Critical manufacturing
an ASM PT company

# MES Interoperability Scenarios

## 12.0.1

November 2026

DOCUMENT ACCESS

Public

DISCLAIMER

The contents of this document are under copyright of Critical Manufacturing S.A. it is released on condition that it shall not be copied in whole, in part or otherwise reproduced (whether by photographic, or any other method) and the contents therefore shall not be divulged to any person other than that of the addressee (save to other authorized offices of his organization having need to know such contents, for the purpose for which disclosure is made) without prior written consent of submitting company.

# Connect IoT - MES Interoperability Scenarios Tutorial

*Estimated time to read: 13 minutes*

Critical Manufacturing MES provides a set of out-of-the-box interoperability scenarios between the MES and Connect IoT. When a scenario is enabled on a **Resource**, the corresponding MES transaction sends a request to the **Automation Controller** instance associated with the **Resource** and waits for the equipment confirmation before completing. This request and reply exchange is called interlocking.

The available scenarios are:

- Resource setup - the begin setup transaction waits for the equipment confirmation.
- Material dispatch, track-in, and track-out - the material transactions wait for the equipment confirmation.
- Container dock and undock - the container transactions wait for the equipment confirmation.
- Recipe list, download, and upload - the MES lists the recipes at the equipment, downloads recipes from the equipment, and uploads recipes to the equipment.

This tutorial shows how to enable the scenarios on a **Resource** and how to build the equipment-side workflows that answer the MES requests. The material dispatch scenario is used as the working example; all other scenarios follow the same pattern.

## Prerequisites

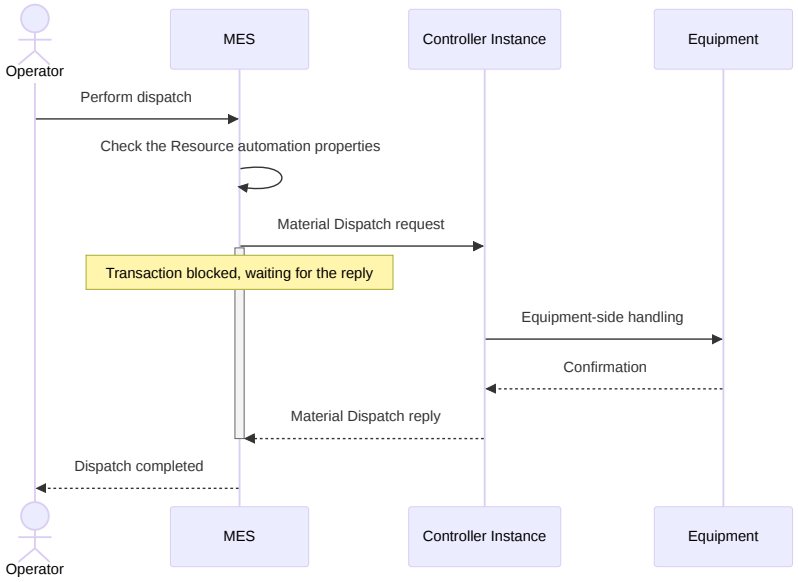
This tutorial assumes a working Connect IoT setup:

- An **Automation Controller** with a controller instance associated with the **Resource** used in the scenarios. If you are not familiar with these entities, complete the Basic Configuration Tutorial first and see How to Create a Controller Instance.

- Familiarity with Data Flow or Control Flow workflows. See the Control Flow Configuration Tutorial for the Control Flow approach.

