

MTConnect Standard

Part 2 – Components and Data Items  
Version 1.1.0

Prepared for: MTConnect Institute

Prepared by: William Sobel

Prepared on: October 15, 2009

MTConnect Specification

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

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

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

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

Table of Contents

1 Overview 1

1.1 MTConnect Document Structure 1

2 Purpose of This Document 2

2.1 Terminology 2

2.2 XML Terminology 4

2.3 Markup Conventions 6

2.4 Document Conventions 6

2.5 Units 7

2.6 Referenced Standards and Specifications 7

3 Devices and Components 8

3.1 Devices 9

3.2 Component 9

3.3 Component Schema 10

3.3.1 Common Component Attributes 10

3.3.2 Component Elements 11

3.4 Types of Components 12

3.4.1 Device 12

3.4.2 Axes 14

3.4.3 Controller 16

3.4.4 Power 16

4 Data Items 17

4.1 DataItem Element 18

4.1.1 Data Item Attributes 18

4.1.2 Data Item Elements 19

4.1.3 Data Item attribute: category 19

4.1.4 Data Item attribute: coordinateSystem 19

4.1.5 Data Item attribute: units 20

4.2 Types and Subtypes of Data Items 20

4.2.1 Data Item Types for SAMPLE Category 22

4.2.2 Data Item Types for EVENT Category 24

5 Component and Data Item Relationships 25

5.1 Overview 25

5.2 Device 25

5.2.1 DataItem types 25

5.2.2 Sub-components of Device 25

5.3 Common Components and Data Items 25

5.3.1 Axes 25

5.3.2 Linear 26

5.3.3 Rotary 26

5.3.4 Controller 26

5.3.5 Power 26

5.3.6 Thermostat 27

5.3.7 Vibration 27

5.4 Cutting Machine Tool Components and Data Items 27

5.4.1 Spindle 27

6 Annotated XML Examples 28

6.1 Simplest Device 28

6.2 More Complex Example of probe 29

7 Bibliography 32

Table of Figures

Figure 1: Component Schema 10

Figure 2: Example Devices Structure 8

Figure 3: Device Schema Diagram 14

Figure 4: Axes Schema Diagram 15

Figure 5: Axes Example With Three Linear Axes and one Spindle 15

Figure 6: DataItem Schema Diagram 17

# 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 industry, 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 will focus on a limited set of the characteristics mentioned above that were selected based on the fact that they can have an immediate affect on the efficiency of operations.

## 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 and Samples

Extensions to the standard will be made according to this scheme and new sections will be added as new areas are addressed. Documents will be named as follows: MTC\_Part\_<Number>\_<Description>.doc. All documents will be developed in Microsoft® Word format and released in Adobe® PDF format. For example, this document is MTC\_Part\_1\_Overview.doc.

# Purpose of This Document

This document is 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 2 of the MTConnect standard focuses on structure and description of what information is available from the device. The actual device state is not provided in this section, but is covered in Part 3 covering streams, samples, and events. The descriptive data is similar to the schema of the data, it describes the components available in this devices and what data items are provided by each component.

This part also covers instructions on how a machine tool should be modeled, the structure of the component hierarchy, the names for each component (if restricted), and allowable data items for each of the component. Some components, like Linear axis, use the naming conventions as laid out in this document. This allows for a consistent meaning across devices.

## Terminology

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

**Agent** A process that implements the MTConnect specification, acting as an interface to the device.

**Alarm** An alarm indicates an event that requires attention and indicates a deviation from normal operation.

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

**Attribute** A part of an element that provides additional information about that element. For example, the name 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 <mt:Alarm ...>This is some text</mt:Alarm>.

**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 is composed of a set of components that provide data to the application. The device is a separate entity with at least one Controller managing its operation.

**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.*

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

**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 and Events 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 interprocess 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 and samples 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 interprocess 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 element and conforms to the XML specification and schema.

## XML Terminology

In the document there will be references to XML constructs, including elements, attributes, CDATA, and more. XML consists of a hierarchy of elements. The elements can contain sub-elements, CDATA, or both. For this specification, however, an element never contains mixed content or both sub-elements and CDATA. Attributes are additional information associated with an *element*. The textual representation of an element is referred to as a *tag*. In the example:

* <Foo name=“bob”>Ack!</Foo>

an *element* consists of a named opening and closing tag. In the above example, <Foo...> is referred to as the opening tag and </Foo> is referred to as the closing tag. The text Ack! in between the opening and closing tags is called the CDATA. CDATA can be restricted to certain formats, patterns, or words. In the document when it refers to an element having CDATA, it indicates that the element has no sub-elements and only contains data.

When one looks at an XML Document there are two parts. The first part is typically referred to as an XML declaration and is only a single line. It looks something like this:

* <?xml version="1.0" encoding="UTF-8"?>

This line indicates the XML version being used and the character encoding. Though it is possible to leave this line off, it is usually considered good form to include this line in the beginning of the document. The second part contains the XML document and consists of the rest of the document.

