MTConnect® Standard

Part 4.0 – Assets Information Model

Version 1.8.0

Prepared for: MTConnect Institute
Prepared on: September 6, 2021

September 6, 2021

MTConnect Specification and Materials

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

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

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

This Material and all *MTConnect* Specifications and Materials are provided "as is" and *MTConnect* Institute and AMT, and each of their respective members, officers, affiliates, sponsors and agents, make no representation or warranty of any kind relating to these materials or to any implementation of the *MTConnect* Specifications or Materials in any product, including, without limitation, any expressed or implied warranty of 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 *MTConnect* Specifications or Materials for the cost of procuring substitute goods or services, lost profits, loss of use, loss of data or any incidental, consequential, indirect, special or punitive damages or other direct damages, whether under contract, tort, warranty or otherwise, arising in any way out of access, use or inability to use the *MTConnect* Specification or other *MTConnect* Materials, whether or not they had advance notice of the possibility of such damage.

Table of Contents

Table of Figures

List of Tables

1 Purpose of This Document

This document, *MTConnect Standard: Part 4.0 - Assets Information Model* of the MTConnect Standard, details information that is common to all types of *MTConnect Assets*. Part 4.0 and its sub-parts of the MTConnect Standard provide semantic models for entities that are used in the manufacturing process, but are not considered to be a piece of equipment. These entities are defined as *MTConnect Assets*. These *Assets* may be removed from a piece of equipment without detriment to the function of the equipment and can be associated with other pieces of equipment during their lifecycle. The data associated with these *Assets* may be retrieved from multiple sources that are each responsible for providing their knowledge of the *Asset*.

2 Terminology and Conventions

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

2.1 Glossary

CDATA

General meaning:

An abbreviation for Character Data.

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

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

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

Appears in the documents in the following form: CDATA

NMTOKEN

The data type for XML identifiers.

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

Appears in the documents in the following form: NMTOKEN.

XML

Stands for eXtensible Markup Language.

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

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

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

Agent

Refers to an MTConnect Agent.

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

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

Asset

item, thing or entity that has potential or actual value to an organization *Ref:ISO 55000:2014(en)*

Note 1 to entry: Value can be tangible or intangible, financial or non-financial, and includes consideration of risks and liabilities. It can be positive or negative at different stages of the asset life.

Note 2 to entry: Physical assets usually refer to equipment, inventory and properties owned by the organization. Physical assets are the opposite of intangible assets, which are non-physical assets such as leases, brands, digital assets, use rights, licences, intellectual property rights, reputation or agreements.

Note 3 to entry: A grouping of assets referred to as an asset system could also be considered as an asset.

Child Element

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

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

Component

General meaning:

A *Structural Element* that represents a physical or logical part or subpart of a piece of equipment.

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

Used in *Information Models*:

A data modeling element used to organize the data being retrieved from a piece of equipment.

- When used as an XML container to organize *Lower Level* `Component` elements.

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

- When used as an abstract XML element. `Component` is replaced in a data model by a type of *Component* element. `Component` is also an XML container used to organize *Lower Level* `Component` elements, *Data Entities*, or both.

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

Current Request

A *Current Request* is a *Request* to an *Agent* to produce an *MTConnectStreams Response Document* containing the *Observations Information Model* for a snapshot of the latest *observations* at the moment of the *Request* or at a given *sequence number*.

Data Entity

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

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

Devices Information Model

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

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

Equipment Metadata

See *Metadata*

Information Model

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

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

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

Lower Level

A nested element that is below a higher level element.

Metadata

Data that provides information about other data.

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

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

MTConnect Agent

See definition for *Agent*.

MTConnect Asset

An *MTConnect Asset* is an *Asset* used by the manufacturing process to perform tasks.

> Note 1 to entry: An *MTConnect Asset* relies upon an *MTConnect Device* to provide *observations* and information about itself and the *MTConnect Device* revises the information to reflect changes to the *MTConnect Asset* during their interaction. Examples of *MTConnect Assets* are Cutting Tools, Part Information, Manufacturing Processes, Fixtures, and Files.

> Note 2 to entry: A singular `assetId` uniquely identifies an *MTConnect Asset* throughout its lifecycle and is used to track and relate the *MTConnect Asset* to other *MTConnect Devices* and entities.

> Note 3 to entry: *MTConnect Assets* are temporally associated with a device and can be removed from the device without damage or alteration to its primary functions.

MTConnect Device

An *MTConnect Device* is a piece of equipment or a manufacturing system that produces *observations* about itself and/or publishes data using the *MTConnect Information Model*.

MTConnect Information Model

See *Information Model*

MTConnectDevices Response Document

A *Response Document* published by an *MTConnect Agent* in response to a *Probe Request*.

MTConnectStreams Response Document

A *Response Document* published by an *MTConnect Agent* in response to a *Current Request* or a *Sample Request*.

observation

The observed value of a property at a point in time.

