numbers
MINIMUM	Minimum value recorded for the `DataItem` 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 `DataItem` during the calculation period. Also represents Peak-to-Peak measurement in a waveform.

Statistic	Description
ROOT_MEAN_SQUARE	Mathematical Root Mean Value (RMS) value calculated for the DataItem during the calculation period
STANDARD_DEVIATION	Statistical Standard Deviation value calculated for the DataItem during the calculation period

3.5.8 Data Item attribute: representation

The representation attribute defines the format for data consisting of multiple data points or a file presented as a single DataItem. Each representation will have a unique format defined for each representation. At this time, the only representations defined are TIME_SERIES and VALUE.

Details on the structure and format of each representation is provided in *Part 3, Section 3.8.3* of the MTConnect Standard.

Representation	Description
VALUE	The measured value of a sample. If no representation is specified for a DataItem, the representation **MUST** be determined to be VALUE.
TIME_SERIES	A series of sampled data. The data is collected for a specified number of samples and each sample is collected with a fixed period

3.5.9 Data Item Attribute: nativeUnits

The nativeUnits attribute adds additional values to the units values. This is the list of nativeUnits currently supported by MTConnect® and the MTConnect® schema.

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.

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

3.5.10 Data Item Types for SAMPLE Category

The types are given in **bold** and the subtypes are indented and in plain text.

Data Item type/subtype	Description	Units
ACCELERATION	Rate of change of velocity	MILLIMETER/SECOND^2
ACCUMULATED_TIME	The measurement of accumulated time associated with a Component	SECOND
ANGULAR_ACCELERATION	Rate of change of angular velocity.	DEGREE/SECOND^2
ANGULAR_VELOCITY	Rate of change of angular position.	DEGREE/SECOND
AMPERAGE	The measurement of AC Current or a DC current	AMPERE
ALTERNATING	The measurement of alternating current. If not specified further in statistic, defaults to RMS Current	AMPERE
DIRECT	The measurement of DC current	AMPERE
ANGLE	The angular position of a component relative to the parent.	DEGREE
ACTUAL	The angular position as read from the physical component.	DEGREE
COMMANDED	The angular position computed by the Controller.	DEGREE
AXIS_FEEDRATE	The feedrate of a linear axis.	MILLIMETER/SECOND
ACTUAL	The actual federate of a linear axis.	MILLIMETER/SECOND
COMMANDED	The feedrate as specified in the program.	MILLIMETER/SECOND
OVERRIDE	The operator's overridden value. Percent of commanded.	PERCENT
CLOCK_TIME	The reading of a timing device at a specific point in time. Clock time **MUST** be reported in W3C ISO 8601 format.	YYYY-MM-DDThh:mm:ss.ffff
CONCENTRATION	Percentage of one component within a mixture of components	PERCENT
CONDUCTIVITY	The ability of a material to conduct electricity	SIEMENS/METER
DISPLACEMENT	The displacement as the change in position of an object	MILLIMETER
ELECTRICAL_ENERGY	The measurement of electrical energy consumption by a component	WATT_SECOND

Data Item 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 Rel. 1.1	
~~**LEVEL**~~	**DEPRECATED** in Rel. 1.2 See FILL_LEVEL	
LINEAR_FORCE	The measure of the push or pull introduced by an actuator or exerted on an object	NEWTON
LOAD	The measurement of the percentage of the standard rating of a device	PERCENT
MASS	The measurement of the mass of an object(s) or an amount of material	KILOGRAM
PATH_FEEDRATE	The feedrate of the tool path.	MILLIMETER/SECOND
ACTUAL	The three-dimensional feedrate derived from the Controller.	MILLIMETER/SECOND
COMMANDED	The feedrate as specified in the program	MILLIMETER/SECOND
OVERRIDE	The operator's overridden value. Percent of commanded.	PERCENT
PATH_POSITION	The current program control point or program coordinate in WORK coordinates. The coordinate system will revert to MACHINE coordinates if WORK coordinates are not available.	MILLIMETER_3D
ACTUAL	The position of the Component as read from the device.	MILLIMETER_3D
COMMANDED	The position computed by the Controller.	MILLIMETER_3D
TARGET	The target position for the movement.	MILLIMETER_3D
PROBE	The position provided by a probe	MILLIMETER_3D
PH	The measure of the acidity or alkalinity.	PH
POSITION	The position of the Component. Defaults to MACHINE coordinates.	MILLIMETER
ACTUAL	The position of the Component.	MILLIMETER
COMMANDED	The position as given by the Controller.	MILLIMETER

Data Item type/subtype	Description	Units
TARGET	The target position for the movement.	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
RESISTANCE	The measurement of the degree to which an object opposes an electric current through it	OHM
ROTARY_VELOCITY	The rotational speed of a rotary axis.	REVOLUTION/MINUTE
ACTUAL	The rotational speed the rotary axis is spinning at. ROTARY_MODE **MUST** be SPINDLE.	REVOLUTION/MINUTE
COMMANDED	The rotational speed as specified in the program.	REVOLUTION/MINUTE
OVERRIDE	The operator's overridden value. Percent of commanded.	PERCENT
SOUND_LEVEL	Measurement of a sound level or sound pressure level relative to atmospheric pressure	DECIBEL
NO_SCALE	No weighting factor on the frequency scale	DECIBEL
A_SCALE	A Scale weighting factor. This is the default weighting factor if no factor is specified	DECIBEL
B_SCALE	B Scale weighting factor	DECIBEL
C_SCALE	C Scale weighting factor	DECIBEL
D_SCALE	D Scale weighting factor	DECIBEL
SPINDLE SPEED	**DEPRECATED in REL 1.2.** Replaced by ROTARY_VELOCITY	
ACTUAL	The rotational speed of a rotary axis. ROTARY_MODE **MUST** be SPINDLE.	REVOLUTION/MINUTE
COMMANDED	The rotational speed the as specified in the program.	REVOLUTION/MINUTE
OVERRIDE	The operator's overridden value. Percent of commanded.	PERCENT
STRAIN	Strain is the amount of deformation per unit length of an object when a load is applied.	PERCENT
TEMPERATURE	The measurement of temperature	CELSIUS
TILT	A measurement of angular displacement	MICRO_RADIAN
TORQUE	The turning force exerted on an object or by an object	NEWTON_METER

Data Item type/subtype	Description	Units
VOLT_AMPERE	The measure of the apparent power in an electrical circuit, equal to the product of root-mean-square (RMS) voltage and RMS current' (commonly referred to as VA)	VOLT_AMPERE
VOLT_AMPERE_REACTIVE	The measurement of reactive power in an AC electrical circuit (commonly referred to as var)	VOLT_AMPERE_REACTIVE
VELOCITY	The rate of change of position.	MILLIMETER/SECOND
VISCOSITY	A measurement of a fluid's resistance to flow	PASCAL_SECOND
VOLTAGE	The measurement of electrical potential between two points	VOLT
ALTERNATING	The measurement of alternating voltage. If not specified further in statistic, defaults to RMS voltage	VOLT
DIRECT	The measurement of DC voltage	VOLT
WATTAGE	The measurement of power consumed or dissipated by an electrical circuit or device	WATT

3.5.11 Data Item Types for EVENT Category

Note: The Event does not have any units since these values are not scalars.

Data Item type/subtype	Description
ACTUATOR_STATE	The state of the Actuator - ACTIVE or INACTIVE.
ALARM	DEPRECATED: Replaced with CONDITION category. *Rel. 1.1.*
ACTIVE_AXES	The set of axes associated with a Path that the Controller is controlling. If this DataItem is not provided, it will be assumed the Controller is controlling all axes.
AVAILABILITY	Represents the ability of a Component to communicate. This **MUST** be provided for a Device and **MAY** be provided for any other Component. AVAILABLE or UNAVAILABLE.
AXIS_COUPLING	Describes the way the axes will be associated to each other. This is used in conjunction with COUPLED_AXES to indicate the way they are interacting. The possible values are: TANDEM, SYNCHRONOUS, MASTER, and SLAVE. The coupling **MUST** be viewed from the perspective of the axis, therefore a MASTER coupling indicates that this axis is the master of the COUPLED_AXES.
BLOCK	The block of code being executed. Block contains the entire expression for a line of program code.
CODE	DEPRECATED. *Rel 1.1.0*

Data Item type/subtype	Description
CONTROLLER_MODE	The current mode of the Controller. AUTOMATIC, MANUAL, MANUAL_DATA_INPUT, or SEMI_AUTOMATIC.
COUPLED_AXES	Refers to the set of associated axes. The value will be a space delimited set of axes names.
DIRECTION	The direction of motion. CLOCKWISE or COUNTER_CLOCKWISE
ROTARY	The rotational direction of a rotary device using the right hand rule convention as defined in *Appendix B*. CLOCKWISE or COUNTER_CLOCKWISE
LINEAR	The direction of motion of a linear device. POSTIVE or NEGATIVE
DOOR_STATE	The opened or closed state of the door. OPEN, UNLATCHED, or CLOSED.
EMERGENCY_STOP	The current state of the emergency stop actuator. ARMED (the circuit is complete and the device is operating) or TRIGGERED (the circuit is open and the device MUST cease operation).
EXECUTION	The execution status of the Controller. READY, ACTIVE, INTERRUPTED, FEED_HOLD, or STOPPED
LINE	The current line of code being executed
MAXIMUM	The maximum line number of the code being executed.
MINIMUM	The minimum line number of the code being executed.
MESSAGE	An uninterpreted textual notification.
PALLET_ID	The identifier for the pallet currently in use for a given Path
PART_COUNT	The current count of parts produced as represented by the Controller. **MUST** be an integer value.
ALL	The count of all the parts produced. If the subtype is not given, this is the default.
GOOD	Indicates the count of correct parts made.
BAD	Indicates the count of incorrect parts produced.
PART_ID	An identifier of the current part in the device
PATH_MODE	The operational mode for this Path. SYNCHRONOUS, MIRROR, or INDEPENDENT. Default value is INDEPENDENT if not specified.
POWER_STATE	The ON or OFF status of the Component. **DEPRECATION WARNING**: **MAY** be deprecated in the future.
LINE	The state of the high voltage line.
CONTROL	The state of the low power line.
POWER_STATUS	DEPRECATED. *Rel. 1.1.*
PROGRAM	The name of the program being executed
ROTARY_MODE	The mode for the Rotary axis. SPINDLE, INDEX, or CONTOUR.
~~**TOOL_ID**~~	**DEPRECATED in *Rel. 1.2*.** See Tool_ASSET_ID. ~~The identifier of the tool currently in use for a given Path~~
TOOL_ASSET_ID	The identifier of an individual tool asset.
TOOL_NUMBER	The identifier of a tool provided by the device controller.
WORKHOLDING_ID	The identifier for the workholding currently in use for a given Path

3.5.12 Data Item Types for CONDITION Category

`Condition` is a `DataItem` that indicates the device's health and ability to operate. They are reported differently than `Samples` or `Events`: they **MUST** be reported as `NORMAL`, `WARNING`, `FAULT`, or `UNAVAILABLE`. Unlike the other two categories, a `Component` or `Device` **MAY** have a `Condition` type `DataItem` that has multiple concurrently active values at any point in time. Additionally, these items **MAY** be further defined to provide differentiation for different condition states; example an AMPERAGE *Condition* may differentiate between HIGH amperage and LOW amperage. These differences are further defined as *qualifier* in *Part 3, Section 3.11* of the MTConnect Standard.

