AMT + VT | COLLEGE OF ENGINEERING
GRADO DEPARTMENT OF
INDUSTRIAL AND SYSTEMS ENGINEERING
VIRGINIA TECH.

DIGITAL MANUFACTURING

AN MTCONNECT USE CASE STUDY AT THE VIRGINIA TECH LEARNING FACTORY

Address

AMT

7901 Jones Branch Dr., Ste. 900,
McLean, Virginia 22102
United States

Online

Website: www.AMTonline.org

Table of Contents

MTConnect License Use

This handbook discusses the technical and developmental aspects of MTConnect as an open-source communication standard. It can be used by anyone but is specifically designed for those seeking to implement and utilize MTConnect in their manufacturing project. It discusses the advantages of MTConnect, offers implementation suggestions, and outlines future implications to guide users in the implementation process and further utilization of MTConnect.

To get started, you can navigate through the handbook using the Table of Contents (above).

Comments and feedback are welcome as MTConnect is an open-source communication standard created for all users. Please contact AMT on the MTConnect webpage through the "How can we help you?" chat provided in the bottom right-hand corner. Your contributions and/or suggestions are appreciated.

If you need more information about the MTConnect license agreement, please visit the MTConnect legal webpage.

Introduction to MTConnect

What is MTConnect?

The open-source MTConnect standard is the product of industry; specifically, the consensus-driven Standards Committee of the MTConnect Institute. It provides uniform, structured, contextualized data with no proprietary format, which allows users to implement a streamlined communication standard between machines. This eliminates the need for users to be specialized in any one particular machine software; applications that consume MTConnect data provide more efficient operations, improved production optimization, and increased productivity. Additionally, because scalable system architectures depend on standards, MTConnect provides domain-specific vocabulary and data models, is extensible, and integrates with other standards by design.

MTConnect Advantages

The high-level purpose of this project is to integrate manufacturing technology with the open-source standard, MTConnect. MTConnect streamlines data collection from different machines – for example, the Haas CNC (Computer Numerical Control) machine, a milling machine – by linking adapters and agent software to the MTConnect language. The outcome is machine data is successfully extracted from a manufacturing technology, stored in a preferred database platform, and visualized on an interactive dashboard. Possible data sources include machine tools, production equipment, sensors, and other machining hardware. This improves operations efficiency, optimizes production, and increases productivity.

Is MTConnect Right for You?

As an open standard, MTConnect is available to any and all manufacturing researchers and engineers. This document will help further identify if MTConnect can accomplish the needs of your project and/or organization.

Reference the MTConnect User's Portal and the MTConnect License Use within this document for further information.

MTConnect Overview

Requirements for Setting Up MTConnect

Setting up MTConnect in a manufacturing environment requires machines capable of interfacing with an external connection for data extraction. One of the best use cases of MTConnect is with a CNC mill because its software can generate hundreds of machine parameters. You will need a computer to serve as the external interface with the machine as well as a local network for connectivity between devices. Additionally, it is also helpful to have a second computer to view current samples of data at any given point in order to verify that parameters are successfully being extracted. Below is a summary of devices involved with the implementation covered by this handbook:

- Machine: Haas CNC VF-3 mill
- Direct External Interface: Any Windows/Mac computer with Linux capability
- Network: Local network with administrative credentials
- Additional Computer: Any Windows/Mac computer with Linux capability

Equipment

Equipment, as defined by MTConnect, is "any data source [1]." Examples of equipment can include, but are not limited to, "machine tools, ovens, sensor units, workstations, software applications, and bar feeders [1]." Ensuring that you have some way to interface with your machine is always a good idea. In other words, is it possible to extract data from the machine? If there is no way to extract data from the equipment, then implementing the MTConnect standard would be impractical. In our particular case, the equipment utilized is a Haas VF-3 3-axis CNC mill; we extracted data through its built-in serial port. Every piece of equipment is different, so be sure to consult the manufacturer's documentation to see how, or if, you can extract data from the equipment.