Observations Information Model

An *Information Model* that describes the *Streaming Data* reported by a piece of equipment.

Parent Element

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

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

Probe Request

A *Probe Request* is a *Request* to an *Agent* to produce an *MTConnectDevices Response Document* containing the *Devices Information Model*.

Request

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

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

Response Document

An electronic document published by an *MTConnect Agent* in response to a *Probe Request*, *Current Request*, *Sample Request* or *Asset Request*.

Sample Request

A *Sample Request* is a *Request* to an *Agent* to produce an *MTConnectStreams Response Document* containing the *Observations Information Model* for a set of timestamped *observations* made by *Components*.

semantic data model

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

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

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

sequence number

The primary key identifier used to manage and locate a specific piece of *Streaming Data* in an *Agent*.

sequence number is a monotonically increasing number within an instance of an *Agent*.

Appears in the documents in the following form: *sequence number*.

Streaming Data

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

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

Structural Element

General meaning:

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

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

Used to indicate hierarchy of Components:

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

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

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

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

Top Level

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

Valid Data Value

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

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

XML Schema

In the MTConnect Standard, an instantiation of a schema defining a specific document encoded in XML.

2.2 Acronyms

AMT

The Association for Manufacturing Technology

2.3 MTConnect References

[MTConnect Part 1.0] *MTConnect Standard Part 1.0 - Overview and Fundamentals*. Version 1.8.0.

[MTConnect Part 3.0] *MTConnect Standard: Part 3.0 - Streams Information Model*. Version 1.8.0.

[MTConnect Part 4.0] *MTConnect Standard: Part 4.0 - Assets Information Model*. Version 1.8.0.

3 MTConnect Assets

3.1 Overview

The MTConnect Standard supports a simple distributed storage mechanism that allows applications and equipment to share and exchange complex information models in a similar way to a distributed data store. The *Asset Information Model* associates each electronic `MTConnectAssets` document with a unique identifier and allows for some predefined mechanisms to find, create, request, updated, and delete these electronic documents in a way that provides for consistency across multiple pieces of equipment.

The protocol provides a limited mechanism of accessing *MTConnect Assets* using the following properties: `assetId`, *Asset* type (element name of *Asset* root), and the piece of equipment associated with the *Asset*. These access strategies will provide the following services and answer the following questions: What *Assets* are from a particular piece of equipment? What are the *Assets* of a particular type? What *Assets* is stored for a given `assetId`?

Although these mechanisms are provided, an *Agent* should not be considered a data store or a system of reference. The *Agent* is providing an ephemeral storage capability that will temporarily manage the data for applications wishing to communicate and manage data as need-ed by the various processes. An application cannot rely on an *Agent* for long term persistence or durability since the *Agent* is only required to temporarily store the *Asset* data and may require an-other system to provide the source data upon initialization. An *Agent* is always providing the best-known equipment centric view of the data given the limitations of that piece of equipment.