Data Item type/ qualifier	Description
ACCELERATION	Rate of Change of Velocity
ACCUMULATED TIME	The measurement of accumulated time associated with a `Component`
ACTUATOR	An actuator related condition.
AMPERAGE	A high or low condition for the electrical current.
ANGLE	The angular position of a `Component`.
ANGULAR-ACCELERATION	Rate of change of angular velocity.
ANGULAR VELOCITY	Rate of change of angular position
COMMUNICATIONS	A communications failure indicator.
CONCENTRATION	Percentage of one ingredient within a mixture of ingredients
CONDUCTIVITY	The ability of a material to conduct electricity
DATA RANGE	Information provided is outside of expected value range
DIRECTION	The direction of motion of a `Component`
DISPLACEMENT	The change in position of an object
ELECTRICAL_ENERGY	The measurement of electrical energy consumption by a `Component`
FILL_LEVEL	Represents the amount of a substance remaining compared to the planned maximum amount of that substance
FLOW	The rate of flow of a fluid
FREQUENCY	The number of occurrences of a repeating event per unit time
HARDWARE	The hardware subsystem of the `Component's` operation condition.
~~**LEVEL**~~	**DEPRECATED in *Rel 1.2*. See FILL_LEVEL**
LINEAR FORCE	The measure of the push or pull introduced by an actuator or exerted by an object
LOAD	The measure of the percentage of the standard rating of a device
LOGIC PROGRAM	An error occurred in the logic program or PLC (programmable logic controller).
MASS	The measurement of the mass of an object(s) or an amount of material
MOTION PROGRAM	An error occurred in the motion program.
PATH FEEDRATE	The federate of the tool path
PATH POSITION	The current control point of the path

Data Item type/ qualifier	Description
PH	The measure of acidity or alkalinity
POSITION	The position of a `Component`.
POWER_FACTOR	The ratio of real power flowing to a load to the apparent power in that AC circuit.
PRESSURE	The measurement of the force per unit area exerted by a gas or liquid.
RESISTANCE	The measurement of the degree to which an object opposes an electric current through it
ROTARY_VELOCITY	The rotational speed of a rotary axis
SOUND_LEVEL	The measurement of sound pressure level
SPINDLE_SPEED	**DEPRECATED in *Rel 1.2*. See ROTARY_VELOCITY**
STRAIN	Indicates the amount of deformation per unit length of an object when a load is applied
SYSTEM	A condition representing something that is not the operator, program, or hardware. This is often used for operating system issues.
TEMPERATURE	Indicates the temperature of a `Component`.
TILT	The measure of angular displacement
TORQUE	The measured of the turning force exerted on an object or by an object
VOLT_AMPERAGE	The measure of the apparent power in an electrical circuit (commonly referred to as VA)
VOLT_AMPERAGE_REACTIVE	The measure of reactive power in an AC electrical power circuit (commonly referred to as var).
VELOCITY	Indicated the velocity of a component.
VISCOSITY	The measure of a fluid's resistance to flow
VOLTAGE	The measurement of electrical potential between two points
WATTAGE	The measurement of power consumed or dissipated by an electrical circuit or device

3.5.13 Schema Structure for DataItems

The following document structure defines a typical machine with rotary and linear axes and a controller.

```
MTConnectDevices
   Devices
      Device
         Components
            Axes
               Rotary [C]
                  DataItems
                     DataItem [Cvel]
                        Constraints SPINDLE
               Linear  [X]
                  DataItems
                     DataItem [Xpos]
               Linear  [Y]
                  DataItems
                     DataItem [Ypos]
               Linear  [Z]
                  DataItems
                     DataItem [Zpos]
            Controller
               Path
                  DataItems
                     DataItem [mode]
                     DataItem [execution]
```

The above example shows how the various containers make it easier to address individual parts of the XML document. For example, if one wanted to retrieve only the `DataItems` for the `Controller,` you can express this using the following XPath: `//Controller/DataItems/*`. If you were interested in retrieving only the *subcomponets* of the `Axes` component, you would write the following XPath: `//Axes/Components/*`.

3.6 `Component` Types and *Subcomponents*

`Component` is an abstract type that allows for extensibility. As the specification progresses, more component types will be added to support new devices and parts of new devices. Some examples of component types are `Axes`, `Controller`, and `Systems`. Any of these component types can have data items and *subcomponents*. Appendix B contains reference models for common equipment to guide developers in implementing MTConnect on their devices.

The `Component` types presently define include:

3.6.1 Axes

Axes is the root of all device components that have linear or rotational motion. Currently there are only Linear and Rotary axes supported and when axes are defined the Axes component **MUST** contain at least one Linear or Rotary axis. The Linear axes **MUST** be named X, Y, Z with numbers appended for additional axes in the same plane, for example X2, Y2, and Z2 are the secondary axes to X, Y, and Z. Rotary axes **MUST** be named A, B, and C and 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.

The Axes represent the physical data for the axis components. When data is defined specifically for the physical axes, positions **MUST** be given in MACHINE coordinates. The WORK coordinates are represented in the Path component of the Controller.

DEPRECATION WARNING: In *Version 1.1* of the MTConnect® standard, the Spindle component was no longer supported. The Spindle will now be represented by a rotary axis that has a RotaryMode of SPINDLE. The S(n) axis nomenclature **SHOULD** be removed and replaced with A, B, or C to clearly identify which primary plane the rotary axis is rotating around. All associated DataItems **SHOULD** now be named accordingly.

Note: The convention for multiple linear and rotary axes having the same designation is to index the axes letter with a number. For this standard, the secondary axis number starts at 2 (i.e. X, X2, X3, … or C, C2, C3, C4, …). This is in compliance with the ISO-841-2001. Please refer to that specification for more details.

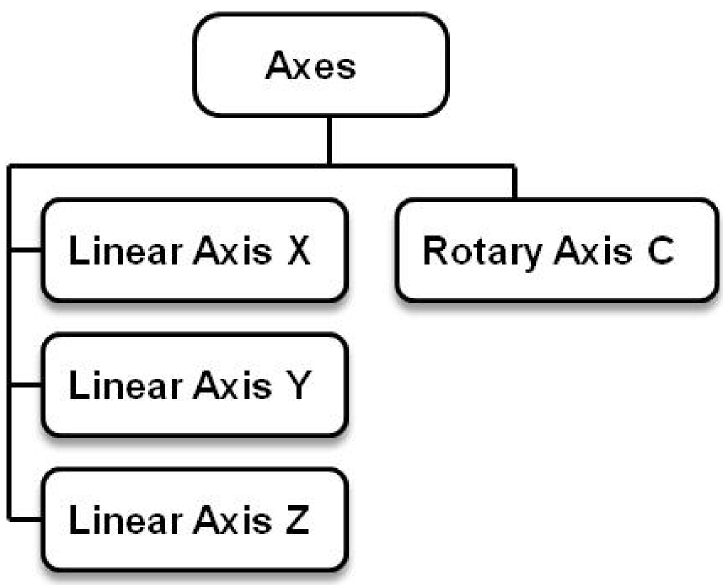

Figure 9: Axes Example With Three Linear Axes and one Rotary Axis

`Linear` A linear axis represents the movement of a physical device, or a portion of a device, in a straight line. Movement may be in either a positive or negative direction.

`Rotary` An axis whose function is to provide rotary motion may function as a continuous rotation (i.e. spindle mode), continuous-path contour cutting in a rotary motion (i.e. contouring), or repositioning (i.e. indexing) different faces of the part. As such, a rotary axis **MUST** operate in one of the three following modes: `SPINDLE`, `INDEX`, or `CONTOUR`.

3.6.2 Controller

The `Controller` component represents an intelligent device. Examples include a `CNC` (Computer Numerical Control) or `PAC` (Programmable Automation Control) which may be referred to as a *Motion Control* or *General Purpose Motion Control*. The `Controller` provides information regarding the execution of a control program and the execution state of the device. There are no required *subcomponents* of the `Controller`.

Note: MTConnect *Version 1.1.0* and later implementations **SHOULD** use a `Path` sub-component to represent an individual tool path and execution state (see `Path`). When the machine is capable of executing more than one simultaneous program, the implementation **MUST** use the `Path` component.

3.6.2.1 Path

For more complex devices and controllers, each path will be represented by a `Path` *subcomponent*. A `Path` represents the motion of a control point as it moves through space as controlled by a set of control instructions (i.e. vector move). The `Path` will encapsulate the position, feedrate, and rotation of the control point as presented by the controller. The control point is the positioning of a tool at a point in space.

If the controller is capable of running more than one task simultaneously, a `Path` component **MUST** be given for each task under the `Controller` component.

3.6.3 ~~Power~~ DEPRECATED in *Rel. 1.1*

NOTE: Power as an indication of `Availability` will be changed to a `DataItem` called `Availability` and electrical current and power consumption will be represented by the `Electric` system, see *3.6.9.5 Electric* below.

3.6.4 Door

This component represents a door closure that can be opened or closed. It **MUST** have a `DataItem` called `DoorState` to indicate if it is opened, closed or unlatched. A device may contain multiple door components.

3.6.5 Actuator

An `Actuator` is a device for moving or controlling a mechanism or system. It takes energy, usually transported by air, electric current, or liquid and converts it into some kind of motion. An `Actuator` may be a `Component` of a `Device` or it may be a *subcomponent* of a parent `Component`.

3.6.6 Sensor

`Sensor` is an abstract type component that provides measurement data related to a `Device` or `Component`. Depending on the type of data provided by the sensor, it may be modeled in the XML schema in different ways. However, it will always be modeled to associate the data contained in `Sensor` with the `Component` XML Element to which the data is most closely associated.

A sensor is typically comprised of two major components – the *sensing element* (provides a signal or measured value) and the *sensor interface* (signal processing, conversion, and communications). In MTConnect, the *sensor interface* is modeled as a `Component` called `Sensor`. The *sensing element* or measured value is modeled as a `DataItem`. 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 `DataItem` of type PRESSURE.

`Sensor` **MUST NOT** be modeled in the plural. `Sensor` will always refer to the *sensor interface*. Each *sensor interface* may have multiple *sensing elements*; each representing the data for a variety of measured values.

3.6.6.1 Sensor data

The most basic implementation of a *sensing element* is the providing of a measured value associated with a `Component` which is the `Sensor` data. An example would be the measured value of the `Temperature` of the spindle (Rotary Axis C). This would be represented as a `DataItem` called `Temperature` that is associated with the Rotary Axis C as follows:

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

A sensor may measure values associated with any `Component`, `Sub-Component`, or `Device`. Some examples of how sensor data may be modeled are represented in Figure 6 below:

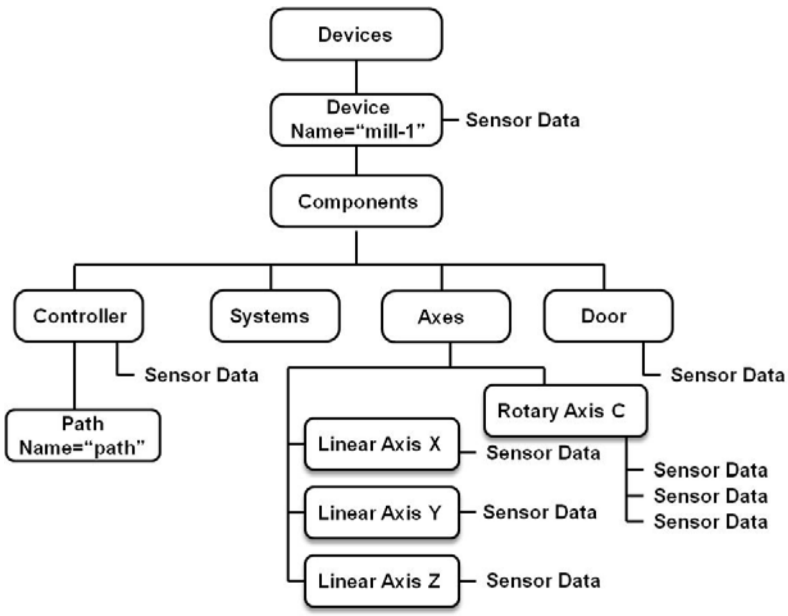

Figure 10: `Sensor` Data Associations

3.6.6.2 Sensor Interface

Sensing element(s) are most typically connected to a *sensor interface*. The *sensor interface* provides additional information concerning the *sensing element(s)*.