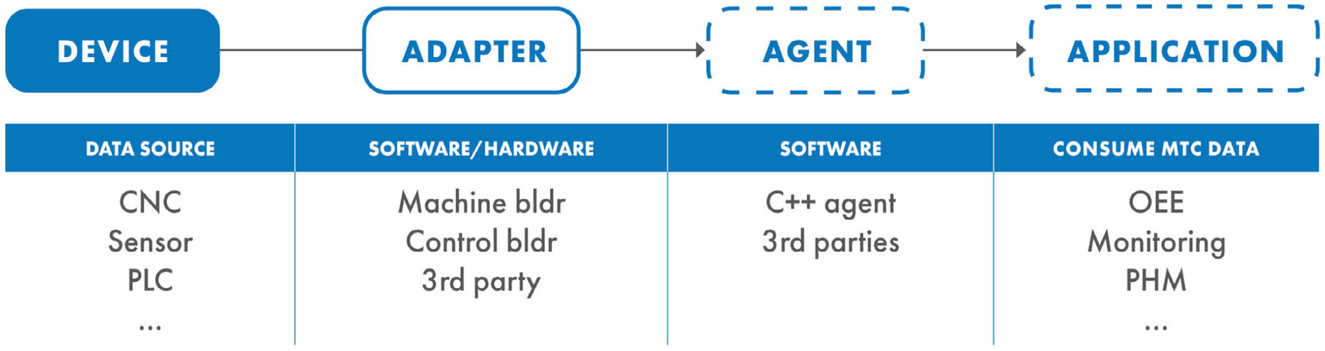


Figure 1. https://www.mtconnect.org/getting-started

Agent

An agent is software that collects all the machine data extracted by the adapter and transforms them into a computer-readable format. MTConnect documentation describes it as "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 [1]." It functions as a standalone server for hosting the display of information in an XML (eXtensible Markup Language) tree on the network.

Adapter

An adapter is software that links the machine, such as the Haas CNC mill, and the MTConnect agent. MTConnect documentation describes it as “an optional piece of hardware or software that transforms information provided by a piece of equipment into a form that can be received by an Agent [1].” Its purpose is to extract device-specific data and then act as a “translator” to stream this information to the agent to be hosted on the network. Some equipment will not need an adapter if it is able to communicate directly with the agent, but most equipment will need this piece of software to function with MTConnect.

Application

When implementing MTConnect in machines, controls, or pieces of equipment, the standard provides universal data definitions that are always the same, regardless of the manufacturer. The data is extracted from the MTConnect agent and is consumed for several software applications like machine monitoring, predictive maintenance, process analytics, etc. Applications can range anywhere between custom code to industry standard applications that allow for MTConnect plug-in and the creation of dashboards to display information such as equipment usage and performance.

Agent

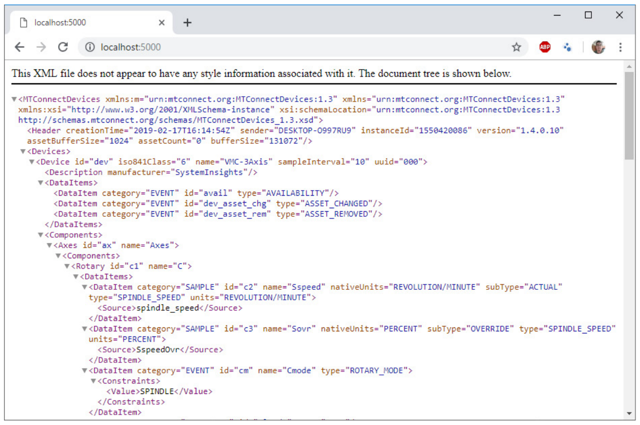
Installing Agent

Installing the adapter is a very simple process, but it depends heavily on what operating system the agent will be using. The agent files can be easily located at the MTConnect Institute GitHub page, with the C++ agent we used being specifically located in the cppagent GitHub folder. In our specific setup, we ran the agent on a Raspberry Pi with Linux Raspian so the setup process utilized steps for a Linux operating system. We utilized a step-by-step guide by Machining Code, detailing every step to get the agent running on our platform. Your mileage may vary with this guide, so if you are utilizing a different operating system, it is essential to perform research on how to install the agent.