Every XML Document contains one and only one root element. In the case of MTConnect, it is the MTConnectDevices, MTConnectStreams, or MTConnectError element. When these root elements are used in the examples, you will sometimes notice that it is prefixed with mt: as in mt:MTConnectDevices. The mt: is what is referred to as a namespace. In XML, to allow for multiple XML Schemas to be used within the same XML Document, a namespace will indicate which XML Schema is in effect for this section of the document. This convention allows for multiple XML Schemas to be used within the same XML Document, even if they have the same element names. The namespace is optional and is only required if multiple schemas are required.

An *attribute* is additional data that can be included in each XML element. For example, in the following MTConnect DataItem, there are several attributes describing the data item:

1. <DataItem name=“Xpos” type=“POSITION” subType=“ACTUAL” category=“SAMPLE” />

The name, type, subType, and category are attributes of the element. Each attribute can only occur once within an element declaration, and it can either be required or optional.

An element can have any number of sub-elements. The XML Schema specifies which sub-elements and how many times a given sub-element can occur. Here’s an example:

1. <TopLevel>
2. <FirstLevel>
3. <SecondLevel>
4. <ThirdLevel name=“first”></ThirdLevel>
5. <ThirdLevel name=“second”></ThirdLevel>
6. </SecondLevel>
7. </FirstLevel>
8. </TopLevel>

In the above example, the FirstLevel has a sub-element SecondLevel which in turn has two sub-elements, ThirdLevel, with different names. Each level is an element and its children are its sub-elements and so forth.

An XML Document can be validated. The most basic check is to make sure it is well-formed, meaning that each element has a closing tag, as in <foo>...</foo> and the document does not contain any illegal characters (<>) when not specifying a tag. If the closing </foo> was left off or an extra > was in the document, the document would not be well-formed and may be rejected by the receiver. The document can also be validated against a schema to ensure it is valid. This second level of analysis checks to make sure that required elements and attributes are present and only occur the correct number of times. A valid document must be well-formed.

All MTConnect documents must be valid and conform to the XML Schema provided along with this specification. The schema will be versioned along with this specification. The greatest possible care will be taken to make sure that the schema is backward compatible.

For more information, visit the w3c website for the XML Standards documentation: <http://www.w3.org/XML/>

## Markup Conventions

MTConnect follows industry conventions on tag format and notations when developing the XML schema. The general guidelines are as follows:

1. All tag names will be specified in Pascal case (first letter of each word is capitalized). For example: <ComponentEvents />
2. Attribute names will also be camel case, similar to Pascal case, but the first letter will be lower case. For example: <MyElement attributeName=“bob”/>
3. All values that are part of a limited or controlled vocabulary will be in upper case. For example: ON, OFF, ACTUAL, etc…
4. Dates and times will follow the W3C ISO 8601 format with arbitrary fractions of a second allowed. Refer to the following specification for details: <http://www.w3.org/TR/NOTE-datetime> The format will be YYYY-MM-DDThh:mm:ss.ffff, for example 2007-09-13T13:01.213415. The accuracy and number of fractional digits of the timestamp is determined by the capabilities of the device collecting the data. All times will be given in UTC (GMT).
5. Element names will be spelled-out and abbreviations will be avoided. The one exception is the word identifier that will be abbreviated Id. For example: SequenceNumber will be used instead of SeqNum.

## Document Conventions

The following documentation conventions will be used in the text:

* The word **MUST** is used to indicate provisions that are mandatory. Any deviation from those provisions will not be permitted.
* The word **SHOULD** is used to indicate a provision that is recommended but the exclusion of which will not invalidate the implementation.
* The word **MAY** will be used to indicate provisions that are optional and are up to the implementor to decide if they are relevant to their device.

In the tables where elements are described, the Occurrence column indicates if the attribute or sub-elements are required by the specification.

For attributes:

1. If the Occurrence is 1, the attribute **MUST** be provided.
2. If the Occurrence is 0..1, the attribute **MAY** be provided, and at most one occurrence of the attribute may be given.

For elements:

1. If the Occurrence is 1, the element **MUST** be provided.
2. If the Occurrence is 0..1, the element **MAY** be provided, and at most one occurrence of the element may be given.
3. If the Occurrence is 1..INF, one or more elements **MUST** be provided.
4. If the Occurrence is a number, e.g. 2, exactly that number of elements **MUST** be provided.

Font styles used:

Code samples as well as any XML elements or attributes will always be given in fixed width fonts. References to other *Documents* or *Sections* will be presented in italics.

## Units

MTConnect will adopt the units common to most standards specifications for exchanging data items. This will allow for greatest interoperability with other specifications. It is assumed that all MTConnect Agents will be responsible for converting the units from the native device units.

| **Property** | **Symbol** | **Unit** |
| --- | --- | --- |
| Angle |  | decimal degree |
| Angular Acceleration | /s2 | degree per second squared |
| Angular Velocity | /s | degrees per second |
| Elapsed time | s | seconds with fractions |
| Force | N | newtons |
| Length | mm | millimeters |
| Linear Acceleration | mm/s2 | millimeter per second squared |
| Linear Velocity | mm/s | millimeter per second |
| Mass | kg | Kilogram |
| Spindle Speed | rev/min | revolution per minute |
| Temperature | C | degree Celsius |
| Time | Sec | second |

Additional units will be added as needed. The decision to require the Agent to convert to the standard simplifies the applications and will provide greater interoperability and accuracy.