3.2 MTConnectAssets

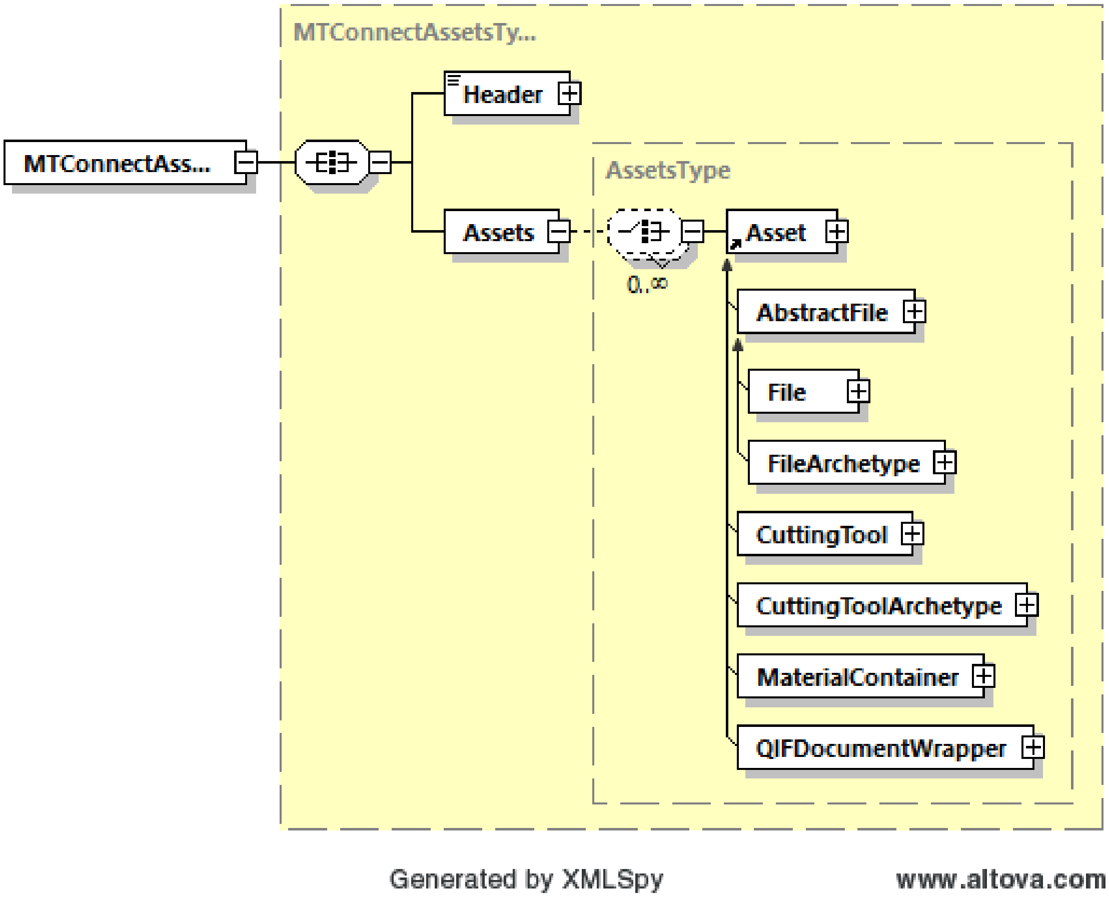

Figure 1: MTConnectAssets Schema

At the top level of the `MTConnectAssets` document is a standard header, as stated in *MTConnect Standard Part 1.0 - Overview and Fundamentals*, and one or more *MTConnect Assets*. Each *Asset* is required to have an `assetId` that serves as a unique identifier of that *Asset*. `assetId` allows an application to request the *Asset* data from an *Agent*.

In the remaining Part 4.x sub-part documents of *MTConnect Assets*, various types of *Assets* will be introduced such as cutting tools and other *Asset* types.

3.2.1 MTConnectAssets Header

The `MTConnectAssets` header is where the protocol sequence information **MUST** be provided. The *XML Schema* in *Figure 2* represents the structure of the `MTConnectAssets` header showing the attributes defined for `MTConnectAssets`.

Refer to *MTConnect Standard Part 1.0 - Overview and Fundamentals* for more information on headers.