Configuring Agent

Once you get your agent installed, you will have to configure the software. This requires you to navigate to your agent folder located within the cppagent directory. In this agent folder, there is a file called agent.cfg, which is the file where you configure various settings for your machines. Within this file, the two main areas you should focus on are the adapter section and the schema section. The adapter section focuses on the IP address and port through which your agent will look for data from an adapter. It is critical that your IP and port are the same in your adapter code and in this agent.cfg file. Next, the schema section of the agent.cfg section specifies where the agent should look for the desired XML schema for MTConnect. An XML schema can be thought of as the "skeleton" structure in which your live machine parameter values are placed when the agent and adapter are running. The agent comes with a standard XML schema out of the box, but once you start deciding what parameters and information you want in MTConnect, you will most likely want to make it more customized. Further information on XML schemas can be found here. Finally, you can find additional information on the entire agent.cfg file on the official MTConnect Github.

Verification



This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
▼<MTConnectDevices xmlns:m="urn:mtconnect.org:MTConnectDevices:1.3" xmlns="urn:mtconnect.org:MTConnectDevices:1.3"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="urn:mtconnect.org:MTConnectDevices:1.3
 http://schemas.mtconnect.org/schemas/MTConnectDevices_1.3.xsd">
   <Header creationTime="2019-02-17T16:14:54Z" sender="DESKTOP-O997RU9" instanceId="1550420086" version="1.4.0.10"
   assetBufferSize="1024" assetCount="0" bufferSize="131072"/>
  ▼<Devices>
    ▼<Device id="dev" iso841Class="6" name="VMC-3Axis" sampleInterval="10" uuid="000">
       <Description manufacturer="SystemInsights"/>
      ▼<DataItems>
         <DataItem category="EVENT" id="avail" type="AVAILABILITY"/>
         <DataItem category="EVENT" id="dev_asset_chg" type="ASSET_CHANGED"/>
         <DataItem category="EVENT" id="dev_asset_rem" type="ASSET_REMOVED"/>
       </DataItems>
      ▼<Components>
        ▼<Axes id="ax" name="Axes">
          ▼<Components>
            ▼<Rotary id="c1" name="C">
              ▼<DataItems>
                ▼<DataItem category="SAMPLE" id="c2" name="Sspeed" nativeUnits="REVOLUTION/MINUTE" subType="ACTUAL"
                 type="SPINDLE_SPEED" units="REVOLUTION/MINUTE">
                   <Source>spindle_speed</Source>
                 </DataItem>
                ▼<DataItem category="SAMPLE" id="c3" name="Sovr" nativeUnits="PERCENT" subType="OVERRIDE" type="SPINDLE_SPEED"
                 units="PERCENT">
                   <Source>SspeedOvr</Source>
                 </DataItem>
                ▼<DataItem category="EVENT" id="cm" name="Cmode" type="ROTARY_MODE">
                  ▼<Constraints>
                     <Value>SPINDLE</Value>
                   </Constraints>
                 </DataItem>
```

Figure 2. https://i0.wp.com/machiningcode.com/wp-content/uploads/2019/02/agent.png?w=960&ssl=1

Once you have completed all the setup steps, you should be able to navigate to "`localhost:5000`" in a web browser if you are using the device running the agent. If you want to verify the operation of an MTConnect agent running on another computer, enter the `<IP address of the computer running the agent>:5000` (i.e., `192.168.1.3:5000`). You should immediately see a webpage that contains XML on the screen. This XML should match the schema you supplied to the config file in the previous step. When accessing the MTConnect agent, you have many options when it comes to requesting data from the agent, the most basic being sample and current data. Sample data (located at `<host's computer ip address>:5000/sample or localhost:5000/sample`) reports all the data points from your connected machines. Current data (located at `<host's computer ip address>:5000/current` or `localhost:5000/current`) reports the most current data points from your connected machines. Additionally, you should be able to further parse the data you want by using HTTP parameters. You may use this link to learn more about how the XML is structured and how to use HTTP parameters in your data request from the agent.

Adapter