## Referenced Standards and Specifications

A large number of specifications are being used to normalize and harmonize the schema and the vocabulary (names of tags and attributes) specified in MTConnect *(See Bibliography for complete references).*

# Devices and Components

A device can be thought of as a group of components. For example, the Device is a three axis mill. The mill has components, one of the components is a Power component, often thought of as the main power supply. The mill also has sub-components of the Axes component; these are the three Linear axes and a Spindle. The Controller component controls the axes and runs the program. These are all sub-components of the Device.

For example, this three axis mill is modeled as a device that has a power supply, a controller, three linear axes and one spindle:

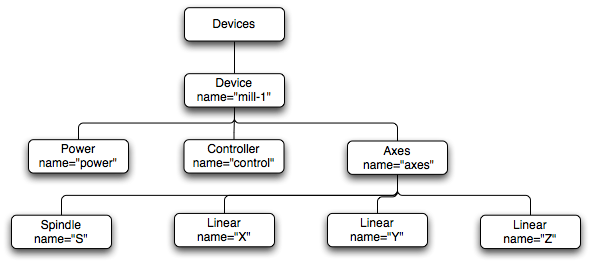


Figure : Example Devices Structure

Multiple devices may be represented in a top level container element called Devices. These container elements have no additional attributes and are only used to group sub-elements together. There are three containers used in the MTConnectDevices document. The first is the Devices element that contains all Device elements. The next container is the Components container that groups all the subcomponents together, like the Axes, Spindle, and Controller. The last container is the DataItems container that groups all data items for a component together.

In the following document structure:  
 MTConnectDevices  
 Devices  
 Device  
 Components  
 Axes  
 Components  
 Spindle [S]  
 Linear [X]  
 DataItems  
 DataItem [Xpos]  
  
 Controller  
 DataItems  
 DataItem [mode]  
 DataItem [execution]

These containers make it easier to address individual parts of the XML document. For example, if one wanted to retrieve just 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/\*.

All Devices, Components, and DataItems require an id attribute. The id attribute must adhere to the w3c standard ID-type and must be unique for the entire in an XML document. The id attributes **MUST** start with a :, \_, or letter (A-Z, a-z) and then may be followed with numbers, letters, -, or a period (.). For more information see: http://www.w3.org/TR/REC-xml/#NT-Name.

## Devices

The Devices element is a top level container for all Devices that is returned from a probe request. The probe response will only return an XML document that is a valid MTConnectDevices document.

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

## Component

The Agent needs to be capable of delivering data associated with each component to an application. The description of these pieces of information is referred to as DataItems and will be discussed in the section 4 of this document. The actual values for those data items are delivered in Streams and will be discussed in Part 3 of the standard on *Streams, Samples, and Events*.

## Component Schema

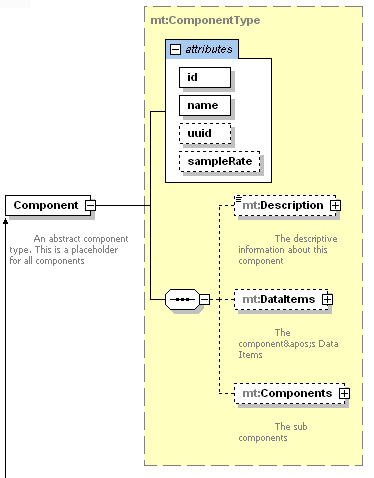


Figure : Component Schema

### Common 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 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 |
| 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 | The rate in seconds that data is obtained from the component. This is the number of milliseconds between data captures. If the sample rate is smaller than one millisecond, the number can be represented as a floating point number. For example, for one 100 microsecond sample rate would be 0.1. | 0..1\*\* |

Notes: \* The uuid **MUST** be provided for the Device, it is optional for all other components. \*\* The sampleRate is used to aid the application in interpolating values. This is the desired sample rate and may vary depending on the capabilities of the device.

### Component Elements

| **Element** | **Description** | **Occurrence** |
| --- | --- | --- |
| Description | An element that can contain any descriptive content. This can contain configuration information and manufacturer specific details. | 0..1 |
| Components | Sub-components of this component. | 0..1\* |
| DataItems | The data items this component provides. The data items are descriptions of the data events for reporting. | 0..1\* |

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

#### Description

| **Attribute** | **Description** | **Occurrence** |
| --- | --- | --- |
| manufacturer | The name of the manufacturer of the component | 0..1 |
| serialNumber | The device’s serial number | 0..1 |
| station | The station the device is located at. When a device is part of a manufacturing unit or cell with multiple stations that share the same physical controller. | 0..1 |

The CDATA of the Description is any additional descriptive information the implementor chooses to include regarding the component.

#### Components

| **Element** | **Description** | **Occurrence** |
| --- | --- | --- |
| Component | One or more components. This can also include the subtypes of Component like Axes, Linear, Power, Thermostat, etc... | 1..INF |

#### DataItems

| **Element** | **Description** | **Occurrence** |
| --- | --- | --- |
| DataItem | Only elements of types DataItem can be specified | 1..INF |

## Types of Components

All the elements in Figure 1 on page 8 are subtypes of Component. A component is an abstract type that allows for extensibility. As the specification progresses, more component types will be added, like Joint (for robotics) and Tool (for presetter).