Typical functions of the *sensor interface* include:

- convert low level signals from the *sensing elements* into data that can be used by other devices. (Example: Convert a non-linear millivolt signal from a temperature sensor into a scaled temperature value that can be transmitted to another device.)

- 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 interface*. (Example: The *sensor interface* may provide diagnostics on each *sensing element* (e.g. open wire detection) and itself (e.g. measure internal temperature of the *sensor interface*).

The *sensor interface* is modeled in the XML schema as a Component called Sensor. Sensor **SHOULD** be modeled in the XML schema so that the Sensor is represented as part of the Component to which it is most closely associated.

Sensor may be associated with any Component, Sub-Component, or Device. Some examples of where a sensor may be modeled are represented in Figure 7 below:

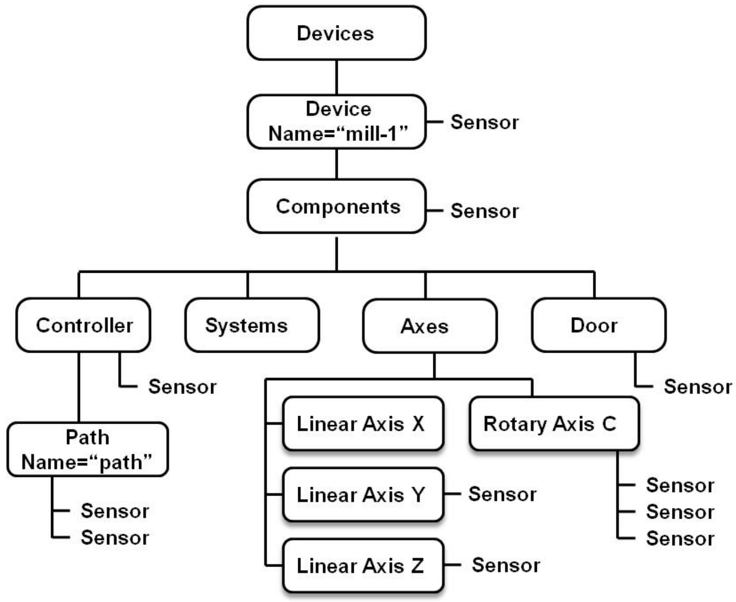

Figure 11: Sensor Associations

When a Sensor is modeled as a Component, it **MAY** have its own uuid so it can be tracked throughout its lifetime.

The following examples demonstrate how Sensor may be modeled in the XML schema differently based on how the sensor functions within the overall Device.

Example#1: If Sensor provides vibration measurement data for the spindle, it should be modeled as a Sensor for Rotary Axis C.

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

Example#2: If `Sensor` provides measurement data for multiple `Components` within a `Device` and is not associated with any particular `Component`, it MAY be modeled in the XML schema as an independent `Component` of the `Device`.

```
<Device id="d1" uuid="HM1" name="HMC_3Axis">
  <Description>3 Axis Mill</Description>
  <Components>
     <Sensor id="sensor" name="sensor"/>
        <DataItems>
           <DataItem type="TEMPERATURE" id="sentemp" category="SAMPLE"
                  name="Sensortemp" units="DEGREE"/>
        </DataItems>
  </Components>
</Device>
```

While `Sensor` MAY be modeled in different ways in the XML schema, the measured value of the *sensing element* **MUST** always be modeled as a `DataItem` associated with the `Component` to which the measured value is most closely associated.

Example#3: In this case, `Sensor` is modeled as a `Component` within a `Device.` Its measured values from the *sensing elements* are associated with other `Components` in the `Device`. The sensor also has internal diagnostics capabilities representing the condition of the sensor itself.

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

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

3.6.6.3 Sensor as a `Device`

A `sensor` may function as an independent device. In this case, it is not associated with a parent `Device` or `Component`.

Examples of a sensor functioning as a `Device` would be a sensor used to monitor the ambient temperature of a building or an air quality monitoring system. Another example would be a vibration monitoring system that is moved from one machine to another. In these cases, the sensor functions as an intelligent device performing a specific function.

A sensor functioning as a `Device` would be modeled in the XML schema as follows:

```
<Device id="s1" uuid="HM1" name="AMBIENT_MONITOR">
  <Description>Ambient Temperature Monitor</Description>
  <DataItems>
    <DataItem type="TEMPERATURE" id="ambtemp" category="SAMPLE"
       name="Ambienttemp" units="DEGREE"/>
  </DataItems>
</Device>
```

A sensor that is modeled as a device **MUST** have an `uuid` so that it can be uniquely tracked.

3.6.7 `Sensor` Configuration

When a sensor is modeled in the XML schema as a `Component` or a `Device`, it may provide additional configuration information for the *sensor elements* and the *sensor interface* itself.

The `Sensor` configuration data provides information required for maintenance and support of the sensor.

`Sensor` configuration data is *only* available when the sensor is modeled as a `Component` or a `Device`. For details on the modeling of `Configuration` data in the XML schema, see *Part 2, Section 3.4.7.1* `Component` *Configuration*. Details specific to `SensorConfigurationType` are provided below.

When `Sensor` represents the *sensor interface* for multiple *sensing element(s)*, each *sensing element* is represented by a `Channel`. Each `Channel` represents one *sensing element* and can have its own `attributes` and `Configuration` data.

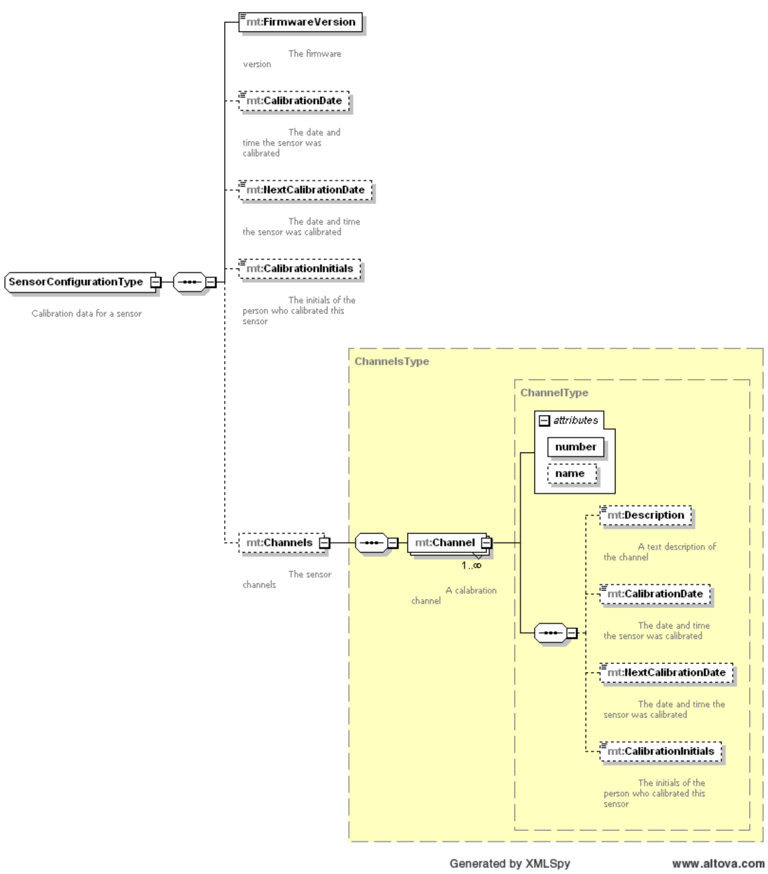

Figure 7: **`Configuration` Data for `Sensors`**

Element	Description	Occurrence
`Configuration` (`SensorConfigurationType`)	An element that can contain descriptive content defining the configuration information for Sensor. For Sensor, the valid configuration is `SensorConfiguration`. `SensorConfiguration` 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

3.6.7.1 SensorConfiguration Elements

Element	Description	Occurrence
`FirmwareVersion`	Version number for the sensor as specified by the manufacturer	1
`CalibrationDate`	Date upon which the sensor was last calibrated. Dates **MUST** be represented in the W3C ISO 8601 format	0..1
`NextCalibrationDate`	Date upon which the sensor is next scheduled to be calibrated. Dates **MUST** be represented in the W3C ISO 8601 format	0..1
`CalibrationInitials`	The initials of the person verifying the validity of the calibration data	0..1
`Channels`	When `Sensor` represents multiple *sensing elements*, each *sensing element* is represented by a `Channel` for the `Sensor`.	0..1

3.6.7.2 Sensor `Channel` Attributes

`Channel` represents each *sensing element* connected to a *sensor interface*. Each `Sensor` `Channel` has the following composition:

Attribute	Description	Occurrence
`Number`	A unique identifier that will only refer to this *sensing element*. For example, this can be the manufacturer code and the serial number. The `Number` should be alphanumeric and not exceeding 255 characters. An NMTOKEN XML type.	1
`Name`	The `Name` of the *sensing element*. This name should be unique within the machine to allow for easier data integration. An NMTOKEN XML type.	0..1

3.6.7.3 Sensor `Channel` Elements

Element	Description	Occurrence
`Description`	An XML element that can contain any descriptive content. This can contain information about the *sensor element* and manufacturer specific details.	0..1
`CalibrationDate`	Date upon which the *sensor element* was last calibrated. Dates **MUST** be represented in the W3C ISO 8601 format	0..1
`NextCalibrationDate`	Date upon which the *sensor element* is next scheduled to be calibrated. Dates **MUST** be represented in the W3C ISO 8601 format	0..1
`CalibrationInitials`	The initials of the person verifying the validity of the calibration data	0..1

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

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

3.6.8 Sensor Types

Types of measurements provided by `Sensor` include:

3.6.8.1 Thermostat (DEPRECATED in Rel. 1.2. See TEMPERATURE)

3.6.8.2 Vibration (DEPRECATED in Rel. 1.2. See DISPLACEMENT, FREQUENCY, etc.)

3.6.8.3 Acceleration

The measurement of the linear acceleration of an object.

3.6.8.4 Angular Acceleration

The measurement of the acceleration of a rotating object

3.6.8.5 Angular Velocity

The measurement of the velocity of a rotating object

3.6.8.6 Amperage

The measurement of electrical current flow.

3.6.8.7 Angle

The measurement of angular position of an object.

3.6.8.8 Concentration

The measurement of how much of a substance is mixed with another substance.

3.6.8.9 Conductivity

The measurement of the ability of a material to conduct electricity.