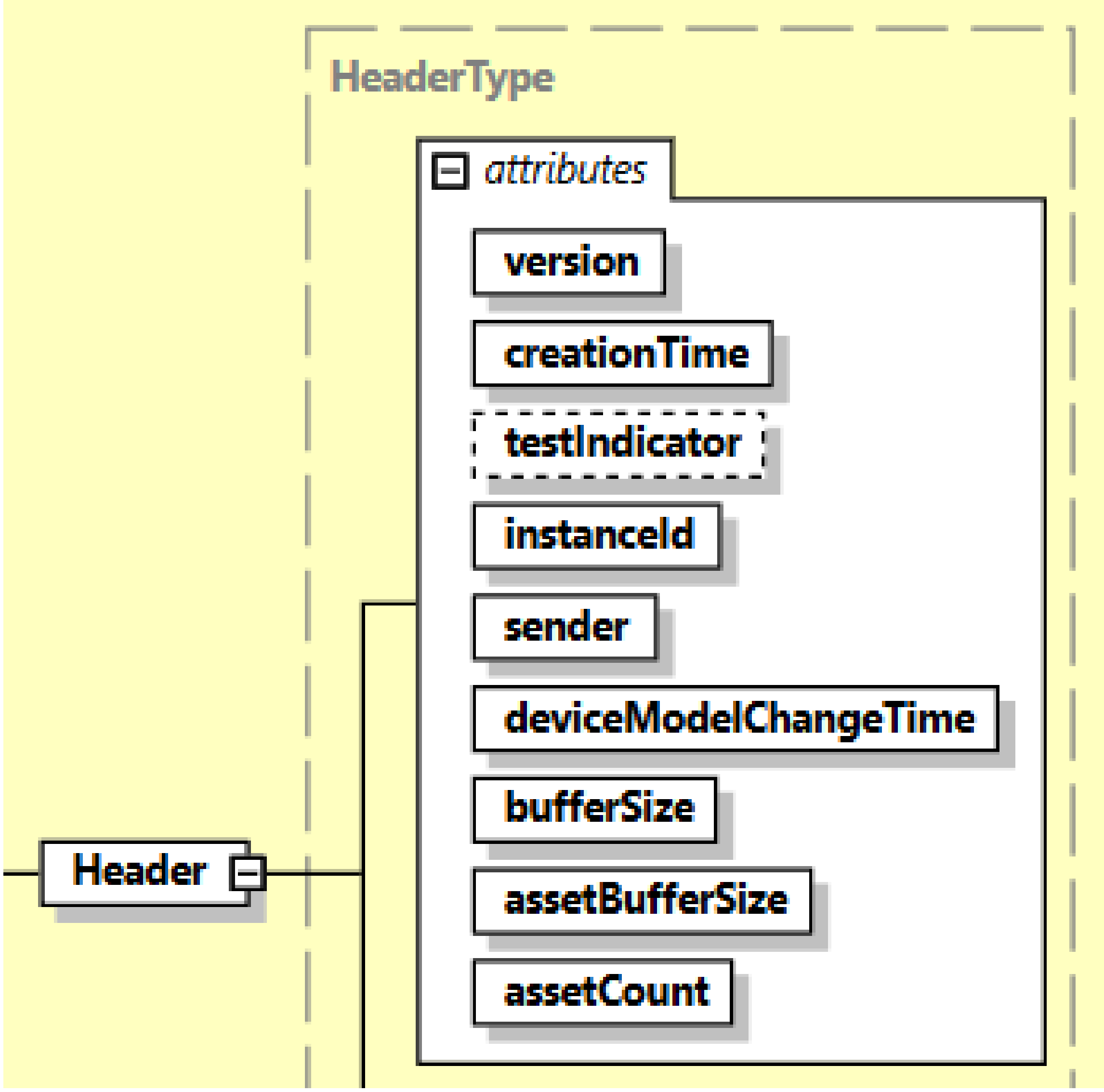

Figure 2: MTConnectAssets Header

3.2.1.1 Header Attributes

Table 1 defines the attributes used to provide information for an `MTConnectAssets` header.

Table 1: MTConnectAssets Header

Attribute	Description	Occurrence
`version`	The protocol version number. This is the *major* and *minor* version number of the MTConnect Standard being used. For example, if the version number of the Standard used is `10.21.33`, the `version` will be `10.21`. `version` is a required attribute.	1
`creationTime`	The time the response was created. `creationTime` is a required attribute.	1
`testIndicator`	Optional flag that indicates the system is operating in test mode. This data is only for testing and indicates that the data is simulated. `testIndicator` is an optional attribute.	0..1
`instanceId`	A number indicating which invocation of the *Agent*. This is used to differentiate between separate instances of the *Agent*. This value **MUST** have a maximum value of $2^{64} - 1$ and **MUST** be stored in an unsigned 64-bit integer. `instanceId` is a required attribute.	1
`sender`	The *Agent* identification information. `sender` is a required attribute.	1
`assetBufferSize`	The maximum number of *MTConnect Assets* that will be retained by the *Agent*. The `assetBufferSize` **MUST** be an unsigned positive integer value with a maximum value of $2^{32} - 1$. `assetBufferSize` is a required attribute.	1

Continuation of Table 1		
Attribute	Description	Occurrence
assetCount	The total number of *MTConnect Assets* in an*Agent*. This **MUST** be an unsigned positive integer value with a maximum value of $2^{32} - 1$. This value **MUST NOT** be greater than assetBufferSize. assetCount is a required attribute.	1
deviceModelChangeTime	A timestamp in 8601 format of the last update of the Device information for any device.	1

Example 1: MTConnectAssets Header Example

```
<Header creationTime="2010-03-13T07:59:11+00:00"
        sender="localhost" instanceId="1268463594"
        assetBufferSize="1024" version="1.1"
        assetCount="12" />
```

3.2.2 Assets

Assets is an XML container used to group information about various *MTConnect Asset* types. Assets contains one or more Asset XML elements.

Table 2: MTConnect Assets Element

Element	Description	Occurrence
Assets	An XML container that consists of one or more types of Asset XML elements.	0..1

3.2.3 Asset

An Asset XML element is a container type XML element used to organize information describing an entity that is not a piece of equipment. Asset is an abstract type XML element and will never appear directly in the MTConnect XML document. As an abstract type XML element, `Asset` will be replaced in the XML document by specific *MTConnect Asset* type.

Table 3: MTConnect Asset Element

Element	Description	Occurrence
`Asset`	An abstract XML element. Replaced in the XML document by types of `Asset` elements representing entities that are not pieces of equipment. There can be multiple types of `Asset` XML elements in the document.	1..*

There are various types of entities or *Asset* types. Each type of *Asset* is described in sub-parts of *MTConnect Standard: Part 4.0 - Assets Information Model*. These sub-parts are designated by a Part 4.x document number.

For all *MTConnect Asset* types there are some common attributes and elements that apply to all of them. The following defines these common attributes and elements.

3.2.3.1 Common Asset Attributes

The *XML Schema* in *Figure 3* represents the structure of `Asset` showing the attributes defined for `Asset`.