### Device

At the top of the component tree there **MUST** be the root element Device. A Device is a container that holds all the components associated with this piece of equipment. The Device **MUST** have an alarm data item that provides a place for all Device general alarms that cannot be assigned to a sub-component.

#### Device Attributes

| **Attribute** | **Description** | **Occurrence** |
| --- | --- | --- |
| iso841Class | The ISO 841 classification for the device. | 1 |

A device **MUST** be classified using one of the following identifiers from the ISO 841 specification. The following classification is taken from the appendix of the ISO 841 specification, please use the diagram that best matches the figures in the appendix of ISO 841. If there is no diagram that matches the device, use iso841Class=”1”. Please provide us with a diagram of your device and its respective components and we will attempt to create a new classification for the device.

| **MTC ISO 841 Classification** | **Description** | **Figure** |
| --- | --- | --- |
| 1 | Other (Device not included in list) |  |
| 2 | Parallel lathe (engine lathe) | A.2 |
| 3 | Twin turret lathe with programmable tailstock | A.3 |
| 4 | Vertical turning and boring lathe | A.4 |
| 5 | Milling machine with horizontal spindle | A.5 |
| 6 | Milling machine with vertical spindle (with W axis) | A.6 |
| 7 | Boring and milling machine with horizontal spindle | A.7 |
| 8 | Milling machine with vertical spindle | A.8 |
| 9 | Portal-type milling machine | A.9 |
| 10 | Gantry-type milling machine | A.10 |
| 11 | Planer-type horizontal boring machine | A.11 |
| 12 | Profile and contouring milling machine with movable table | A.12 |
| 13 | Profile and contour milling machine with horizontal spindle | A.13 |
| 14 | Profile and contour milling machine with tilting head | A.14 |
| 15 | Profile and contour milling machine with tilting table | A.15 |
| 16 | External cylindrical grinding machine | A.16 |
| 17 | Tool and cutter grinding machine | A.17 |
| 18 | Openside planer | A.18 |
| 19 | Vertical filament winding machine | A.19 |
| 20 | Horizontal filament winding machine | A.20 |
| 21 | Flame cutting machine | A.21 |
| 22 | Punch press | A.22 |
| 23 | Drafting machine | A.23 |
| 24 | Right-hand tube bender | A.24 |
| 25 | Surface grinding machine with vertical grinding wheel | A.25 |
| 26 | Cavity sinking EDM machine | A.26 |
| 27 | Surface grinding machine | A.27 |
| 28 | Coordinate measuring machine | A.28 |
| 29 | Press brake | A.29 |
| 30 | Wire electrical discharge machine | A.30 |
| 31 | Laser cutting machine | A.31 |
| 32… | Reserved for future use. |  |

#### Device Structure

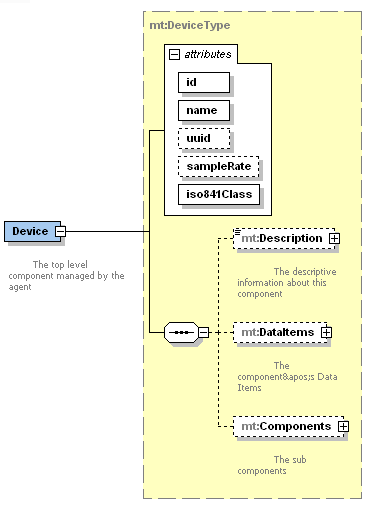


Figure : Device Schema Diagram

### Axes

There can be an arbitrary number of axes. This flexibility will accommodate the more complex multi-axis, multi-spindle machines in the future. An Axis can be one of three different types: Linear, Rotary, and Spindle. The Linear axes **MUST** be named X, Y, Z and U, V, W as defined in the ISO-841-2001 specification. Rotary axes **MUST** be named as A, B, and C and rotate around the Linear X, Y, and Z axes respectively as defined in ISO-841-2001.

When a device has an axis that serves two purposes, such as a rotational axis that can become a spindle, this **MUST** be modeled as two separate axis. The first axis will be a Spindle where the name will be “S” and the second axis will be a Rotary axis with the name “C”. At any time only one of the two axes will be active. (Note: we need to have a way to determine which is active at any given time.)

*Note:* The convention to be used for multiple linear, rotary, and spindle axes having the same designation is to index the letter with a number. For this standard the number starts at 2 (i.e. X, X2, X3, … or S, S2, S3, S4, …). This is in compliance with the ISO-841-2001. Please refer to that specification for more details.

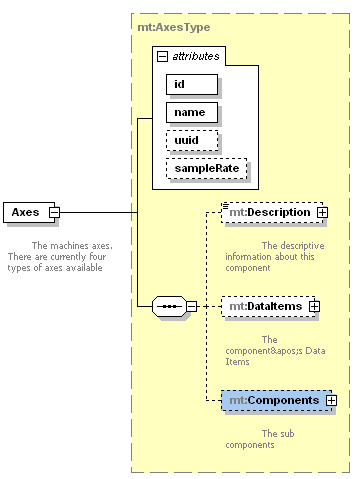


Figure : Axes Schema Diagram

The Axes component **MUST** contain at least one Axis component. The possible axis components are as follows:

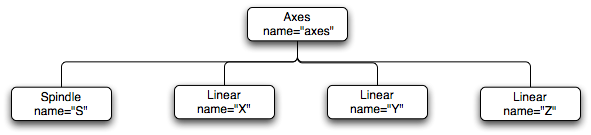


Figure : Axes Example With Three Linear Axes and one Spindle

**Linear** A linear axis moves in the direction parallel to the motion direction of a linearly moving component. Because of various errors, the direction of the linear axis can best be defined as a least-squared fit of a straight line to the appropriate straightness data.