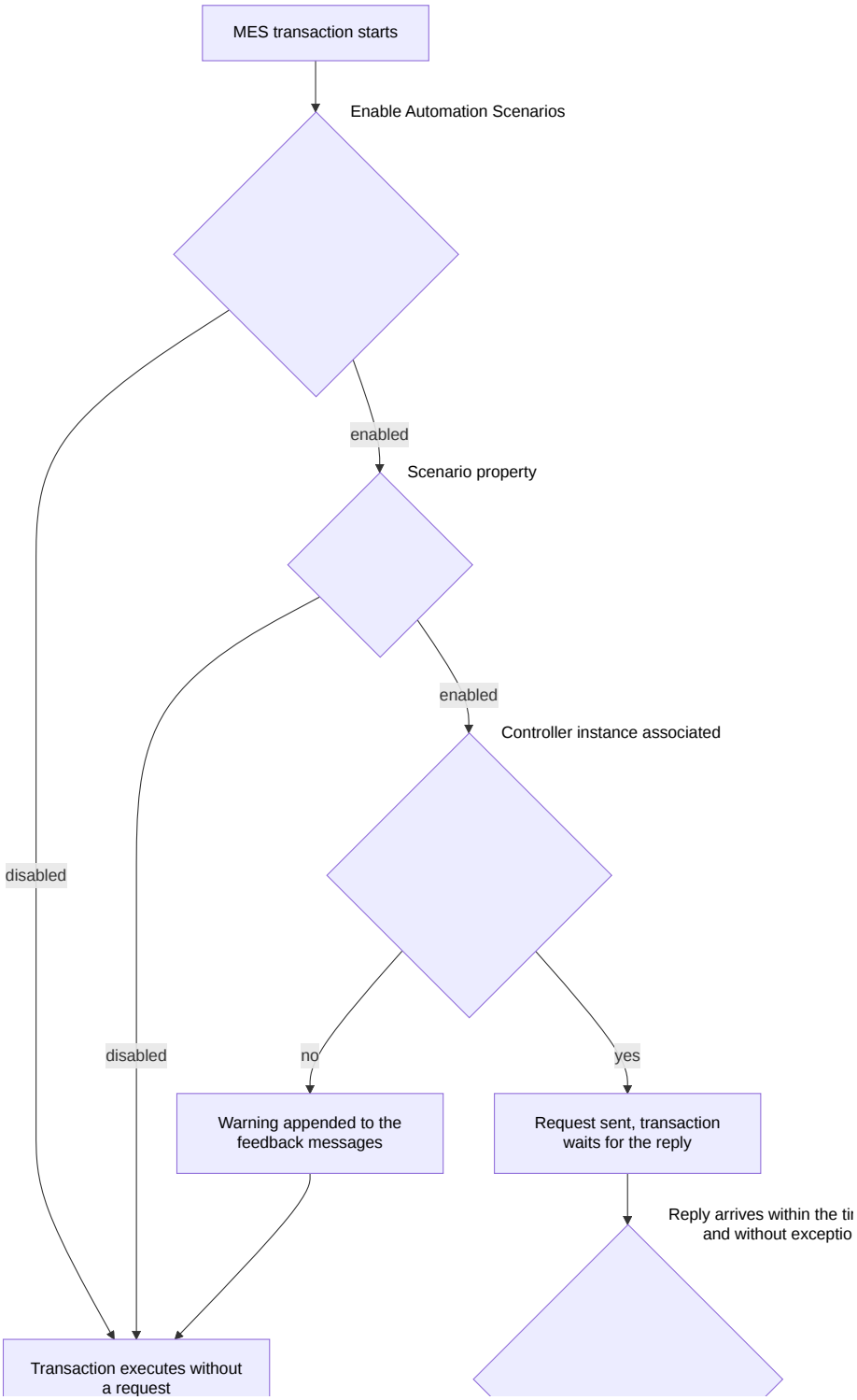
## How the Scenarios Work

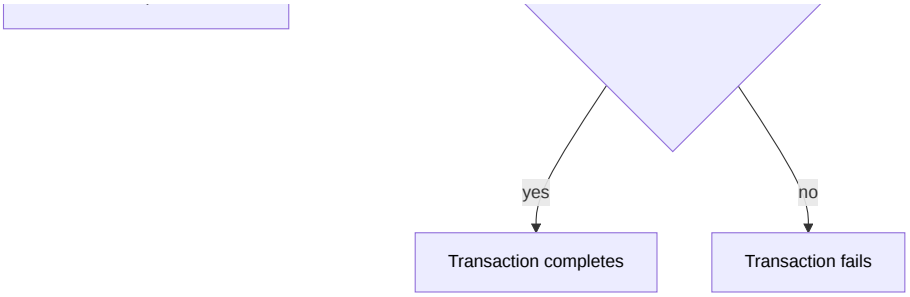
The diagram below shows the interlocking exchange for the material dispatch scenario. The same sequence applies to all scenarios, only the transaction and the tasks change.




Each MES transaction covered by a scenario checks the automation properties of the **Resource** involved in the operation:

1. If Enable Automation Scenarios is `false`, or the property of the specific scenario is not enabled, the transaction executes as usual and no request is sent.
2. If the scenario is enabled, the MES retrieves the **Automation Controller** instance of the **Resource** or, when defined, of its Automation Scenarios Owner Entity.
3. The MES sends the request to the controller instance and stays blocked waiting for the reply.
4. The workflow handles the request on the equipment side and sends the reply back.
5. If the reply carries an exception, or no reply arrives within the reply timeout, the MES transaction fails. See Connect IoT Administration for the interlocking timeout configuration.


MES transaction starts
Enable Automation Scenarios
enabled
Scenario property
enabled
Controller instance associated
disabled
disabled
no
yes
Warning appended to the feedback messages
Request sent, transaction waits for the reply
Reply arrives within the ti
and without exceptio
Transaction executes without a request




**Note**

If a scenario is enabled but the Resource has no associated controller instance, the transaction does not fail. The MES appends a warning to the transaction feedback messages instead.