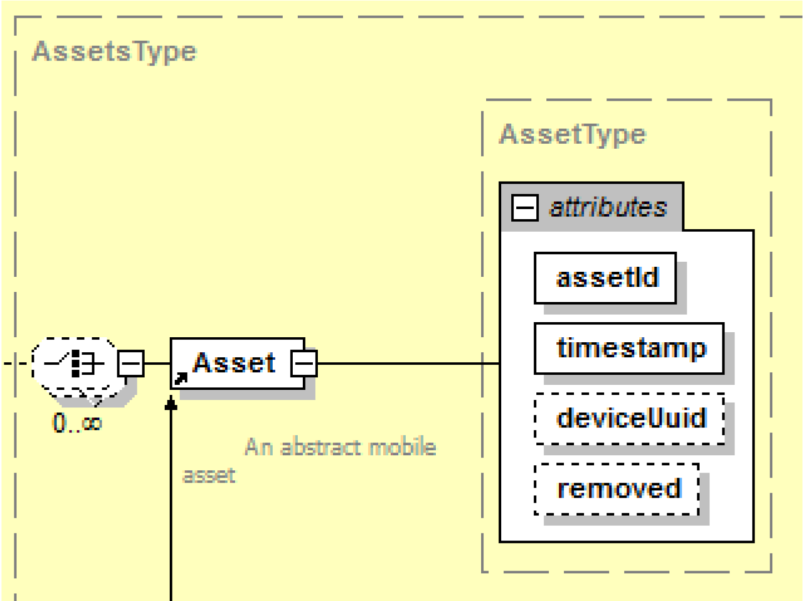

Figure 3: Asset Schema

266 *Table 4* defines the attributes that are used to provide information for the `Asset` element.

Table 4: Attributes for Asset

Attribute	Description	Occurrence
`assetId`	The unique identifier for the *MTConnect Asset*. The identifier **MUST** be unique with respect to all other *Assets* in an MTConnect installation. The identifier **SHOULD** be globally unique with respect to all other *Assets*. `assetId` is a required attribute.	1
`timestamp`	The time this *MTConnect Asset* was last modified. Always given in UTC. The `timestamp` **MUST** be provided in UTC (Universal Time Coordinate, also known as GMT). This is the time the *Asset* data was last modified. `timestamp` is a required attribute.	1

Continuation of Table 4		
Attribute	Description	Occurrence
`deviceUuid`	The piece of equipments UUID that supplied this data. This is an optional element references to the UUID attribute given in the `Device` element. This can be any series of numbers and letters as defined by the XML type NMTOKEN.	0..1
`removed`	This is an optional attribute that is an indicator that the *MTConnect Asset* has been removed from the piece of equipment. If the *Asset* is marked as removed, it will not be visible to the client application unless the=true parameter is provided in the URL. If this attribute is not present it MUST be assumed to be false. The value is an `xsi:boolean` type and **MUST** be `true` or `false`.	0..1

All *MTConnect Assets* **MUST** have a unique value for `assetId` and it **SHOULD** be globally unique, such as a RFC 4122 UUID.

The following attributes **MUST** be provided and are common to all *MTConnect Asset* types: the `assetId` attribute providing the unique identifier for the *Asset*, and the `timestamp` providing the time the *Asset* was inserted or updated. A removed flag that if `true` indicates the *Asset* has been removed (deleted) from the equipment is optional, however the *Asset* will still be available if requested directly or a request is made that includes removed *Assets*.

An `MTConnectAssets` document contains information pertaining to something that is not a direct component of the piece of equipment and can be relocated to another piece of equipment or location during its lifecycle. The `Asset` will contain data that will be changed as a unit, meaning that at any given point in time the latest version of the complete state for this *Asset* will be provided.

Each piece of equipment or location may have a different view of this *Asset* and it is the responsibility of an application to collect and determine the aggregate information and keep a historical record if required. An *Agent* will allow any application or other equipment to request this information. The piece of equipment **MUST** supply the latest and most accurate information regarding a given *Asset*.

285 #### 3.2.3.2 Common Asset Elements

286 The element Description is the only element common to all *Asset* types.

287 The *XML Schema* in *Figure 4* represents the structure of Description.

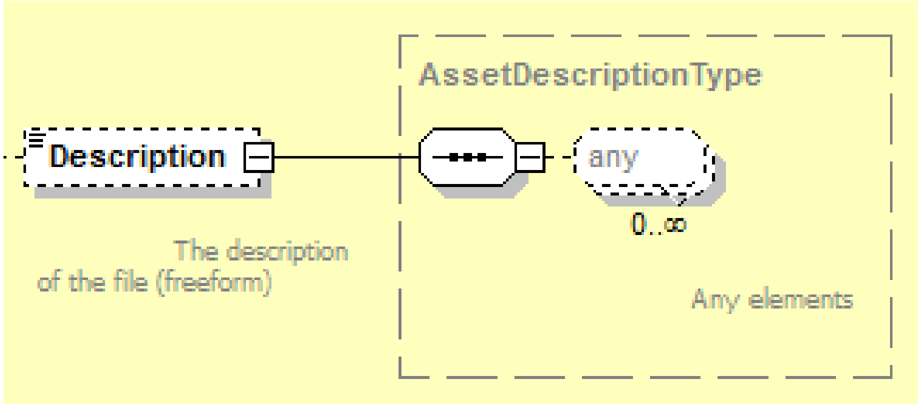

Figure 4: Description Schema

288 *Table 5* defines the elements that are used to provide information for Asset.