**Rotary** An axis whose function is to provide rotary motion either for the purposes of positioning and can be used for continuous-path contour cutting in a rotary direction or for repositioning different faces of the part for the purpose of metal removal.

**Spindle** Device that provides an axis of rotation for the purpose of rapidly rotating a part or a tool to provide sufficient surface speed for cutting operations.

### Controller

The Controller component represents the CNC (Computer Numerical Control) or PAC (Programmable Automation Control) which has been referred to as a *Motion Control* or *General Purpose Motion Control*. The Control provides information regarding the execution of a control program and the execution state of the device. There are no required sub-components of the Controller.

For more complex devices and controllers, it has been considered splitting out the individual execution program for each channel. This may require multiple Control components of a single device associated with specific axes. The modeling will be deferred to later revisions of the standard.

### Power

The Power component is provided to report on the power status and possibly the voltage associated with its parent component. The device **MUST** contain a power component and **MUST** only contain the POWER\_STATUS (on/off status). Any other data items **MAY** be added. Any other component, such as a spindle, that can be switched on or off separately from the Device **SHOULD** have a power component. There are no sub-components of Power.

Power **MUST** only be set to on if the device is reachable and its power indicator is on. If the device is unreachable from the *Agent*, it **MUST** be considered OFF. OFF is defined as the power to any other component than the computer controller is disconnect from the power supply.

# Data Items

A DataItem describes a piece of information that can be collected from a component. The data item **MUST** specify the type of data being collected, the name of the data item, and the category of the item. There will only be one category for each type, but it **MUST** be included to aid the application in determining the location for the data stream. The data item **MAY** specify a Source sub-element to provide the native name for the data feed.

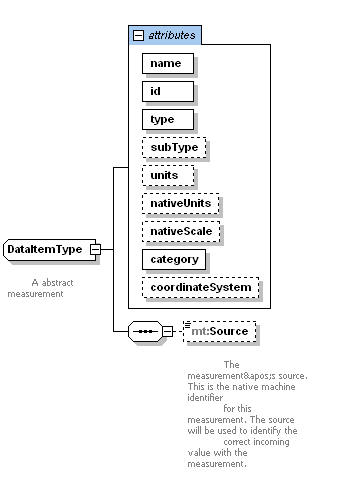


Figure : DataItem Schema Diagram

A DataItem **MAY** also specify the subType, to further qualify the type of data being requested. Subtypes are required for certain data itemsFor example, the POSITION has two subtypes: ACTUAL and COMMANDED. These are two separate data items that can be reported independently. See section 4.2.1 for a complete list of type/subtype relations.

The units **MUST** be specified for any numeric data type. The nativeUnits **MAY** 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”.

## DataItem Element

### Data Item Attributes

| **Attribute** | **Description** | **Occurrence** |
| --- | --- | --- |
| id | The unique identifier for this data item. 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 data item. A data item will have a unique name within the component. If there are multiple data items of the same type, like Position, the name will distinguish the data item. An NMTOKEN XML type. | 1 |
| type | The type of data being measured. Examples of types are POSITION, VELOCITY, ANGLE, CODE, BLOCK, SPINDLE SPEED, etc. The types are part of a controlled vocabulary that is fixed version 1.0. | 1 |
| subType | A sub-categorization of the data item type. Examples of position subtypes of POSITION are ACTUAL and COMMANDED. Not all types have subtypes and this can be left off. The subtypes are part of a controlled vocabulary that is fixed in version 1.0. | 0..1 |
| category | This is how the data item will be sampled. The two options are SAMPLE and EVENT. | 1 |
| nativeUnits | The native units used by the component. These units will be converted before they are delivered to the application. | 0..1 |
| units | The units delivered to the application. These will always be the same for this data item type. This **MUST** be specified for all numeric values. | 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 |
| 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 |
| coordinateSystem | The coordinate system being used. | 0..1 |

### Data Item Elements

| **Element** | **Description** | **Occurrence** |
| --- | --- | --- |
| Source | Source is an optional element that contains the long name of the data item if it is too complex for the name attribute. For example, if the data item has the name Xact and the axis position is delivered as Axis.channel.0.position from the device. The name attribute is Xact and the source is Axis.channel.0.position. If the source is not specified, it will be assumed to be the same as the name. | 0..1 |

### Data Item attribute: category

MTConnect provides two different categories of data items, SAMPLE and EVENT. 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 data item has a value that varies between different values in a manner that can be interpolated. A continuous value can be sampled at any point-in-time and will always produce a result. An example of a continuous data item is the S axis spindle speed. Sample data items 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 data items that have a limited number of possible values.