Creating and installing the adapter is arguably going to be the most difficult process in setting up MTConnect for your specific use case. Most adapters will likely have to be coded specifically for the machines that you are attempting to establish communication with unless you can find pre-coded adapters for the machine(s) you are using in your project. Google and GitHub are very good sources to see if someone has already developed an adapter for your machine. Finally, finding a pre-coded adapter script for your machine does not guarantee that it will work in your setup, as there are many factors at play when developing an adapter.

Creating Your Own Adapter

If you would like to create your own adapter script from scratch, we suggest you first consult the "Day 1: Understanding and Building Agents and Adapters" video, which details the basics you need to know when you are starting to create your adapter. If you have further questions, you should consult the MTConnect Institute's Youtube page, where they have created numerous videos detailing the inner workings of MTConnect.

Using Existing Adapters

There are numerous variations of adapters written in different languages that have been developed by independent organizations and published online. This is likely the easiest option for those without proficient coding abilities or machine-specific knowledge, as most of the necessary functionalities will have already been developed. However, the challenge with this approach is that understanding what each component does and how everything integrates may be rather difficult. Even more so, using an existing adapter script will most certainly require you to modify the code so that it can interact and pull data from a specific machine. Consulting the user manual of the machine you are working with is important in determining how the parameters should be mapped to their corresponding MTConnect variables. GitHub is a great resource to search for existing adapters from which you can replicate any publicly available codes.

Connecting Adapter and Agent

If you coded the adapter correctly (the IP address and port on both the adapter and agent must match), you should see your live data in the agent, viewable by accessing "`localhost:5000/current`" in a web browser.

Ensuring Operation

In order to ensure operation, you could include print statements in your adapter code to check that you are pulling data from the machine correctly. Different machines stream varying amounts of data, and you most likely will not need to utilize all of them for your needs. Also, to ensure data flow from the adapter to the agent, verify that values are showing up in their respective locations in the XML tree when you navigate to the "`localhost:5000/current`".

Application

Assessing Application needs

An application in MTConnect is any component that utilizes the information coming from the agent so that it could be used by you or your organization for a particular need. These can be anything from simple graphs over time to entire suites of software using the agent data. Your application needs are 100% custom to what you are trying to accomplish with MTConnect. It is important to carefully consider the context and create a list of requirements for your application before starting to research potential applications that would work for you. There are some applications that will natively support an MTConnect agent, but there are also many that will not work right out of the box. Do not let a lack of native support deter you from using a certain application. Just as you can make a custom script for the adapter, it is entirely possible to create a custom script that allows your agent to work with your applications. Our team implemented a scraper script for the agent to translate machine data into our application. It can be found in our GitHub repository.

Options

Some examples of applications that you may want to utilize for yourself include:

- SQC (Statistical Quality Control) software
- Databases for time series storage
- Dynamic visual dashboard
- Completely custom applications for specific business needs

Database

Creating Your Own Database

In many cases, you will want to implement a database into your MTConnect application solution in order to be able to store data from your machines. There are countless amounts of databases and methods to set them up, so your solution will probably be very different from what we implemented in our case. Your database selections and configuration completely depend on many variables not limited to the operating system, data types, and features you would like to develop. You should consult documentation on your chosen database for installation and configuration.

Connecting Database to Agent