Table 5: Elements for Asset

Elements	Description	Occurrence
Description	An optional element that can contain any descriptive content. This can contain configuration information and manufacturer specific details. This element is defined to contain mixed content and XML elements can be added to extend the descriptive semantics of MTConnect Standard.	0..1

4 MTConnect Assets Architecture

4.1 Agent Asset Storage

The *Agent* stores *MTConnect Assets* in a similar fashion as the *Agent* data storage described in *MTConnect Standard Part 1.0 - Overview and Fundamentals*. The storage of information is contained in the *asset buffer*. The *Agent* provides a limited number of *Assets* that can be stored at one time and uses the same method of pushing out the oldest *Asset* when the *asset buffer* is full. The *asset buffer* size for the *Asset* storage is maintained separately from the `Sample`, `Event`, and `Condition` storage.

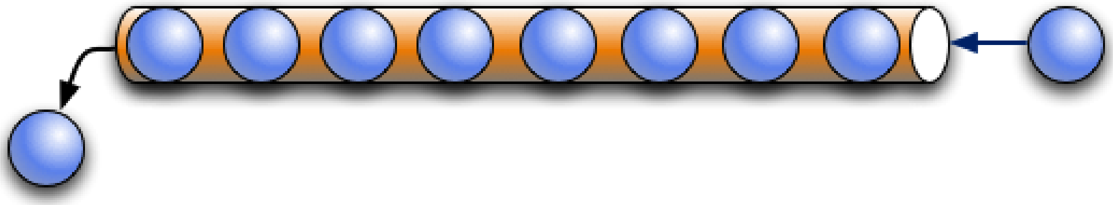

Figure 5: MTConnect Assets storage as First in First Out

MTConnect Assets also behave like a key/value in memory database. In the case of the *Asset*, the key is the `assetId` and the value is the XML document describing the *Asset*. The key can be any string of letters, punctuation or digits and represent the domain specific coding scheme for their assets. Each *Asset* type will have a recommended way to construct a unique `assetId`, for example, a cutting tool **SHOULD** be identified by the tool ID and serial number as a composed synthetic identifier.

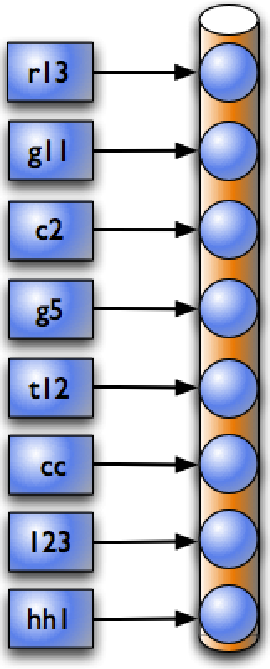

Figure 6: MTConnect Assets storage as Key/Value pairs

As in *Figure 6* , each of the *Assets* is referred to by their key. The key is independent of the order in the *asset buffer* storage.

4.2 Asset Protocol

MTConnect Standard provides methods to retrieve an *MTConnect Asset* or a set of *Assets* given various criteria. These criteria are as follows: The `assetId`, the *Asset* `type` as defined by the name of the *Asset*'s topmost element, and the originating piece of equipment.

The URL format is similar to the `probe` and `sample` structure. Reference each `assetId` directly to request an *MTConnect Asset* by `assetId`.

4.2.1 Asset by assetId

Example 2: Asset by assetId Example

```
url: http://example.com/asset/e39d23ba-ef2d-
    11e6-b12c15028cfe91a82ef
```

Example 2 returns the `MTConnectAssets` document for *Asset* `e39d23ba-ef2d-11e6-b12c-28cfe91a82ef`

Request multiple *Assets* by each `assetId`:

Example 3: Assets by assetId Example

```
url: http://example.com/asset/e39d23ba-ef2d-11e6-b12c155;
     8cfe91a82ef;e46d5256-ef2d-11e6-96aa-28cfe91a82ef
```

Example 3 returns the `MTConnectAssets` document for *Assets* `e39d23ba-ef2d-11e6-b12c-28cfe91a82ef` and `e46d5256-ef2d-11e6-96aa-28cfe91a82ef`.

Request for all the *Assets* in the *Agent*:

Example 4: Get all Assets Example

```
url: http://example.com/assets
```

Example 4 returns all available *MTConnect Assets* in the *Agent*. The *Agent* **MAY** return a limited set if there are too many *Asset* records. The *Assets* **MUST** be added to the beginning with the most recently modified *Asset*.

4.2.2 Asset for a Given Type

Example 5: Asset for a Given Type Example

```
url: http://example.com/assets?type="CuttingTool"
```

Example 5 returns all available `CuttingTool` *Assets* from the *Agent* of the type `CuttingTool`. The *Agent* **MAY** return a limited set if there are too many *Asset* records. The *Assets* **MUST** be added to the beginning with the most recently modified assets.