**EVENT** Unexpected or discrete occurrence in a component. This includes state changes and alarms. Events do not have intermediate values that differ at intermediate times, as do samples.

### Data Item attribute: coordinateSystem

A data item can specify an optional coordinate system that is being used. If not specified, the coordinates **MUST** be **MACHINE** . The possible values of coordinates are:

**MACHINE** An unchangeable coordinate system that has machine zero as its origin.

**WORK** The position that acts as the origin for a particular workpiece.

### Data Item attribute: units

| **Unit** | **Description** |
| --- | --- |
| AMPERE | Amps |
| CELCIUS | Degrees Celsius |
| COUNT | A counted event |
| DEGREE | Angle in degrees |
| DEGREE/SECOND | Degrees per second |
| DEGREE/SECOND^2 | Acceleration in degrees per second squared |
| HERTZ | Frequency measured in cycles per second |
| KILOGRAM | Kilograms |
| LITER | Liters |
| MILLIMETER | Millimeters |
| MILLIMETER/SECOND | Millimeters per second |
| MILLIMETER/SECOND^2 | Acceleration in millimeters per second squared |
| NEWTON | Force in newtons |
| PASCAL | Pressure in Newtons per square meter |
| PERCENT | Percent |
| REVOLUTION/MINUTE | Revolutions per minute |
| SECOND | A measurement of time. |
| STATUS | A status that conforms to the data item’s controlled vocabulary. Used in events to indicate states or status. |
| NEWTON\_METER | Torque, a unit for force times distance. The SI units will be used. |
| VOLT | Volts |
| WATT | Watts |

## Types and Subtypes of Data Items

What follows is the association between the various types and subtypes of data items. Each data item type **MUST** be translated into a Sample or Event with the following rules: The type name will be all in capitals with an underscore (\_) between words. The element of the event or sample will be the transformation of the data item type by capitalizing the first character of each word and then removing the underscore. For example, the data item type POWER\_STATUS is PowerStatus, POSITION is Position, and SPINDLE\_SPEED is SpindleSpeed.

An example of this transformation between the DataItem name and the Stream element is as follows:

<Controller name="Controller" id="8">

<DataItems>

<DataItem type="**LINE**" category="EVENT" id="19" subType="**ACTUAL**" name="line" />

<DataItem type="**CONTROLLER\_MODE**" category="EVENT" id="20" name="mode" />

<DataItem type="PROGRAM" category="EVENT" id="21" name="program" />

<DataItem type="EXECUTION" category="EVENT" id="22" name="execution" />

<DataItem type="BLOCK" category="EVENT" id="23" name="block" />

</DataItems>

</Controller>

The transformation from the probe *(as defined in Part 1 of the standard)* to the current or sample will occur as follows. This also illustrates how the subType is also placed in the ComponentStream as well. The probe will provide the category meaning the sub-element of the ComponentStream the items will appear in. Also note how the CONTROLLER\_MODE was changed to ControllerMode in the current request below.

<ComponentStream componentId="8" component="Controller" name="Controller">

<Events>

<**Line** dataItemId="19" timestamp="2009-03-04T19:45:50.458305" subType="**ACTUAL**" name="line" sequence="150651130">702</Line>

<Block dataItemId="23" timestamp="2009-03-04T19:45:50.458305" name="block" sequence="150651134">x0.371524 y-0.483808</Block>

<**ControllerMode** dataItemId="20" timestamp="2009-02-26T02:02:35.716224" name="mode" sequence="182">AUTOMATIC</ControllerMode>

</Events>

</ComponentStream>

### 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 |
| **ANGULAR\_ACCELERATION** | Rate of change of angular velocity. | DEGREE/SECOND^2 |
| **ANGULAR\_VELOCITY** | Rate of change of angular position. | DEGREE/SECOND |
| **AMPERAGE** | The line 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 the axis. | MILLIMETER/SECOND |
| * + ACTUAL | The single dimension feedrate. | MILLIMETER/SECOND |
| * + COMMANDED | The feedrate as specified in the program. | MILLIMETER/SECOND |
| * + OVERRIDE | The operator’s overridden value. Percent of commanded. | PERCENT |
| **DISPLACEMENT** | The displacement as measured from zero to peak | MILLIMETER |
| **FREQUENCY** | The frequency as measure in cycles per second | HERTZ |
| **GLOBAL\_POSITION** | The position in three-dimensional space. The X, Y, and Z positions will be provided. | MILLIMETER |
| * + ACTUAL | The position of the component as read from the device. | MILLIMETER |
| * + COMMANDED | The position computed by the controller. | MILLIMETER |
| **LOAD** | The load on the component. | NEWTON |
| **PATH\_FEEDRATE** | The feedrate of the tool path. | MILLIMETER/SECOND |
| * + ACTUAL | The three-dimensional feedrate derived from all components. | MILLIMETER/SECOND |
| * + COMMANDED | The feedrate as specified in the program | MILLIMETER/SECOND |
| * + OVERRIDE | The operator’s overridden value. Percent of commanded. | PERCENT |
| **PRESSURE** | The pressure on the component | PASCAL |
| **POSITION** | The position of the component. Defaults to machine coordinates. | MILLIMETER |
| * + ACTUAL | The position of the component as read from the device. | MILLIMETER |
| * + COMMANDED | The position as given by the Controller. | MILLIMETER |
| **SPINDLE\_SPEED** | The rotational speed of the spindle. | REVOLUTION/MINUTE |
| * + ACTUAL | The rotational speed the spindle is spinning at. | REVOLUTION/MINUTE |
| * + COMMANDED | The rotational speed the as specified in the program. | REVOLUTION/MINUTE |
| * + OVERRIDE | The operator’s overridden value. Percent of commanded. | PERCENT |
| **TEMPERATURE** | The temperature | CELSIUS |
| **TORQUE** | The torque | NEWTON\_METER |
| **VELOCITY** | The rate of change of position. | MILLIMETER/SECOND |
| **VOLTAGE** | The voltage | VOLT |
| **WATTAGE** | The wattage | WATT |