3.6.8.10 Direction

The direction of movement of an object. Normally, this will be reported as clockwise and counter clockwise for rotary motions and positive or negative for linear motions.

3.6.8.11 Displacement

The measurement of the distance of movement of an object.

3.6.8.1 Electrical Energy

The measurement of electrical energy consumed by a component over a period of time.

3.6.8.2 Flow

The measurement of the rate at which a volume of a fluid moves within a system.

3.6.8.3 Frequency

The measurement of the number of occurrences of a repeating event per unit time.

3.6.8.4 Fill Level

The measurement of the amount of a substance remaining compared to the planned maximum of that substance.

3.6.8.5 Linear Force

The magnitude of push or pull introduced by an actuator or exerted on an object.

3.6.8.6 Load

The measurement of the percentage of the standard rating of a device.

3.6.8.7 Mass

The measurement of the mass of an object(s) or an amount of material.

3.6.8.8 PH

The measurement of the acidity or alkalinity.

3.6.8.9 Pressure

The measurement of the force per unit area exerted by a gas or liquid.

3.6.8.10 Position

The measurement of an object's position relative to a coordinate system.

3.6.8.11 Power Factor

The measurement of the ratio of real power flowing to a load to the apparent power in an AC circuit.

3.6.8.12 Resistance

The measurement of the degree to which an object opposes an electric current through it

3.6.8.13 Rotary Velocity

The measurement of the rotational speed of a rotating object.

3.6.8.14 Sound Level

The measurement of sound level.

3.6.8.15 Strain

The measurement of the amount of deformation per unit length of an object.

3.6.8.16 Temperature

The measurement of the temperature of an object.

3.6.8.17 Time

The measurement of time: may be reported as accumulated time associated with a component or clock time.

3.6.8.18 Tilt

The measurement of the angular displacement of an object.

3.6.8.19 Torque

The measurement of torque applied to or by an object.

3.6.8.20 Volt Ampere (VA)

The measure of the apparent power in an electrical circuit, equal to the product of root-mean-square (RMS) voltage and RMS current.

3.6.8.21 Volt Ampere Reactive (var)

The measurement of reactive power in an AC electric power system

3.6.8.22 Velocity

The measurement of the linear velocity of an object.

3.6.8.23 Viscosity

The measurement of a fluid's resistance to flow.

3.6.8.24 Voltage

The measurement of electrical potential between two points

3.6.8.25 Wattage

The measurement of power consumed or dissipated by an electrical circuit or device

3.6.9 Systems

A component similar to `Axes` that groups *subcomponents* that comprise complex `Components` that are not easily deconstructed. `Systems` will be used to represent general information about the health and viability of all of its parts and sub-parts.

3.6.9.1 Hydraulic

A hydraulic system comprises all the parts involved in moving and distributing pressurized liquid for the purpose of delivering a source of power to specific types of actuators.

3.6.9.2 Pneumatic

A pneumatic system comprises all the parts involved in moving and distributing pressurized gas regardless of purpose or activity.

3.6.9.3 Coolant

The coolant system comprises all the parts involved in distribution and management of coolants.

3.6.9.4 Lubrication

The lubrication system comprises all the parts involved in distribution and management of the lubricants.

3.6.9.5 Electric

The electric system represents the main power supply or generator for the device. The electric system will provide all the data with regard to current, voltage, and frequency that applies overall to the `Device`. Data regarding electric power that is specific to a component or *subcomponent* will be reported as a `DataItem` of that `Component`.

4 Component and Data Item Relationships

This section will discuss the association between `Component`, `DataItem`, and `DataItem` categories (`Events`, `Condition`, and `Samples`). For each `Component`, there are a limited set of allowable *subcomponents* and a limited set of `DataItems`. For example, an `Axes` component may not have a `Device` or a `Controller` as a child, and it may not have `Block` as a `DataItem` type, since it is incapable of running a program.

Many types of `DataItem` can be applied to a wide variety of `Component(s)`. In the sections below, only those types of `DataItem` that are specific to each `Component` will be defined. By inference, all other types of `DataItem` may be applied to these `Component(s)` as required.

4.1 Device

The `Device` is the only top level element in the component tree. Since an MTConnect® *Agent* can manage multiple devices, the schema provides a top level container `Devices` to hold the `Device` elements.

A device **MUST** always contain an `Availability` data item that represents this device is functioning and able to communicate.

4.1.1 Device DataItems

- `EMERGENCY_STOP` - The emergency stop state of the machine or device.
- `AVAILABILITY` - **Required**

4.1.2 Components of Device

- `Axes`
- `Controller`
- `Systems`
- `Door`
- `Actuator`
- `Sensor`

4.2 Common Components and Related Data Items

A common set of `DataItems` have been defined to provide a wide variety of information about a machine or process. Any `DataItem` can be used with any `Device` or `Component` providing that the standard naming conventions are implemented. Any `Component` **MAY** also include an arbitrary set of sensors that may be modeled as either a *subcomponent* or a `DataItem`. A `Sensor` may be an external device that will collect data and report it to the *Agent*.

Additionally, `Conditions` are defined as a specific category of `DataItem` that indicates the health of a `Component` or `Device`. Any `Condition` can be used with any `Device` or `Component` providing that the standard naming conventions are implemented.

Only the types of `DataItem` unique to each `Component` are detailed below. It can be assumed that all other type of `DataItem` can be applied to any of the `Components`.

4.2.1 Axes

The `Axes` component is a container for the actual axes of which there are currently two types: `Linear` and `Rotary`.

~~4.2.1.1 Axes DataItems~~

- ~~GLOBAL_POSITION~~ – DEPRECATED in Rel 1.1
- ~~PATH_FEEDRATE~~ – Moved to Path
- ~~ACCELERATION~~ – Moved to Path
- ~~VELOCITY~~ – Moved to Path

4.2.1.2 *Subcomponents* of Axes

- Linear
- Rotary
- ~~Spindle~~ – DEPRECATED in *Rel 1.1*

4.2.2 Linear (Subcomponent of Axes)

A linear axis represents travel along a straight line. The name of the linear axis **SHOULD** follow the conventions of the industry.

4.2.2.1 Linear Axes' DataItems (Sample and Event)

- AXIS_FEEDRATE
- DIRECTION
- POSITION
- ~~SLAVE_OF_AXIS~~ (DEPRECATED in *Rel. 1.1*)
- LINEAR_FORCE
- VELOCITY

4.2.2.2 Linear Axes' Condition

- POSITION
- LINEAR_FORCE
- VELOCITY

4.2.3 Rotary (*Subcomponent* of Axes)

A rotary axis revolves around a line or vector.

4.2.3.1 Rotary Axes' DataItems (Sample and Event)

- ANGLE
- ANGULAR_ACCELERATION
- ANGULAR_VELOCITY
- DIRECTION
- ROTARY_MODE
- ROTARY_VELOCITY
- ~~SLAVE_OF_AXIS~~ **DEPRECATED in *Rel 1.1***
- ~~SPINDLE_SPEED~~ **DEPRECATED in *Rel 1.2*. Replaced by ROTARY_VELOCITY**
- TORQUE

4.2.3.2 Rotary Axes' Condition

- ANGLE
- ANGULAR_ACCELERATION
- ANGULAR_VELOCITY
- ROTARY_VELOCITY
- TORQUE

4.2.4 Controller

The Controller component is the Component that controls a device, executes a program, and sends instructions to the other components of the machine. It is the brains of the machine and can be asked for its current execution state and program name.

4.2.4.1 *Subcomponents* of Controller

- Path

4.2.4.2 Controller DataItems (Sample and Event)

- ~~CODE~~ **DEPRECATED in *Rel 1.1***
- CONTROLLER_MODE
- EXECUTION
- EMERGENCY_STOP
- MESSAGE
- PALLET_ID
- PART_COUNT
- PART_ID
- PROGRAM
- ~~TOOL_ID~~ **DEPRECATED in *Rel 1.2*.**
- TOOL_ASSET_ID
- WORKHOLDING_ID

4.2.4.3 Controller Condition

- COMMUNICATIONS
- HARDWARE
- LOGIC_PROGRAM
- MOTION_PROGRAM
- SYSTEM

4.2.5 Path (*Subcomponent* of Controller)

A Path represents the motion of a control point as it moves through space as controlled by a set of control instructions (i.e. vector move). When Path is not defined, DataItems relative to the Path **MAY** be reported for the Controller.

4.2.5.1 Path DataItems (Sample and Event)

- ACTIVE_AXES
- AXIS_COUPLING
- BLOCK
- ~~CODE~~ **DEPRECATED**
- COUPLED_AXES
- CONTROLLER_MODE
- EMERGENCY_STOP
- EXECUTION
- LINE
- PALLET_ID
- PART_COUNT
- PART_ID
- PATH_FEEDRATE
- PATH_POSITION
- PROGRAM
- ~~TOOL_ID~~ **DEPRECATED in Rel 1.2.**
- TOOL_ASSET_ID
- VELOCITY
- WORKHOLDING_ID

4.2.5.2 Path Condition

- MOTION_PROGRAM

4.2.6 ~~Power~~ DEPRECATED in *Rel 1.1*

4.2.7 Sensors

Sensor is a component that may or may not be integral to a parent component or device. When Sensor is not integral to a parent device or component – it can function as a device. Sensor data **MUST** be associated with its most relevant Component and **MUST** be represented as a DataItem for that Component.

4.2.7.1 Sensor Condition

- COMMUNICATION
- HARDWARE

4.2.8 ~~Thermostat~~ DEPRECATED in *REL 1.2*. Replaced with a **DataItem** called **Temperature**

~~A sensor capable of measuring the temperature of a component. The temperature is always given in Celsius.~~

~~4.2.8.1 DataItem types~~

- ~~TEMPERATURE~~

~~4.2.8.2 Condition types~~

- ~~COMMUNICATION~~
- ~~HARDWARE~~
- ~~TEMPERATURE~~

4.2.9 ~~Vibration~~ DEPRECATED in *REL 1.2*. Replaced with **DataItems** to measure vibration **(Displacement, Frequency,** etc).

~~A sensor capable of measuring the vibration of a component.~~

~~4.2.9.1 DataItem types~~

- ~~ACCELERATION~~
- ~~DISPLACEMENT~~
- ~~FREQUENCY~~
- ~~VELOCITY~~

~~4.2.9.2 Condition types~~

- ~~ACCELERATION~~
- ~~COMMUNICATION~~
- ~~DISPLACEMENT~~
- ~~HARDWARE~~
- ~~VIBRATION~~

~~4.2.10 Pressure~~ DEPRECATED in *REL 1.2*. Replace with **`DataItem Pressure`**

~~A sensor capable of measuring the pressure.~~

~~4.2.10.1 DataItem types~~

- ~~PRESSURE~~

~~4.2.10.2 Condition types~~

- ~~COMMUNICATION~~
- ~~HARDWARE~~
- ~~PRESSURE~~

4.2.11 Door

A opening that can be closed.

4.2.11.1 Door DataItems (Sample and Event)

- DOOR_STATE

4.2.11.2 Door Condition

- DOOR_STATE
- COMMUNICATIONS
- HARDWARE

4.2.12 Actuator

A mechanical device for moving or controlling a mechanism or system.

4.2.12.1 Acutator DataItems (Sample and Event)

- ACTUATOR_STATE

4.2.12.2 Actuator Condition

- COMMUNICATIONS
- HARDWARE

4.2.13 ~~Spindle~~ – DEPRECATED in *Rel. 1.1*

~~The spindle is a rotational axis that revolves at high speed and has its speed expressed in REVOLUTION/MINUTE~~

4.2.14 Systems

The `Systems` component is a place holder for all the system types.