Request for all *Assets* of a given type in the *Agent* up to a maximum count:

Example 6: Asset for a Given Type with Maximum count Example

```
url: http://example.com/assets?type="CuttingTool"
```

Example 6 returns all available `CuttingTool` *Assets* from the *Agent*. The *Agent* **MUST** return up to 1000 *Assets* beginning with the most recently modified *Assets* if they exist.

4.2.3 Assets Including Removed Assets

Example 7: Assets Including Removed Assets Example

```
url: http://example.com/assets?type=CuttingTool&removed=true
```

Example 7 returns all available `CuttingTool` *Assets* from the *Agent*. With the removed flag, *Assets* that have been removed but are included in the result set.

4.2.4 Assets for a Piece of Equipment

If no `assetId` is provided with a general *Assets* request, it would be as shown in *Example 8*:

Example 8: Assets For a Piece of Equipment Example

```
url: http://example.com/Mill123/assets
```

All *MTConnect Assets* will be provided for that piece of equipment (`Device`) up to the *Agent*'s maximum count or as specified with the count parameter. These *Assets* will be returned starting from the newest to oldest list.

Any of the previous constraints can also be applied to the request, for example, to get all the `CuttingTool` instances for a given piece of equipment:

Example 9: Assets For a Piece of Equipment For a Given Type Example

```
url: http://example.com/Mill123/asset/
          ?type=CuttingTool&count=100
```

The request in *Example 9* will get the newest 100 Cutting Tool Instance *Assets* from the *Agent* for `Mill123`. Similarly:

Example 10: Assets For a Piece of Equipment For a Given Type Example 2

```
url: http://example.com/Mill123/asset/
          ?type=CuttingToolArchetype
```

Example 10 will provide all Cutting Tool Archetype *Assets* with the `deviceUuid` of `Mill123`.

5 Extensions to Part 2.0 - Devices Information Model

This document will add the following data item types to support change notification when an *MTConnect Asset* is added or updated. The data item **MUST** be placed in the DataItems container associated with Device. The Device **MUST** be the piece of equipment that is supplying the asset data.

5.1 Data Item Types added for EVENT Category

Table 6: DataItem Type for EVENT category

DataItem Type SubType	Description
ASSET_CHANGED	The event generated when an asset is added or changed. AssetChanged **MUST** be discrete and the value of the DataItem's discrete attribute **MUST** be true.
ASSET_REMOVED	The value of the CDATA for the event **MUST** be the assetId of the asset that has been removed. The asset will still be visible if requested with the includeRemoved parameter as described in the protocol section. When assets are removed they are not moved to the beginning of the most recently modified list.

5.1.1 ASSET_CHANGED Data Item Type

When an *MTConnect Asset* is added or modified, an AssetChanged event **MUST** be published to inform an application that new asset data is available. The application can request the new asset data from the piece of equipment at that time. Every time the asset data is modified an AssetChanged event will be published. Since the asset data is a complete electronic document, the system will publish a single AssetChanged event for the entire set of changes.

The asset data **MUST** remain constant until the AssetChanged event is published. Once it is published the data **MUST** change to reflect the new content at that instant. The timestamp of the asset will reflect the time the last change was made to the asset data.

5.1.2 ASSET_REMOVED Data Item Type

When an *MTConnect Asset* has been removed from an *Agent*, or marked as removed, an `AssetRemoved` event **MUST** be generated in a similar way to the `AssetChanged` event. The CDATA of the `AssetRemoved` event **MUST** contain the `assetId` that was just removed.

Every time an *MTConnect Asset* is modified or added it will be moved to the beginning of the *asset buffer* and become the newest *Asset*. As the *asset buffer* fills up, the oldest *Asset* will be pushed out and its information will be removed. The MTConnect Standard does not specify the maximum size of the *asset buffer*, and if the implementation desires, permanent storage **MAY** be used to store the *Assets*. A value of 4,294,967,296 or 2^{32} can be given to indicate unlimited storage.

There is no requirement for persistent *Asset* storage. If the *Agent* fails, all existing *MTConnect Assets* **MAY** be lost. It is the responsibility of the implementation to restore the lost *Asset* data and it is the responsibility of the application to persist the *Asset* data. The *Agent* **MAY** make no guarantees about availability of *Asset* data after the *Agent* stops.

6 Extensions to Part 3.0 - Streams Information Model

The associated modifications **MUST** be added to *MTConnect Standard: Part 3.0 - Streams Information Model* to add the following event to the Events in the streams.

6.1 AssetChanged Extension to Events

The AssetChanged element extends the base Event type XML data element defined in *MTConnect Standard: Part 3.0 - Streams Information Model* and adds the assetType attribute to the base Event. This new Event will signal whenever a new *MTConnect Asset* is added or the existing definition of an *Asset* is updated. The assetId is provided as the CDATA value and can be used to request the *Asset* data from the *Agent*.