### Data Item Types for EVENT Category

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

| **Data Item type/subtype** | **Description** |
| --- | --- |
| **BLOCK** | The block of code being executed. The block contains the entire expression of the step in the program. |
| **CODE** | The programmatic code being executed |
| **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 producted. |
| **DIRECTION** | The rotational direction of the Axis. CLOCKWISE or COUNTER\_CLOCKWISE |
| **EXECUTION** | The execution status of the Controller. READY, ACTIVE, INTERRUPTED, or STOPPED |
| **LINE** | The current line of code being executed |
| **POWER\_STATUS** | The ON/OFF status of the component. |
| **PROGRAM** | The name of the program being executed |
| **ALARM** | An alarm is a special data item that will report any alarm for this component. An alarm **MUST** be included as a DataItem for the Device |
| **CONTROLLER\_MODE** | The current controller’s mode. AUTOMATIC, MANUAL, or  MANUAL\_DATA\_INPUT |

# Component and Data Item Relationships

This section will discuss the association between Component, DataItems, and Events and Samples. For each component, there are a limited set of allowable sub-components and a limited set of data items. For example, an Axes component may not have a Device or a Controller as a child, and it may not have as a Block DataItem type, since it is incapable of running a program.

## Overview

At the top level, a device **MUST** always contain a Power component as the main power supply. Every component that is capable of managing its own power supply, **SHOULD** have a Power sub-component. For example, a spindle **SHOULD** have a Power sub-component if it can be turned off separately from the device.

Any component **MAY** also include an arbitrary set of sensors as sub-components. The sensor is currently a placeholder for extensible data collection devices and is not modeled in this version of the specification. A sensor will be an external device that will collect data and report it to the Agent. The sensor **MUST** be correctly associated with its most relevant component. The rules governing this association will be covered in a later version of this specification.

## 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.

### DataItem types

* ALARM - An alarm placeholder for all alarms that are not associated with another component.

### Sub-components of Device

* Power
* Controller
* Axes

## Common Components and Data Items

### Axes

The Axes component serves two functions: it is a container for the actual axes as well the global data items for kinematics, path feedrate and other aggregates of all the Axis components below it. An Axes **MAY** have one or more of these:

#### DataItem types

* GLOBAL\_POSITION
* PATH\_FEEDRATE
* ACCELERATION
* VELOCITY

#### Sub-components of Axes

* Linear
* Rotary
* Spindle

### Linear

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

#### DataItem types

* POSITION
* ACCELERATION
* VELOCITY
* LOAD
* AXIS\_FEEDRATE

### Rotary

A rotary axis revolves around a point.

#### DataItem types

* ANGLE
* ANGULAR\_ACCELERATION
* ANGULAR\_VELOCITY
* LOAD
* AXIS\_FEEDRATE
* TORQUE

### 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.

#### DataItem types

* PROGRAM
* EXECUTION
* LINE
* BLOCK
* CODE
* CONTROLLER\_MODE
* PART\_COUNT

### Power

The power component represents the electrical activation of the component. The data items the power component can collect are a simple status (on/off) and three power related measurements, voltage, amperage and watts. There are no sub-components of Power. The reason for making this a separate component is the need to support legacy equipment.

For the top-level device Power component, the Power represents the power to all other components than the computer controller. Since the controller may be hosting the MTConnect *Agent*, it would be impossible to report Power OFF if the controller is off. If network or physical connectivity to the device is interrupted, the Power **MUST** be considered off.

For all other components, the definition of OFF is the component is not connected to the power source.

#### DataItem types

* POWER\_STATUS
* VOLTAGE
* AMPERAGE
* WATTS

### Thermostat

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

#### DataItem types

* TEMPERATURE

### Vibration

A sensor capable of measuring the vibration of a component.

#### DataItem types

* DISPLACEMENT
* FREQUENCY
* VELOCITY
* ACCELERATION

## Cutting Machine Tool Components and Data Items

### Spindle

The spindle is a rotational axis that revolves at high speed and has its speed expressed in REVOLUTION/MINUTE. The spindle can also have additional data items. Spindle speed has been specified as a separate data item since it receives special treatment in many applications. Velocity is used for linear axes other than spindle.