For almost all databases, you will have to create a piece of code that extracts current data in the agent and injects them into the database. In our case, we have a "scraper" script that pulls our desired values from our agent's current XML tree and transfers them to a MySQL server running on the same computer. You can find this file on our GitHub page. You most likely will have to consult your database's documentation to learn how to specifically code a script that will allow you to send information to that particular database.

Ensure Operation

You can ensure operation by logging into your database and observing if any new table rows are being added with data that matches your machine's current parameters. It is essential to verify that the data being extracted and stored are the correct information that you would like to utilize. One of the challenges that our team faced was dealing with "junk" data being extracted from the Haas CNC VF-3. From this, we learned that each machine will stream live information differently. A helpful step to take would have been to consult the equipment's manufacturer's guide and familiarize ourselves with the intricacies of the machine's software.

Dashboard

A dashboard is a type of graphical user interface (GUI) that displays information coming from a specific software or hardware. In the case of our implementation, the dashboard takes values coming from the database (or directly from the machine) and consolidates them into a more visually appealing display, offering graphs of the data over time and even visual cues or warnings when certain factors are out of specification. A dashboard may or may not be useful in your particular case, depending on your need for the data collected from your machine(s). For instance, a dashboard is a useful tool that can be utilized to monitor machine performance at a glance, and this is widely applied throughout the industry.

There are many options when it comes to choosing a dashboard, ranging from completely free and open source to very expensive and proprietary software. The most important thing to consider in a dashboard is what functionality you want to get out of it and whether it will integrate successfully with the rest of your MTConnect system. The compatibility aspect mostly has to do with your database choice, as most dashboard providers only natively support certain databases. You may also want to consider the different features you would like to include in your dashboard.

Creating Your Own Dashboard

In our particular case, we chose to utilize Grafana as the dashboard for our MTConnect system due to its free-tier options and full support for MySQL databases. It also provides basic functionality, like the graphing of variables over time, a good variety of live variable visualizations, and the ability to set certain limits for variables that will trigger events if exceeded. You also have numerous other options depending on your decision to use open-source or proprietary software as well as the context from which you are developing an implementation. Researching on Google should yield several alternatives that you may take, with potentially easier applications.

Connecting Dashboard to Database

Most off-the-shelf dashboards available on the market will usually have some level of built-in connectivity with pre-existing and popular data sources. In the case of Grafana (and almost all pre-built dashboards), a page displaying their integrations are provided. If a dashboard is necessary that does not have a particular integration with your database, a customizable dashboard is recommended. Along those same lines, in terms of a process flow, finding a dashboard that suits the needs of the project at hand would be most useful; choose a dashboard based on the necessary integrations and ensure that the dashboard officially supports all of it.

You may choose to write your own dashboard application from scratch. While this is entirely possible to do, and would allow for further customization, we found that utilizing pre-existing dashboards is typically much easier and more efficient. During the course of our implementation, we explored the idea of creating our own dashboard from powerful frameworks and languages such as Angular and TypeScript. Be sure that your needs are specific and customized enough to justify building a dashboard from the ground up.

Ensure Operation

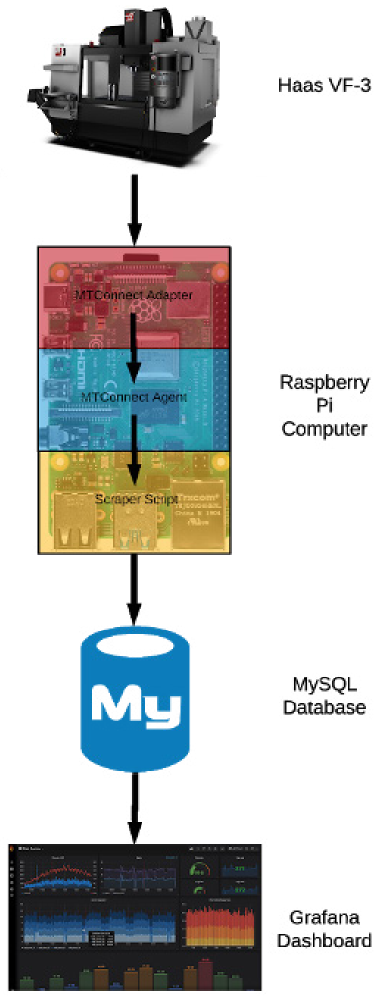
Ensuring operation of your dashboard is a highly customized process. Some dashboards are hosted on the network and available anywhere, while others only run on a single computer. For this reason, you should consult the software company's documentation for your specific dashboard when you are ensuring that your dashboard is working as intended. An operational dashboard should successfully read data from a connected database and be able to visually display them in any form supported by the dashboard software.