4.2.14.1 *Subcomponents* of Systems

- Hydraulic
- Pneumatic
- Coolant
- Lubrication
- Electric

4.2.15 Hydraulic (*Subcomponent* of Systems)

A component representing the hydraulics and hydraulic distribution system of a device.

4.2.15.1 Hydraulic Condition

- COMMUNICATIONS
- HARDWARE

4.2.16 Pneumatic (*Subcomponent* of Systems)

A component representing the pneumatics and compressed air distribution system of a device.

4.2.16.1 Pneumatic Condition

- COMMUNICATIONS
- HARDWARE

4.2.17 Coolant (*Subcomponent* of Systems)

A Component representing the coolant and coolant distribution system of a device.

4.2.17.1 Coolant DataItems (Sample and Event)

- CONCENTRATION
- CONDUCTIVITY
- PH
- VISCOSITY

4.2.17.2 Coolant Condition

- COMMUNICATIONS
- HARDWARE
- CONCENTRATION
- CONDUCTIVITY
- PH
- VISCOSITY

4.2.18 Lubrication (*Subcomponent* of Systems)

A Component representing the lubricant and lubrication distribution system of a device.

4.2.18.1 Lubrication DataItems (Sample and Event)

- PH
- VISCOSITY

4.2.18.2 Lubrication Condition

- COMMUNICATIONS
- HARDWARE
- PH
- VISCOSITY

4.2.19 `Electric` (*Subcomponent* of `Systems`)

A `Component` representing the electrical supply for a device.

4.2.19.1 Electrical DataItems (Sample and Event)

- AMPERAGE
- ELECTRICAL_ENERGY
- FREQUENCY
- POWER_FACTOR
- POWER_STATE
- VOLTAGE
- VOLT_AMPERE
- VOLT_AMPERE_REACTIVE
- WATTAGE

4.2.19.2 Electric Condition

- AMPERAGE
- FREQUENCY
- VOLTAGE
- WATTAGE

5 Annotated XML Examples

5.1 Simplest Device

For the simplest possible device, we are modeling a saw that has only an `Availability` (the minimal set of `DataItem`). To retrieve this information, we send the following request to the *Agent*:

```
http://10.1.23.10/ LinuxCNC/probe
```

The *Agent* responds as follows:

```
1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <MTConnectDevices xmlns:m="urn:mtconnect.com:MTConnectDevices:0.9"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="urn:mtconnect.com:MTConnectDevices:0.9"
    xsi:schemaLocation="urn:mtconnect.com:MTConnectDevices:0.9
    /schemas/MTConnectDevices.xsd">
3.     <Header sender="10.1.23.10" bufferSize="100000"
          creationTime="2008-07-07T23:07:50-07:00" version="0.9"
          instanceId="1214527986"/>
```

Line 3 provides the `instanceId` as a unique number for this request. For this example, the *Agent* does not persist the `Samples`, `Events`, and `Condition`. Therefore, this number will change every time that it is recorded. The `bufferSize` indicates that this *Agent* is capable of storing 100,000 DataItem of category `Sample`, `Event`, and `Condition`.

```
4.   <Devices>
5.     <Device iso841Class="6" uuid="linux-01" name="LinuxCNC"
               sampleInterval="100.0" id="d">
6.       <Description manufacturer="NIST" serialNumber="01"/>
```

The above device description includes the unique `id` and a sample interval of ten times per second. Since there are no telemetry data being collected, sampling at once per second is typically adequate.

```
7.       </Components>
8.        <DataItems>
9.           <DataItem type="AVAILABILITY" name="avail" category="EVENT"
                  id="a"/>
10.        </DataItems>
```

As was stated previously, the device is only required to have one `DataItem` and it is of the type `AVAILABILITY` which **MUST** report the device's represent ability to communicate. The `DataItem` on line 9 has an `id` of "a". This will allow events responding to this `DataItem` to be easily associated.

```
11.        </Components>
12.      </Device>
13.    </Devices>
14.  </MTConnectDevices>
```

Lines 11 through 14 terminate each element type and close the document.

5.2 More Complex Example of `Probe`

The `Sample` was generated with the following request:

```
http://10.1.23.5/LinuxCNC/probe
```

The following is an example of a 3 axis mill simulation. The mill has three linear axes and one rotary axis (spindle):

```
1.   <?xml version="1.0" encoding="UTF-8"?>
2.   <MTConnectDevices xmlns:m="urn:mtconnect.com:MTConnectDevices:0.9"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xmlns="urn:mtconnect.com:MTConnectDevices:0.9"
     xsi:schemaLocation="urn:mtconnect.com:MTConnectDevices:0.9
     /schemas/MTConnectDevices.xsd">
3.     <Header sender="10.1.23.5" bufferSize="100000" creationTime=
           "2008-07-07T23:07:50-07:00" version="0.9"
           instanceId="1214527986"/>
4.      <Devices>
5.       <Device iso841Class="6" uuid="linux-01" name="LinuxCNC"
             sampleRate="100.0" id="d1">
```

Here we provide the top level container `Devices` and the information on the `Device`.

```
6.        <Description manufacturer="NIST" serialNumber="01"/>
7.          <DataItems>
8.            <DataItem type="AVAILABILITY" name="avail" category="EVENT"
                id="a"/>
9.          </DataItems>
10.     <Components>
11.      <Axes name="Axes" id="3">
```

On line 11 we introduce the collection of `Axes`. The `Axes` component is a special component that acts as an abstract component as well as a collection. The `Axes` component contains various `DataItems` that have a global context; they are not associated with any one axis but they go across all axes.

```
12.       <Components>
13.         <Rotary name="C" id="c1">
14.           <DataItems>
15.             <DataItem type="ROTARY_VELOCITY" name="Cspeed" category="SAMPLE"
                     id="c2" nativeUnits="REVOLUTION/MINUTE" subType="ACTUAL"
                     units="REVOLUTION/MINUTE">
16.                <Source>Sspeed</Source>
17.             </DataItem>
18.               <DataItem type="ROTARY_MODE" name="Cmode" category="EVENT"
                       id="c3">
19.                  <Constraints>
20.                     <Value>SPINDLE</Value>
21.                  </Constraints>
22.               </DataItem>
23.             </DataItems>
24.         </Rotary>
```

The spindle component (Rotary Axis C) declared on line 13 is the C axis and has spindle specific `DataItems`.

```
25.         <Linear name="X" id="x1">
26.           <DataItems>
27.             <DataItem type="POSITION" name="Xact" category="SAMPLE" id="x2"
               nativeUnits="MILLIMETER" subType="ACTUAL" units="MILLIMETER"/>
28.             <DataItem type="POSITION" name="Xcom" category="SAMPLE" id="x3"
               nativeUnits="MILLIMETER" subType="COMMANDED"
               units="MILLIMETER"/>
29.           </DataItems>
30.         </Linear>
31.         <Linear name="Y" id="y1">
32.           <DataItems>
33.             <DataItem type="POSITION" name="Yact" category="SAMPLE" id="y2"
                nativeUnits="MILLIMETER" subType="ACTUAL"
                units="MILLIMETER"/>
34.             <DataItem type="POSITION" name="Ycom" category="SAMPLE" id="y3"
                nativeUnits="MILLIMETER" subType="COMMANDED"
                units="MILLIMETER"/>
35.           </DataItems>
36.         </Linear>
37.         <Linear name="Z" id="z1">
38.          <DataItems>
39.            <DataItem type="POSITION" name="Zact" category="SAMPLE" id="z2"
               nativeUnits="MILLIMETER" subType="ACTUAL" units="MILLIMETER"/>
40.
```

```
              <DataItem type="POSITION" name="Zcom"
                category="SAMPLE" id="z3"
                nativeUnits="MILLIMETER" subType="COMMANDED"
                units="MILLIMETER"/>
41.         </DataItems>
42.        </Linear>
43.      </Components>
44.      </Axes>
```

Lines 25, 31, and 37 define the three linear axes X, Y, and Z respectively. In this example device, the *Agent* is only collecting the actual and commanded positions.

The `Controller` is capable of providing the program name, block, and the current line being executed:

```
45.       <Controller name="Controller" id="cn8">
46.        <Components>
47.         <Path id="pth1" name="path">
48.           <DataItems>
49.             <DataItem type="LINE" name="line" category="EVENT" id="p1"/>
50.             <DataItem type="CONTROLLER_MODE" name="mode" category="EVENT"
                  id="p2"/>
51.             <DataItem type="PROGRAM" name="program" category="EVENT"
                  id="p3"/>
52.             <DataItem type="EXECUTION" name="execution" category="EVENT"
                  id="p4"/>
53.             <DataItem type="PATH_FEEDRATE" name="feedrate"
54.               category="SAMPLE" id="p5" units="MILLIMETER/SECOND"
                  nativeUnits="MILLIMETER/SECOND" />
55.             <DataItem type="PATH_POSITION" name="position"
56.               category="SAMPLE" id="p6" units="MILLIMETER_3D"
57.               nativeUnits="INCH_3D"/>
58.           </DataItems>
59.         </Path>
60.        </Components>
61.       </Controller>
62.      </Components>
        </Device>
      </Devices>
63.  </MTConnectDevices>
```

Appendices

A. Bibliography

1. Engineering Industries Association. *EIA Standard - EIA-274-D*, Interchangeable Variable, Block Data Format for Positioning, Contouring, and Contouring/Positioning Numerically Controlled Machines. Washington, D.C. 1979.
2. ISO TC 184/SC4/WG3 N1089. *ISO/DIS 10303-238*: Industrial automation systems and integration Product data representation and exchange Part 238: Application Protocols: Application interpreted model for computerized numerical controllers. Geneva, Switzerland, 2004.
3. International Organization for Standardization. *ISO 14649*: Industrial automation systems and integration – Physical device control – Data model for computerized numerical controllers – Part 10: General process data. Geneva, Switzerland, 2004.
4. International Organization for Standardization. *ISO 14649*: Industrial automation systems and integration – Physical device control – Data model for computerized numerical controllers – Part 11: Process data for milling. Geneva, Switzerland, 2000.
5. International Organization for Standardization. *ISO 6983/1* – Numerical Control of machines – Program format and definition of address words – Part 1: Data format for positioning, line and contouring control systems. Geneva, Switzerland, 1982.
6. Electronic Industries Association. *ANSI/EIA-494-B-1992*, 32 Bit Binary CL (BCL) and 7 Bit ASCII CL (ACL) Exchange Input Format for Numerically Controlled Machines. Washington, D.C. 1992.
7. National Aerospace Standard. *Uniform Cutting Tests* - NAS Series: Metal Cutting Equipment Specifications. Washington, D.C. 1969.
8. International Organization for Standardization. *ISO 10303-11*: 1994, Industrial automation systems and integration Product data representation and exchange Part 11: Description methods: The EXPRESS language reference manual. Geneva, Switzerland, 1994.
9. International Organization for Standardization. *ISO 10303-21*: 1996, Industrial automation systems and integration -- Product data representation and exchange -- Part 21: Implementation methods: Clear text encoding of the exchange structure. Geneva, Switzerland, 1996.
10. H.L. Horton, F.D. Jones, and E. Oberg. *Machinery's handbook*. Industrial Press, Inc. New York, 1984.
11. International Organization for Standardization. *ISO 841-2001: Industrial automation systems and integration - Numerical control of machines - Coordinate systems and motion nomenclature.* Geneva, Switzerland, 2001.

