# Critical manufacturing
an ASM PT company

# Introduction

## 10.2

March 2024

# Introduction

## What is Connect IoT

Connect IoT is the Critical Manufacturing MES approach to Equipment Integration. It provides a simplified process of abstraction of different shop floor equipments and enabling a low-code solution to connect to them to the MES.

## Connect IoT Main drivers

This solution was built to:

- Reduce the time and effort of Equipment Integration implementation and maintenance.
- Allow versionable integrations, ensuring that both system tests and validated developments are not changed.
- Have a lighter solution in regards to computational resources and allow compatibility with different operational systems and processors.
- Allow IoT devices to be connected to a secure platform.
- A centralized definition of integration in the MES platform.

## Connect IoT Simplified Architecture
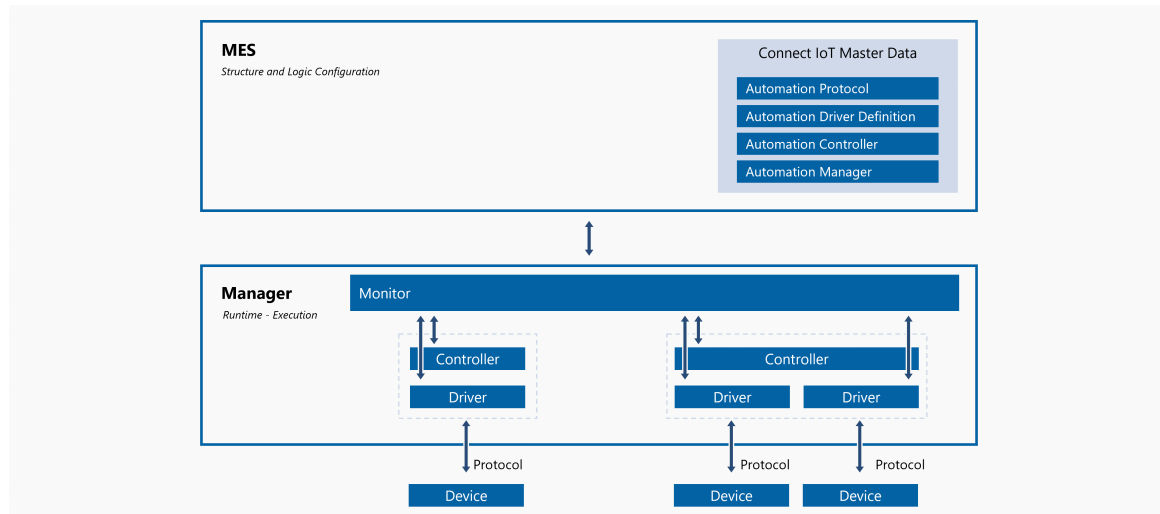


### Automation Protocol

To perform an equipment integration, you need to know which protocol the equipment uses to communicate. A protocol represents a communication convention that controls and allows communication and data transfer between two connected systems.

The Connect IoT solution supports a set of popular communication protocols used in multiple areas in the manufacturing industry, as well as being extensible to provide support to new existing or custom protocols.

The entity **Automation Protocol** in the CMF MES represents the protocol concept.

## Automation Driver Definition

A protocol has a set of defining principles that are intrinsic to the protocol itself. A driver definition is the mapping of that protocol to the specific use case of a machine. A machine may use TCP-IP as its transport layer protocol, but that is insufficient to know that a machine sends a particular stream when it starts, for example.

The **Automation Driver Definition** exists to provide that mapping, splitting machine communication into events and commands. The Automation Driver Definition inherits the protocol definitions specified in the Automation Protocol, and adds on it the mapping to a particular instance of that protocol implementation.

This entity will inform the driver running process on how to connect to the machine, what to expect and what to forward to the controller process as relevant occurrences.

## Automation Controller

The controller is where all business logic is located - event notification, data transformation, commands, etc. - based on the information sent from or to one or more drivers. The actions that result from the business logic may involve interactions with other systems, controllers and/or drivers. This concept is mapped from the MES to the Connect IoT through the **Automation Controller** entity. Along with the entity, a low code tool is provided to enable workflow support for the the development of the required business logic.

When it is associated with an MES entity, it is responsible to communicate with the MES and to establish the workflow logic, managing the interactions with the equipment, data transformation (which may include communicating between drivers, other controllers or systems) and data retrieval and storage. These workflows are designed through the use of tasks, elementary operational logic gates that abstract business logic with a plethora of different options and procedures available that provide vast extensibility to the solution.

As it can be seen, there is a clear separation of concerns. The driver definition is concerned with interfacing and the controller is concerned with what to do with the machine information.

## Automation Manager

The **Automation Manager** is the process orchestrator of your implementation. It is a structure that contains the drivers and controllers. As the parent process it is responsible for starting all other processes and ensuring that they are running or stopped when needed. The Manager is responsible for the process life cycle and it is represented by the **Automation Manager** in the CMF MES.

To be able to start and stop the processes, the manager needs to know the location of where it can download its dependencies. In the MES, this is known as the package repository location, this location is where all the possible dependencies of the manager process will reside. A dependency will be for example, the required source code for a particular driver. This prevents having to ship all possible dependencies with the manager. The manager when starting to run will download what is relevant for a particular implementation and will only use that.

## Automation Monitor

There is an additional technical component that functions as an interface between the Automation Manager and the MES system. This component is responsible for accessing the MES and retrieving all the configurations for that particular Manager. This component does not require a matching entity in the MES.

The Automation Monitor is the interface between a system (MES) and the manager to which the monitor is tightly coupled. It knows the Automation Controllers that the manager contains, the Automation Driver Definitions and Automation Protocols for each one, as well as the workflows that are set to be executed.

It informs the manager on which driver version is used by the controller workflows and therefore needs to be started, as well as the location of the files that support the driver. It is also responsible to verify the current state of the processes (if they are up or down or in some other state).

It needs to know:

- The repository type and location
- Which driver and controller needs to be started
- The address of the MES for interaction between the Manager and the system

# Legal Information

**Disclaimer**

The information contained in this document represents the current view of Critical Manufacturing on the issues discussed as of the date of publication. Because Critical Manufacturing must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Critical Manufacturing, and Critical Manufacturing cannot guarantee the accuracy of any information presented after the date of publication. This document is for informational purposes only.

Critical Manufacturing makes no warranties, express, implied or statutory, as to the information herein contained.

**Confidentiality Notice**

All materials and information included herein are being provided by Critical Manufacturing to its Customer solely for Customer internal use for its business purposes. Critical Manufacturing retains all rights, titles, interests in and copyrights to the materials and information herein. The materials and information contained herein constitute confidential information of Critical Manufacturing and the Customer must not disclose or transfer by any means any of these materials or information, whether total or partial, to any third party without the prior explicit consent by Critical Manufacturing.

**Copyright Information**

All title and copyrights in and to the Software (including but not limited to any source code, binaries, designs, specifications, models, documents, layouts, images, photographs, animations, video, audio, music, text incorporated into the Software), the accompanying printed materials, and any copies of the Software, and any trademarks or service marks of Critical Manufacturing are owned by Critical Manufacturing unless explicitly stated otherwise. All title and intellectual property rights in and to the content that may be accessed through use of the Software is the property of the respective content owner and is protected by applicable copyright or other intellectual property laws and treaties.

**Trademark Information**

Critical Manufacturing is a registered trademark of Critical Manufacturing.

All other trademarks are property of their respective owners.