Learning Factory Setup

In order to extract the data obtained from the Haas CNC VF-3 mill, we utilized an MTConnect agent and adapter running on a Raspberry Pi computer. The adapter is a Haas-specific piece of Python code that interprets the information transmitted by the Haas CNC machine over its USB (Universal Serial Bus) output port. The adapter code then feeds the live machine information into the C++ agent running on the Raspberry Pi computer. The agent keeps track of all the specified parameters of the machine, such as spatial coordinates, and organizes them into an understandable XML format. The agent also hosts all of the live information on the local network; any computers or machines connected to the local network are able to connect to it and receive information.

Next, a MySQL (Structured Query Language) database is built to store the live data streaming from the agent. This is necessary because the MTConnect agent has no storage or database capability built in, so its sole task is to provide up-to-date, live data from the machine. The goal is to store past data in order to collect and form metrics such as machine uptime and part count. These metrics will help users make informed decisions to improve workflows.

Finally, a dashboard is created using an open-source software called Grafana, which allows us to display live data from the machine as well as historical data in a visually appealing and easy-to-understand format. Grafana is designed to connect seamlessly with a MySQL database, which is why we chose this specific software, although there are numerous other alternatives on the internet.



Future Implications

Amazon Web Services' (AWS) Cloud-Based Database

One of the main considerations for this standards-based approach in the integration of data collection and manufacturing technology is making MTConnect available in the cloud. The team at Virginia Tech had the opportunity to explore this possibility and collected the following information.

Cloud computing is fast becoming the technology of the future, and it has grown tremendously over the last decade. Given the nature of MTConnect as a data protocol, one huge benefit would be to store data from interconnected manufacturing technology in the cloud so that it is accessible from anywhere in the world. Some possible avenues to explore include services such as Amazon EC2 (Elastic Cloud Computing), Amazon Redshift, and Amazon S3 (Simple Storage Service).

Reducing Dependence on Highly Specialized Knowledge or Software

With the implementation of the standards-based approach, people who do not have specialized knowledge, such as students and small to medium manufacturing companies, will be able to use this approach to establish connectivity within the manufacturing facility, allowing for more efficient communication.

Facilitating the Adoption of MTConnect in Academia and Industry

Working in an academic environment allows for the experimentation and documentation of the MTConnect standard. This allows students and future manufacturers to explore the limits and strengths of the standard and helps its expansion in manufacturing industries and areas of manufacturing academia. For example, this document was created by a group of Industrial and Systems Engineering students from the Virginia Polytechnic Institute and State University's Grado Department of Industrial and Systems Engineering through their Industry 4.0 research hub provided on campus, called the "Learning Factory."

FAQ

Why is my MTConnect sample request reporting an exorbitant amount of sequence numbers even if my machine parameters are not changing?

If you experience this problem, you have likely coded your adapter in a way where it sends every new set of data points collected as "new" data points, even if they are not "new." This means that every time your adapter loop runs, it collects your specific data points and then sends all of them to the agent without checking if they have changed. The solution to this is to write a logic check to ensure that a parameter is only sent to the agent when it is determined that it has changed since the last time it was collected. You can refer to our agent script to see how we implemented this feature.

Why do my agent, adapter, and/or my other application(s) not start when my computer restarts/boots?