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

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

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

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

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

B. Machine Tool Modeling

The following section will provide example machine tool configurations and reference MTConnect® implementations. The following is the recommended machine modeling and implementation reference.

MTConnect utilizes the right hand rule for all coordinate systems representing physical space and orientation within a machine. The positive movement is given by extending the first three fingers on the right hand and labeling the axes in order of the digits, X, Y, and Z. The fingers will point in the positive direction. All linear axes represent a space within a machine that is defined by coordinates according to the right hand rule.

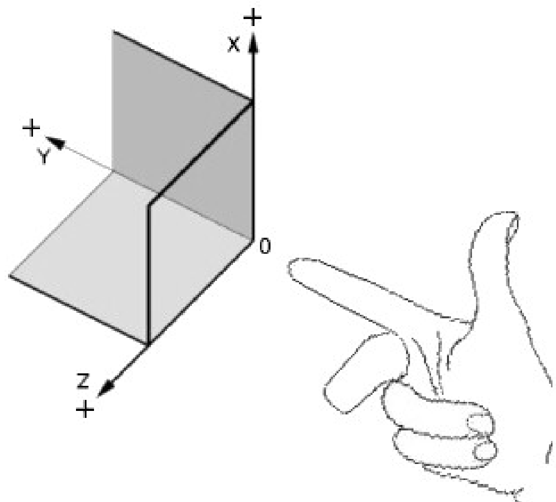

Figure 12: Right Hand Rule Coordinate Planes

For Rotary axes, the right hand rule defines the direction of rotary movement by wrapping one's right-hand fingers around the axis of rotation. Clockwise rotation points the thumb toward the person, and counterclockwise rotation points the thumb away. The thumb indicates in the positive direction of the vector or axis the hand encircles. All rotational angles and movements are given according to the right hand rule for Rotary axes.

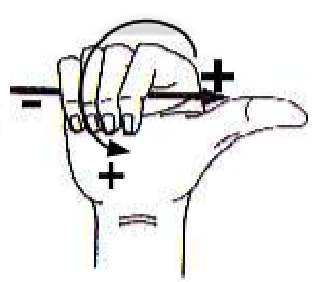

Figure 13: Rotational Right Hand Rule

B.1. Vertical Three Axis Mill

This is a simple machine tool with a vertical spindle and a table that can move in two dimensions. The modeling always starts with the `Linear` Z axis that is aligned with the primary spindle. The X axis is defined as the longest axis perpendicular to the Z axis. The spindle is now defined as a `Rotary` C axis that rotates around the Z axis.

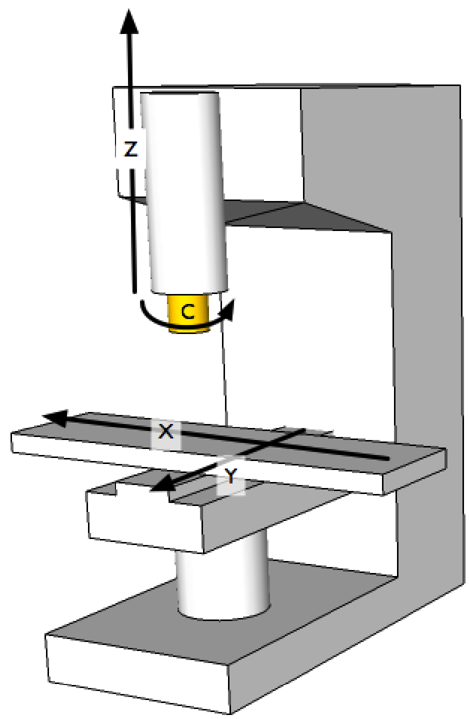

Figure 14: Three Axis Mill

The right hand rule applies when naming the axes and defining positive motion and rotation. In this case the `Rotary` axis only operate as a spindle, so it will have a constant valued `DataItem` called `RotaryMode`. This machine is only capable of executing a single program and therefore only capable of a single path. The following XML describes a simple configuration for this machine.

```xml
<?xml version="1.0" encoding="UTF-8"?>
 <MTConnectDevices xmlns="urn:mtconnect.com:MTConnectDevices:1.1"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:mtconnect.com:MTConnectDevices:1.1
    MTConnectDevices.xsd">
  <Header bufferSize="130000" instanceId="1" creationTime="2009-11-
     13T02:31:40" sender="local" version="1.1"/>
  <Devices>
   <Device id="d1" uuid="HM1" name="HMC_3Axis">
   <Description>3 Axis Mill</Description>
   <Components>
    <Axes id="a" name="base">
     <Components>
```

```xml
          <Linear id="y" name="Y">
           <DataItems>
            <DataItem type="POSITION" subType="ACTUAL" id="yp"
               category="SAMPLE" name="Yact" units="MILLIMETER"
               nativeUnits="MILLIMETER" coordinateSystem="MACHINE"/>
           </DataItems>
          </Linear>
          <Linear id="x" name="X">
           <DataItems>
            <DataItem type="POSITION" subType="ACTUAL" id="xp"
               category="SAMPLE" name="Xact" units="MILLIMETER"
               nativeUnits="MILLIMETER" coordinateSystem="MACHINE"/>
           </DataItems>
          </Linear>
          <Linear id="z" name="Z">
           <DataItems>
            <DataItem type="POSITION" id="zp" category="SAMPLE" name="Zact"
               subType="ACTUAL" units="MILLIMETER" nativeUnits="MILLIMETER"
               coordinateSystem="MACHINE"/>
           </DataItems>
          </Linear>
          <Rotary id="c" name="C">
           <DataItems>
            <DataItem type="ROTARY_VELOCITY" id="cspd" category="SAMPLE"
               name="Sspeed" subType="ACTUAL" units="REVOLUTION/MINUTE"
               nativeUnits="REVOLUTION/MINUTE"/>
            <DataItem type=" ROTARY_VELOCITY " id="cso" category="SAMPLE"
               name="Sovr" subType="OVERRIDE" units="PERCENT"
               nativeUnits="PERCENT"/>
            <DataItem type="ROTARY_MODE" id="rf" category="EVENT"
               name="rfunc">
             <Constraints>
              <Value>SPINDLE</Value>
             </Constraints>
            </DataItem>
           </DataItems>
          </Rotary>
         </Components>
        </Axes>
        <Controller id="cont" name="controller">
         <Components>
          <Path id="path" name="path">
           <DataItems>
            <DataItem type="PROGRAM" id="pgm" category="EVENT"
               name="program"/>
            <DataItem type="BLOCK" id="blk" category="EVENT" name="block"/>
            <DataItem type="LINE" id="ln" category="EVENT" name="line"/>
            <DataItem type="PATH_FEEDRATE" id="pf" category="SAMPLE"
               name="Fact" units="MILLIMETER/SECOND"
               nativeUnits="FOOT/MINUTE" subType="ACTUAL"/>
            <DataItem type="PATH_FEEDRATE" id="pfo" category="SAMPLE"
               name="Fovr" units="PERCENT" nativeUnits="PERCENT"
               subType="OVERRIDE"/>
            <DataItem type="PATH_POSITION" id="pp" category="SAMPLE"
               name="Ppos" units="MILLIMETER_3D" nativeUnits="FOOT_3D"
               coordinateSystem="WORK"/>
            <DataItem type="EXECUTION" id="exec" category="EVENT"
```

```
                name="execution"/>
             <DataItem type="CONTROLLER_MODE" id="cm" category="EVENT"
                name="mode"/>
            </DataItems>
           </Path>
          </Components>
         </Controller>
        </Components>
       </Device>
      </Devices>
     </MTConnectDevices>
```

B.2. Two Axis Lathe

The next machine is a simple two axis horizontal lathe with a Z and an X axis where the Linear Z axis is aligned with the primary spindle `Rotary` C. The material is now held in the C axis and the tool is fixed.

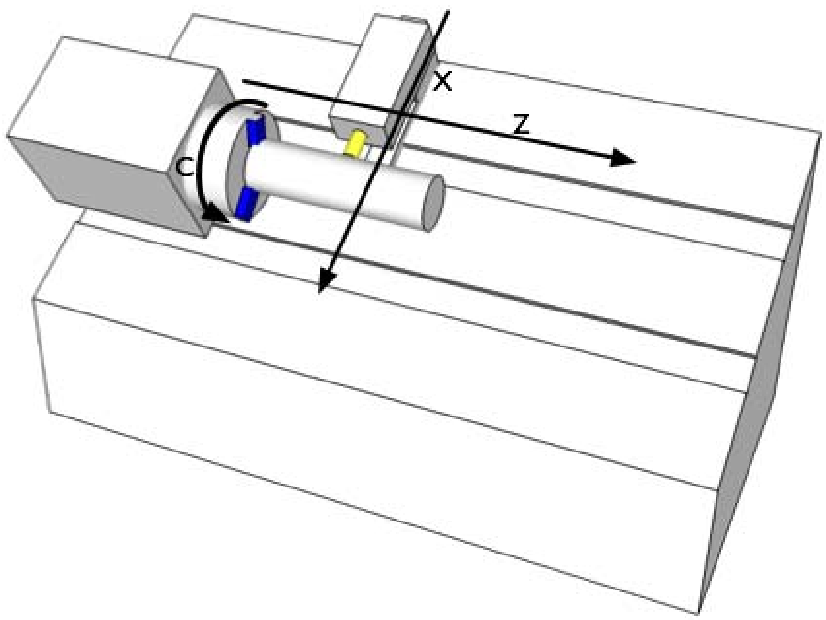

Figure 15: Two Axis Lathe

```
<?xml version="1.0" encoding="UTF-8"?>
 <MTConnectDevices xmlns="urn:mtconnect.com:MTConnectDevices:1.1"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="urn:mtconnect.com:MTConnectDevices:1.1
     MTConnectDevices.xsd">
   <Header bufferSize="130000" instanceId="1"
      creationTime="2009-11-13T02:31:40" sender="local" version="1.1"/>
   <Devices>
     <Device id="d1" uuid="HM1" name="HMC_3Axis">
    <Description>3 Axis Mill</Description>
    <Components>
     <Axes id="a" name="base">
      <Components>
       <Linear id="x" name="X">
        <DataItems>
         <DataItem type="POSITION" subType="ACTUAL" id="xp" category="SAMPLE"
            name="Xact" units="MILLIMETER" nativeUnits="MILLIMETER"
            coordinateSystem="MACHINE"/>
        </DataItems>
       </Linear>
       <Linear id="z" name="Z">
        <DataItems>
```

```xml
         <DataItem type="POSITION" id="zp" category="SAMPLE" name="Zact"
           subType="ACTUAL" units="MILLIMETER" nativeUnits="MILLIMETER"
           coordinateSystem="MACHINE"/>
        </DataItems>
       </Linear>
       <Rotary id="c" name="C">
        <DataItems>
         <DataItem type=" ROTARY_VELOCITY " id="cspd" category="SAMPLE"
           name="Sspeed" subType="ACTUAL" units="REVOLUTION/MINUTE"
           nativeUnits="REVOLUTION/MINUTE"/>
         <DataItem type=" ROTARY_VELOCITY " id="cso" category="SAMPLE"
           name="Sovr" subType="OVERRIDE" units="PERCENT"
           nativeUnits="PERCENT"/>
         <DataItem type="ROTARY_MODE" id="rf" category="EVENT" name="rfunc">
          <Constraints>
           <Value>SPINDLE</Value>
           <Value>INDEX</Value>
          </Constraints>
         </DataItem>
        </DataItems>
       </Rotary>
      </Components>
     </Axes>
     <Controller id="cont" name="controller">
      <Components>
       <Path id="path" name="path">
        <DataItems>
         <DataItem type="PROGRAM" id="pgm" category="EVENT" name="program"/>
         <DataItem type="BLOCK" id="blk" category="EVENT" name="block"/>
         <DataItem type="LINE" id="ln" category="EVENT" name="line"/>
         <DataItem type="PATH_FEEDRATE" id="pf" category="SAMPLE" name="Fact"
           units="MILLIMETER/SECOND" nativeUnits="FOOT/MINUTE"
           subType="ACTUAL"/>
         <DataItem type="PATH_FEEDRATE" id="pfo" category="SAMPLE"
           name="Fovr" units="PERCENT" nativeUnits="PERCENT"
           subType="OVERRIDE"/>
         <DataItem type="PATH_POSITION" id="pp" category="SAMPLE" name="Ppos"
           units="MILLIMETER_3D" nativeUnits="FOOT_3D"
           coordinateSystem="WORK"/>
         <DataItem type="EXECUTION" id="exec" category="EVENT"
           name="execution"/>
         <DataItem type="CONTROLLER_MODE" id="cm" category="EVENT"
           name="mode"/>
        </DataItems>
       </Path>
      </Components>
     </Controller>
    </Components>
   </Device>
  </Devices>
</MTConnectDevices>
```