#### DataItem types

* SPINDLE\_SPEED
* LOAD
* DIRECTION
* TORQUE

# Annotated XML Examples

## Simplest Device

For the simplest possible device we are modeling a saw that has only a power status (the minimal set of components). To retrieve this information we send the following request to the Agent:

[http://10.1.23.10/ LinuxCNC/probe](http://10.1.23.10/%20LinuxCNC/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 run. For this example, the Agent does not persist the samples and events, therefore, this number will change every time. The bufferSize indicates that this Agent is capable of storing 100,000 samples and events.

1. <Devices>
2. <Device iso841Class="6" uuid="linux-01" name="LinuxCNC" sampleRate="100.0" id="1">
3. <Description manufacturer="NIST" serialNumber="01"/>

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

1. <DataItems>
2. <DataItem type="ALARM" name="alarm" category="EVENT" id="10"/>
3. </DataItems>

On line 8 we define the catch-all alarm for this device.

1. <Components>
2. <Power name="power" id="2">
3. <DataItems>
4. <DataItem type="POWER\_STATUS" name="power" category="EVENT" id="9"/>
5. </DataItems>
6. </Power>

As was stated before, the device is only required to have one Power component which **MUST** report its status. The DataItem on line 13 has an id number of 9. This will allow events responding to this data item to be easily associated. One can also see that this has been categorized as an Event and the application should expect PowerStatus in the Events collection of the ComponentStream.

1. </Components>
2. </Device>
3. </Devices>
4. </MTConnectDevices>

## 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 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="1">

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

1. <Description manufacturer="NIST" serialNumber="01"/>
2. <DataItems>
3. <DataItem type="ALARM" name="alarm" category="EVENT" id="10"/>
4. </DataItems>
5. <Components>
6. <Axes name="Axes" id="3">
7. <DataItems>
8. <DataItem type="PATH\_FEEDRATE" name="path\_feedrate" category="SAMPLE" id="11" nativeUnits="PERCENT" subType="OVERRIDE" units="PERCENT"/>
9. </DataItems>

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 data items that have a global context; they are not associated with any one data item, but they go across all axes.

1. <Components>
2. <Spindle name="S" id="7">
3. <DataItems>
4. <DataItem type="SPINDLE\_SPEED" name="Sspeed" category="SAMPLE" id="18" nativeUnits="REVOLUTION/MINUTE" subType="ACTUAL" units="REVOLUTION/MINUTE">
5. <Source>spindle\_speed</Source>
6. </DataItem>
7. <DataItem type="PRESSURE" name="Jet" id="31"/>
8. </DataItems>
9. </Spindle>

The spindle component declared on line 16 is the S axis and has spindle-specific data items.

1. <Linear name="X" id="4">
2. <DataItems>
3. <DataItem type="POSITION" name="Xact" category="SAMPLE" id="12" nativeUnits="MILLIMETER" subType="ACTUAL" units="MILLIMETER"/>
4. <DataItem type="POSITION" name="Xcom" category="SAMPLE" id="13" nativeUnits="MILLIMETER" subType="COMMANDED" units="MILLIMETER"/>
5. </DataItems>
6. </Linear>
7. <Linear name="Y" id="5">
8. <DataItems>
9. <DataItem type="POSITION" name="Yact" category="SAMPLE" id="14" nativeUnits="MILLIMETER" subType="ACTUAL" units="MILLIMETER"/>
10. <DataItem type="POSITION" name="Ycom" category="SAMPLE" id="15" nativeUnits="MILLIMETER" subType="COMMANDED" units="MILLIMETER"/>
11. </DataItems>
12. </Linear>
13. <Linear name="Z" id="6">
14. <DataItems>
15. <DataItem type="POSITION" name="Zact" category="SAMPLE" id="16" nativeUnits="MILLIMETER" subType="ACTUAL" units="MILLIMETER"/>
16. <DataItem type="POSITION" name="Zcom" category="SAMPLE" id="17" nativeUnits="MILLIMETER" subType="COMMANDED" units="MILLIMETER"/>
17. </DataItems>
18. </Linear>

Lines 24, 30, and 36 define the three linear axes X, Y, and Z respectively. In this example device the Agent is only collecting the actual and commanded positions.

1. </Components>
2. </Axes>

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

1. <Controller name="Controller" id="8">
2. <DataItems>
3. <DataItem type="LINE" name="line" category="EVENT" id="19"/>
4. <DataItem type="CONTROLLER\_MODE" name="mode" category="EVENT" id="20"/>
5. <DataItem type="PROGRAM" name="program" category="EVENT" id="21"/>
6. <DataItem type="EXECUTION" name="execution" category="EVENT" id="22"/>
7. </DataItems>
8. </Controller>
9. <Power name="power" id="2">
10. <DataItems>
11. <DataItem type="POWER\_STATUS" name="power" category="EVENT" id="9"/>
12. </DataItems>
13. </Power>
14. </Components>
15. </Device>
16. </Devices>
17. </MTConnectDevices>

# 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 B5.59-2 Version 9c: Data Specification for Properties of Machine Tools for Milling and Turning. 2005.*
13. *ASME/ANSI B5.54: Methods for Performance Evaluation of Computer Numerically Controlled Lathes and Turning Centers. 2005.*
14. OPC Foundation. *OPC Unified Architecture Specification, Part 1: Concepts Version 1.00. July 28, 2006.*