Most programs you install will not run automatically when you start your computer. If you are running on a Raspberry Pi, you can look into both system service commands (`systemclt`) and bash scripts. System service commands make your script or application run when the Raspberry Pi boots by adding it to the system service list. Information about system service commands can be found here. Alternatively, you could choose to make a bash script, which does not start on boot but allows you to start multiple applications and scripts at the same time with one execution. You can find more information about bash scripts here.

How can I control and send commands to my Raspberry Pi (or other computer) if I cannot directly connect to it with a display, keyboard, and mouse?

You can control and send commands to your computers by using ssh. One widely used ssh client is PuTTY1.

My machine does not have any computer or sensors on it. Can I still use MTConnect?

You cannot use MTConnect. If you have no sensors or any other elements that can interface with a computer, you will be unable to send data from your machine, which in turn prevents the connection to your MTConnect agent. However, you may be able to add useful sensors to legacy equipment.

How often can I collect data points and send them to my agent?

There is no hard limit on how often you can collect data points, but your hardware and code will definitely affect the frequency of data collection. The two main elements that would affect how often data could be collected would be the machine and adapter script. Your machine can limit your data collection based on how many requests you can send to it per second. Your adapter script could limit your data collection based on how often your adapter script loop runs.

If I already have an operational agent and adapter, do I need another agent in order to add another adapter?

No, you do not need another agent to add another adapter. If you already have an agent setup, you can add as many adapters to it as you would like to achieve your goal. For every adapter you add, you will just have to add the adapter to your `agent.cfg` file in order to make it operational. Having all the adapters feeding one agent is very advantageous as it allows you to see all the machine's data points on one XML file/screen.

Do the adapter and the agent have to run on different computers?

No, the adapter and the agent do not have to run on different computers. In the experimentation that led to the adaptation of this handbook, the agent, adapter, database, and dashboard were all run on the same Raspberry Pi. However, keep in mind that the more applications you run on your computer, the more processing power it takes on the computer, which will likely reduce performance and speed. This may affect how often you can collect data points. This is really only an issue on low-power computer hardware, like the Raspberry Pi.

How do I decide between “sample” and “current” data selection when I want to pull my machine data?

You should use current data selection when you want to know about the most recent machine data. If you would like to see the progression of the machine data point values over time, you should use a sample.

Why is my adapter working, but the data points are not showing up in the agent?

If you can verify that the adapter is working, but you cannot see the data points updating in the agent, you are probably experiencing one of three problems. The first is that the address your adapter is sending data to does not match the address set in the `agent.cfg` file for that adapter. Second, you could be exporting data in an incorrect format that does not match what the agent requires. You can refer to the adapter code to see our example of a correct way of exporting data, or you could refer to here for more information. Finally, the last problem you could have is a mismatch between your adapter code and XML schema regarding DataItem names. For example, if you wanted to reference a DataItem with `name=“spindlespeed”` in order to populate the MTConnect agent XML field, you would have to reference it exactly as “spindlespeed” in the adapter script (an example output would be `2013-05-13T16:00:05.0000Z|spindlespeed|2500`). Refer to the MTConnect User Portal to understand more about the XML structure of the agent.

Useful Links

Documentation Sources

GitHub Repository of Our Adapter & Database/Grafana Code

Grafana: The open observability platform | Grafana Labs

MTConnect C++ Agent Github

MTConnect Standard Part 1 PDF

Software Download Sources

https://grafana.com/grafana/download

https://www.mysql.com/downloads/

Provided Resources

https://github.com/nombreinvicto/HaasMTConnect

https://www.mtconnect.org/resources

Authors

Jannat Asim

Paula Clares

John Luksas

Dan Nguyen

Technical Advisor

Shaurabh Singh