B.3. HyperQuadrex

Mazak - HyperQuadrex

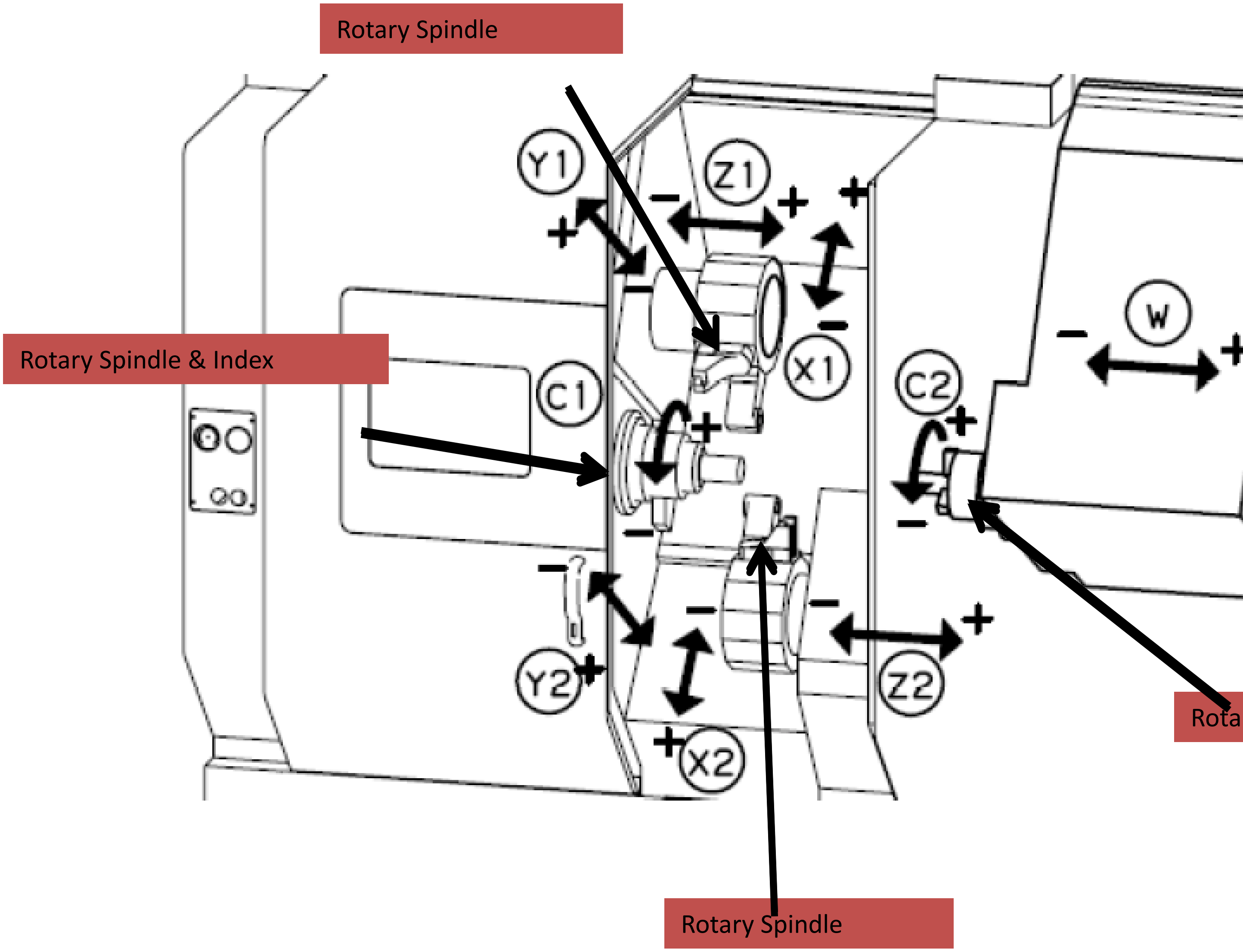

Figure 16: HyperQuadrex Lathe

```xml
<?xml version="1.0" encoding="UTF-8"?>
<MTConnectDevices xmlns="urn:mtconnect.com:MTConnectDevices:1.1"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="urn:mtconnect.com:MTConnectDevices:
   1.1 ../MTConnectDevices.xsd">
 <Header bufferSize="130000" instanceId="1" creationTime="
     2009-11-13T02:31:40" sender="local" version="1.1"/>
 <Devices>
```

```xml
<Device id="d1" uuid="HM1" name="HyperQuadrex">
 <Description>Mazak - HyperQuadrex</Description>
 <Components>
  <Axes id="a" name="base">
   <Components>
    <Linear id="x" name="X" nativeName="X1">
     <DataItems>
      <DataItem type="POSITION" subType="ACTUAL" id="xp" category="SAMPLE"
         name="Xact" units="MILLIMETER" nativeUnits="MILLIMETER"
         coordinateSystem="MACHINE">
       <Source>X1pos</Source>
      </DataItem>
      <DataItem type="LOAD" id="xl" category="SAMPLE" name="Xload"
         units="PERCENT">
       <Source>X1load</Source>
      </DataItem>
     </DataItems>
    </Linear>
    <Linear id="y" name="Y" nativeName="Y1">
     <DataItems>
      <DataItem type="POSITION" subType="ACTUAL" id="yp" category="SAMPLE"
         name="Yact" units="MILLIMETER" nativeUnits="MILLIMETER"
         coordinateSystem="MACHINE">
       <Source>Y1pos</Source>
      </DataItem>
      <DataItem type="LOAD" id="yl" category="SAMPLE" name="Yload"
         units="PERCENT">
       <Source>Y1load</Source>
      </DataItem>
     </DataItems>
    </Linear>
    <Linear id="z" name="Z" nativeName="Z1">
     <DataItems>
      <DataItem type="POSITION" id="zp" category="SAMPLE" name="Zact"
         subType="ACTUAL" units="MILLIMETER" nativeUnits="MILLIMETER"
         coordinateSystem="MACHINE">
       <Source>Z1pos</Source>
      </DataItem>
      <DataItem type="LOAD" id="zl" category="SAMPLE" name="Zload"
         units="PERCENT">
       <Source>Z1load</Source>
      </DataItem>
     </DataItems>
    </Linear>
    <Linear id="x2" name="X2" >
     <DataItems>
      <DataItem type="POSITION" subType="ACTUAL" id="x2p"
         category="SAMPLE" name="X2act" units="MILLIMETER"
         nativeUnits="MILLIMETER" coordinateSystem="MACHINE"/>
      <DataItem type="LOAD" id="x2l" category="SAMPLE" name="X2load"
         units="PERCENT">
       <Source>X2load</Source>
      </DataItem>
     </DataItems>
    </Linear>
    <Linear id="y2" name="Y2">
     <DataItems>
```

```
          <DataItem type="POSITION" subType="ACTUAL" id="y2p"
             category="SAMPLE" name="Y2act" units="MILLIMETER"
             nativeUnits="MILLIMETER" coordinateSystem="MACHINE"/>
          <DataItem type="LOAD" id="y2l" category="SAMPLE" name="Y2load"
             units="PERCENT"/>
         </DataItems>
        </Linear>
        <Linear id="z2" name="Z2">
         <DataItems>
```

```
          <DataItem type="POSITION" id="z2p" category="SAMPLE" name="Z2act"
             subType="ACTUAL" units="MILLIMETER" nativeUnits="MILLIMETER"
             coordinateSystem="MACHINE">
           <Source>Z2pos</Source>
          </DataItem>
          <DataItem type="LOAD" id="z2l" category="SAMPLE" name="Z2load"
             units="PERCENT"/>
         </DataItems>
        </Linear>
        <Linear id="z3" name="Z3" nativeName="W">
         <DataItems>
          <DataItem type="POSITION" id="z3p" category="SAMPLE" name="Z3act"
             subType="ACTUAL" units="MILLIMETER" nativeUnits="MILLIMETER"
             coordinateSystem="MACHINE">
           <Source>Wpos</Source>
          </DataItem>
          <DataItem type="LOAD" id="z3l" category="SAMPLE" name="Z3load"
             units="PERCENT">
           <Source>Wload</Source>
          </DataItem>
         </DataItems>
        </Linear>
        <Rotary id="c" name="C " nativeName="C1">
         <DataItems>
          <DataItem type="LOAD" id="Cl" category="SAMPLE" name="Cload"
             units="PERCENT"/>
          <DataItem type=" ROTARY_VELOCITY " id="cspd" category="SAMPLE"
             name="Sspeed" subType="ACTUAL" units="REVOLUTION/MINUTE"
             nativeUnits="REVOLUTION/MINUTE"/>
          <DataItem type=" ROTARY_VELOCITY " id="cso" category="SAMPLE"
             name="Sovr" subType="OVERRIDE" units="PERCENT"
             nativeUnits="PERCENT"/>
          <DataItem type="DIRECTION" id="cdir" category="EVENT" name="Sdir"/>
          <DataItem type="ANGLE" id="cpos" category="SAMPLE" name="Cpos"
             subType="ACTUAL" units="DEGREE" nativeUnits="DEGREE"
             nativeScale="-1.0"/>
          <DataItem type="ROTARY_MODE" id="rf" category="EVENT" name="rfunc">
           <Constraints>
            <Value>SPINDLE</Value>
            <Value>INDEX</Value>
           </Constraints>
          </DataItem>
         </DataItems>
        </Rotary>
        <Rotary id="c2" name="C2">
         <DataItems>
          <DataItem type="LOAD" id="C2l" category="SAMPLE" name="C2load"
             units="PERCENT"/>
          <DataItem type=" ROTARY_VELOCITY " id="c2spd" category="SAMPLE"
             name="Sspeed" subType="ACTUAL" units="REVOLUTION/MINUTE"
             nativeUnits="REVOLUTION/MINUTE"/>
          <DataItem type=" ROTARY_VELOCITY " id="c2so" category="SAMPLE"
             name="Sovr" subType="OVERRIDE" units="PERCENT"
             nativeUnits="PERCENT"/>
          <DataItem type="DIRECTION" id="c2dir" category="EVENT"
             name="S2dir"/>
```

```xml
              <DataItem type="ROTARY_MODE" id="rf2" category="EVENT" name="rfunc">
                <Constraints>
                  <Value>SPINDLE</Value>
                </Constraints>
              </DataItem>
            </DataItems>
          </Rotary>
          <Rotary id="b" name="B" nativeName="S1">
            <DataItems>
              <DataItem type="LOAD" id="bl" category="SAMPLE" name="Bload"
                  units="PERCENT"/>
              <DataItem type=" ROTARY_VELOCITY " id="bspd" category="SAMPLE"
                  name="Sspeed" subType="ACTUAL" units="REVOLUTION/MINUTE"
                  nativeUnits="REVOLUTION/MINUTE"/>
              <DataItem type=" ROTARY_VELOCITY " id="bso" category="SAMPLE"
                  name="Sovr" subType="OVERRIDE" units="PERCENT"
                  nativeUnits="PERCENT"/>
              <DataItem type="DIRECTION" id="bdir" category="EVENT" name="S3dir"/>
              <DataItem type="ROTARY_MODE" id="brf" category="EVENT" name="rfunc">
                <Constraints>
                  <Value>SPINDLE</Value>
                </Constraints>
              </DataItem>
            </DataItems>
          </Rotary>
          <Rotary id="b2" name="B2" nativeName="S2">
            <DataItems>
              <DataItem type="LOAD" id="b2l" category="SAMPLE" name="B2load"
                  units="PERCENT"/>
              <DataItem type=" ROTARY_VELOCITY " id="b2spd" category="SAMPLE"
                  name="Sspeed" subType="ACTUAL" units="REVOLUTION/MINUTE"
                  nativeUnits="REVOLUTION/MINUTE"/>
              <DataItem type=" ROTARY_VELOCITY " id="b2so" category="SAMPLE"
                  name="Sovr" subType="OVERRIDE" units="PERCENT"
                  nativeUnits="PERCENT"/>
              <DataItem type="DIRECTION" id="b2dir" category="EVENT"
                  name="S3dir"/>
              <DataItem type="ROTARY_MODE" id="b2rf" category="EVENT"