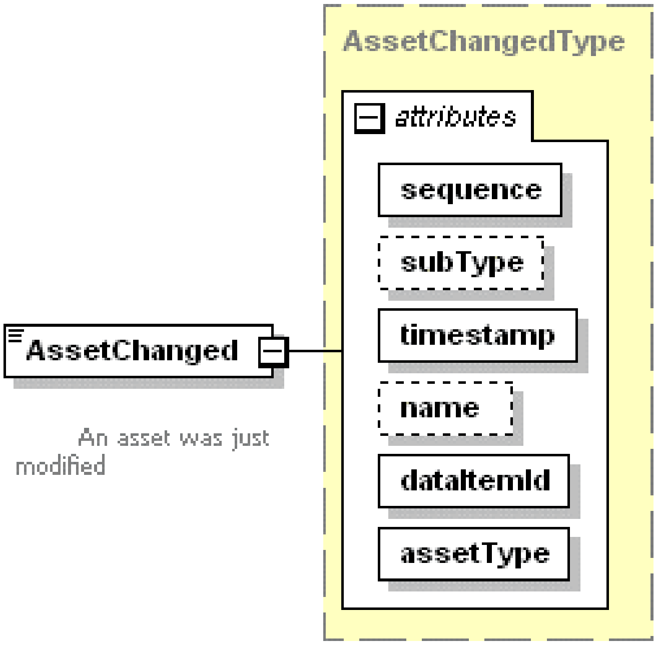

Figure 7: AssetChanged Schema

AssetChanged: An *MTConnect Asset* has been added or modified. The CDATA for the AssetChanged element **MUST** be the assetId of the *Asset* that has been modified.

6.1.1 AssetChanged event Attributes

Table 7: Attributes for AssetChanged

Attribute	Description	Occurrence
assetType	The type of asset changed. assetType is a required attribute. *Valid Data Values*: CuttingTool File QIFDocumentWrapper MaterialContainer	1

6.2 AssetRemoved Extension to Events

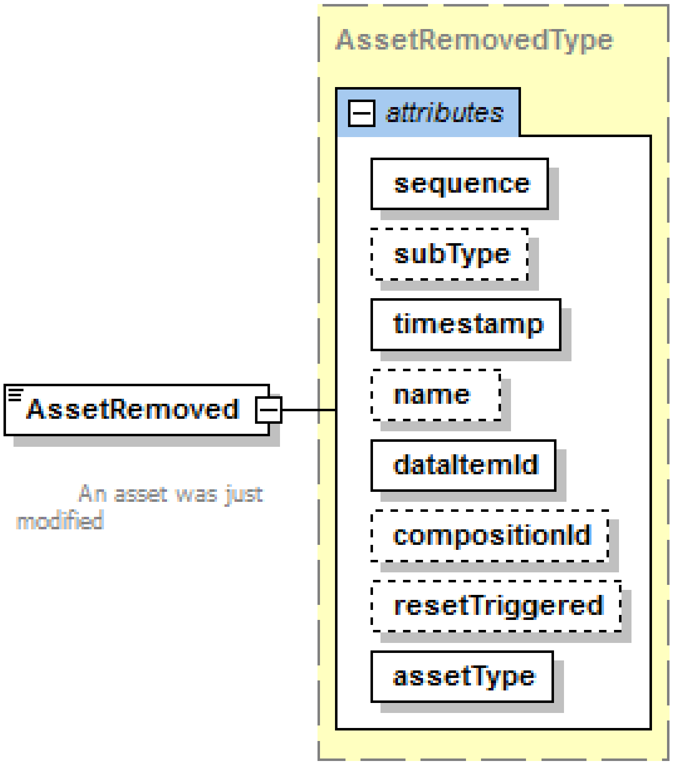

Figure 8: AssetRemoved Schema

`AssetRemoved`: An *MTConnect Asset* has been removed. The CDATA for the `AssetRemoved` element **MUST** be the `assetId` of the *Asset* that has been removed.

6.2.1 AssetRemoved Attributes

Table 8: Attributes for AssetRemoved

Attribute	Description	Occurrence

`assetType`	The type of asset that was removed.
	`assetType` is a required attribute.
	Valid Data Values:
	`CuttingTool`
	`File`
	`QIFDocumentWrapper`
	`MaterialContainer`
		1

The *MTConnect Asset* will still be available if requested if the removed=true argument is supplied. The `assetId` is provide as the CDATA value and can be used to request the *Asset* data from the *Agent*.

Appendices

A Bibliography

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

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

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

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

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

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

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

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

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

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

New York, 1984.

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

ASME B5.59-2 Version 9c: Data Specification for Properties of Machine Tools for Milling and Turning. 2005.

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

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

International Organization for Standardization. *ISO 13399: Cutting tool data representation and exchange*. Geneva, Switzerland, 2000.