                  name="rfunc">
                <Constraints>
                  <Value>SPINDLE</Value>
                </Constraints>
              </DataItem>
            </DataItems>
          </Rotary>
        </Components>
      </Axes>
      <Controller id="cont" name="controller">
        <Components>
          <Path id="path1" name="path1">
            <DataItems>
              <DataItem type="ACTIVE_AXES" category="EVENT" name="axes"
                  id="act_axes1"/>
              <DataItem type="PROGRAM" id="pgm1" category="EVENT" name="program"/>
              <DataItem type="BLOCK" id="blk1" category="EVENT" name="block"/>
              <DataItem type="LINE" id="ln1" category="EVENT" name="line"/>
```

```xml
              <DataItem type="PATH_FEEDRATE" id="pf1" category="SAMPLE"
                  name="Fact"  units="MILLIMETER/SECOND" nativeUnits="FOOT/MINUTE"
                  subType="ACTUAL" coordinateSystem="WORK"/>
              <DataItem type="PATH_FEEDRATE" id="pfo1" category="SAMPLE"
                  name="Fovr" units="PERCENT" nativeUnits="PERCENT"
                  subType="OVERRIDE"/>
              <DataItem type="PATH_POSITION" id="pp1" category="SAMPLE"
                  name="Ppos" units="MILLIMETER_3D" nativeUnits="MILLIMETER_3D"
                  coordinateSystem="WORK"/>
              <DataItem type="TOOL_ASSET_ID" id="tid1" category="EVENT"
                  name="Tid"/>
              <DataItem type="PART_ID" id="pid1" category="EVENT" name="Pid"/>
              <DataItem type="EXECUTION" id="exec1" category="EVENT"
                  name="execution"/>
              <DataItem type="CONTROLLER_MODE" id="cm1" category="EVENT"
                  name="mode"/>
            </DataItems>
          </Path>
          <Path id="path2" name="path2">
            <DataItems>
              <DataItem type="ACTIVE_AXES" category="EVENT" name="axes"
                  id="act_axes2"/>
              <DataItem type="PROGRAM" id="pgm2" category="EVENT" name="program"/>
              <DataItem type="BLOCK" id="blk2" category="EVENT" name="block"/>
              <DataItem type="LINE" id="ln2" category="EVENT" name="line"/>
              <DataItem type="PATH_FEEDRATE" id="pf2" category="SAMPLE"
                  name="Fact" units="MILLIMETER/SECOND" nativeUnits="FOOT/MINUTE"
                  subType="ACTUAL" coordinateSystem="WORK"/>
              <DataItem type="PATH_FEEDRATE" id="pfo2" category="SAMPLE"
                  name="Fovr" units="PERCENT" nativeUnits="PERCENT"
                  subType="OVERRIDE"/>
              <DataItem type="PATH_POSITION" id="pp2" category="SAMPLE"
                  name="Ppos" units=" MILLIMETER_3D" nativeUnits=" MILLIMETER_3D"
                  coordinateSystem="WORK"/>
              <DataItem type="TOOL_ASSET_ID" id="tid2" category="EVENT"
                  name="Tid"/>
              <DataItem type="PART_ID" id="pid2" category="EVENT" name="Pid"/>
              <DataItem type="EXECUTION" id="exec2" category="EVENT"
                  name="execution"/>
              <DataItem type="CONTROLLER_MODE" id="cm2" category="EVENT"
                  name="mode"/>
            </DataItems>
          </Path>
        </Components>
      </Controller>
      <Door id="d" name="door">
        <DataItems>
          <DataItem id="ds" category="EVENT" name="door" type="DOOR_STATE"/>
        </DataItems>
      </Door>
    </Components>
  </Device>
 </Devices>
</MTConnectDevices>
```

B.4. Sensors

Sensors are modeled with the `DataItem` types associated directly with the `Component` that is being measured. In the example below, the spindle has measurement for temperature (thermistor) and vibration (accelerometer). Additionally, the sensor unit may have its own diagnostic measurements – in this case, a temperature measurement (thermistor) to measure the health of the sensor unit.

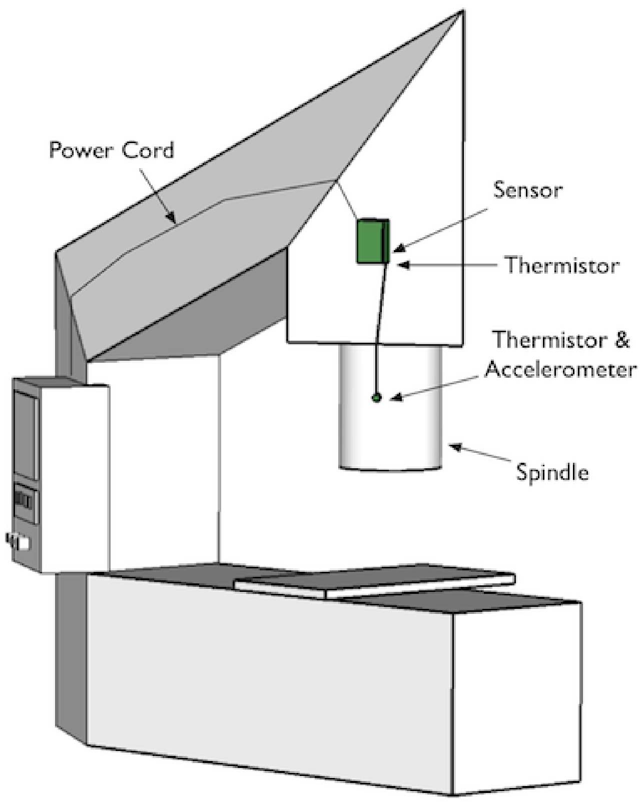

Figure 17: Spindle Sensing System

The basic machine is modeled below – 3 linear axes and a spindle. The spindle has two additional `DataItems` representing the sensors for temperature and acceleration.

```
<?xml version="1.0" encoding="UTF-8"?>
<MTConnectDevices xmlns="urn:mtconnect.org:MTConnectDevices:1.2"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="urn:mtconnect.org:MTConnectDevices:
   1.2 ../MTConnectDevices_1.2.xsd">
  <Header bufferSize="130000" instanceId="1" creationTime="
     2009-11-13T02:31:40" sender="local" version="1.2"/>
```

```xml
  <Devices>
    <Device id="d1" uuid="HM1" name="HMC_3Axis">
      <Description>3 Axis Mill</Description>
      <DataItems>
     <DataItem type="AVAILABILITY" category="EVENT" id="avail" />
      </DataItems>
      <Components>
        <Axes id="a" name="base">
          <Components>
            <Linear id="y" name="Y">
              <DataItems>
                <DataItem type="POSITION" subType="ACTUAL" id="yp"
                   category="SAMPLE" name="Yact" units="MILLIMETER"
                   nativeUnits="MILLIMETER" coordinateSystem="MACHINE"/>
              </DataItems>
            </Linear>
            <Linear id="x" name="X">
              <DataItems>
                <DataItem type="POSITION" subType="ACTUAL" id="xp"
                   category="SAMPLE" name="Xact" units="MILLIMETER"
                   nativeUnits="MILLIMETER" coordinateSystem="MACHINE"/>
              </DataItems>
            </Linear>
            <Linear id="z" name="Z">
              <DataItems>
                <DataItem type="POSITION" id="zp" category="SAMPLE"
                   name="Zact" subType="ACTUAL" units="MILLIMETER"
                   nativeUnits="MILLIMETER" coordinateSystem="MACHINE"/>
              </DataItems>
            </Linear>
            <Rotary id="c" name="C">
              <DataItems>
                <DataItem type="ROTARY_VELOCITY" id="cspd" category="SAMPLE"
                   name="Sspeed" subType="ACTUAL" units="REVOLUTION/MINUTE"
                   nativeUnits="REVOLUTION/MINUTE"/>
                <DataItem type="ROTARY_VELOCITY" id="cso" category="SAMPLE"
                   name="Sovr" subType="OVERRIDE" units="PERCENT"
                   nativeUnits="PERCENT"/>
                <DataItem type="ROTARY_MODE" id="rf" category="EVENT"
                   name="rfunc">
                  <Constraints>
                    <Value>SPINDLE</Value>
                  </Constraints>
                </DataItem>
                <DataItem type="TEMPERATURE" category="SAMPLE" name="Ctemp"
                   id="ct" units="CELSIUS" statistic="AVERAGE">
                  <Source componentId="s1">channel:1</Source>
                <DataItem type="ACCLERATION" category="SAMPLE" name="Sacc"
                   id="sa" units="MILLIMETERS/SECOND^2" statistic="MAXIMUM">
                 <Source componentId="s2">channel:2</Source>
                </DataItem>
              </DataItems>
            </Rotary>
          </Components>
        </Axes>
```

Additionally, the sensor unit is modeled with its configuration information and a `DataItem` of category `Sample` (`Voltage`) and a `DataItem` of type `Condition` (`Voltage`).

```
 <Components>
    <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 number="2" name="A/D:2">
               <Description>A/D With Accelerometer</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>
   </Components>
  </Device>
 </Devices>
</MTConnectDevices>
```