Each scenario pairs an MES operation with a trigger task and, in Data Flow workflows, a reply task:

| Scenario | MES operation | Resource property | Trigger task | Reply task |
|---|---|---|---|---|
| Resource setup | Begin Setup | Resource Setup Interlocking | On Resource Begin Setup Request | Send Resource Begin Setup Reply |
| Material dispatch | Dispatch Material | Material Dispatch Interlocking | On Material Dispatch Request | Send Material Dispatch Reply |
| Material track-in | Track-In Material | Material Track-In Interlocking | On Material Track In Request | Send Material Track In Reply |
| Material track-out | Track-Out Material | Material Track-Out Interlocking | On Material Track Out Request | Send Material Track Out Reply |
| Container dock | Dock Container | Container Dock Interlocking | On Container Dock Request | Send Container Dock Reply |

| Scenario | MES operation | Resource property | Trigger task | Reply task |
|---|---|---|---|---|
| Container undock | Undock Container | Container Undock Interlocking | On Container Undock Request | Send Container Undock Reply |
| Recipe list | Get recipes at the equipment | Recipe List Support | Recipe List Request | Send Recipe List Reply |
| Recipe download | Download recipe from the equipment | Recipe Download Support | Recipe Body Request | Send Recipe Body Reply |
| Recipe upload | Upload recipes to the equipment | Automation Upload Support | Recipe Upload Request | Send Recipe Upload Reply |

Table: Out-of-the-box MES interoperability scenarios

The setup, material, and container scenarios interlock an MES transaction: the transaction waits for the equipment confirmation. The recipe scenarios exchange recipe data instead: the MES requests the recipe list, a recipe body, or delivers an uploaded recipe, and the workflow replies with the data or the confirmation.

## What MES Sends and Expects

Each setup, material, and container scenario is a system DEE action that runs in the `Post` step of its MES service, for example `MaterialManagement.MaterialManagementOrchestration.TrackInMaterials.Post` for track-in. MES sends the request after the service has done its work, and the service call ends only when the reply arrives or the timeout passes.

| Scenario | Requests MES sends | Outputs of the trigger task | What the reply must carry |
|---|---|---|---|
| Resource setup | One per **Resource** of the operation | `resource` | Anything: any reply confirms the |

| Scenario | Requests MES sends | Outputs of the trigger | What the reply must carry |
|---|---|---|---|
| Material dispatch, track-in, and track-out | One per **Material** of the operation, one after the other. For track-out, the **Resource** is the last processed **Resource** of the **Material** | `resource`, `material` | Anything: any reply confirms the operation |
| Container dock and undock | One per operation | `resource`, `container` | Anything: any reply confirms the operation |
| Recipe list | One each time the recipes at the equipment are requested | None besides `Success` | A list of recipe names, or of objects with a non-empty `Name` |
| Recipe download | One per recipe, with the recipe name at the equipment | `recipeName` | The recipe content in `body`, base64-encoded, and its file name in `filename` |
| Recipe upload | One per recipe and **Resource** | `recipeName`, `recipeIsBinary`, `recipeBody` | Anything confirms the upload; an `Error` reports it as failed |

Table: What each scenario sends and expects

For the setup, material, and container scenarios:

- **Any reply confirms the operation.** MES does not read the content, so `{}`, `false`, and `{"accepted": false}` all confirm it.
- **An error refuses the operation with a reason.** In Data Flow, link an `Error` into the `reply` input of the reply task, for example the `Error` output of the task that checks the equipment. In Control Flow, an error that nothing catches in the handler branch, for example one thrown by a **Code** task, has the same effect. The workflow sends `{"__cmf_type__": "Error", "message": "<message>"}`, and MES fails the operation with that message, which the operator sees. An error with an empty message confirms the operation.

- **A missing reply fails the operation.** When the Reply Timeout of the trigger task passes first, the workflow answers with the error `Request for '<subject>' timed out while waiting for a reply!`, and MES fails the operation with that message. When the MES timeout passes first, or the instance is not running and nothing answers, MES fails the operation with its own timeout message. See Connect IoT Administration.
- **Each request has its own timeout.** MES sends the requests of one operation one after the other, so a track-in of 25 **Materials** can wait up to 25 times the timeout.
- **One refused request undoes the whole operation.** MES runs each service call in one transaction, so when one request fails, the operation is rolled back for every **Material**, including those whose requests the workflow already confirmed. The equipment has already received those requests, so do not let a confirmation start work that a later refusal cannot undo.
- **Other entry points send the same requests.** The services `ComplexTrackInMaterials` and `ComplexDispatchAndTrackInMaterials` run `TrackInMaterials`, and `ComplexDispatchAndTrackInMaterials` also runs `DispatchMaterials`. A **Dispatch and Track-In** therefore sends the requests of both the dispatch and the track-in scenarios.

For the recipe scenarios:

- **Recipe list** - link an array to the reply task. The trigger turns each recipe name into `{"Name": "<name>"}`. A text is sent as one item without a name, and an item without a name fails the request.
- **Recipe download** - MES decodes `body` from base64. A buffer is encoded for you; a text must already be base64-encoded. In Control Flow, return an object with `body` and `filename`.
- **Recipe upload** - for any reply except an error, MES records the upload on the recipe of the **Resource**, with its last upload version and date. An `Error` linked into `Failure Reason` adds its message to the feedback of the upload, and a request without a reply adds a timeout message. The upload operation itself does not fail.
- When no reply arrives for a recipe list or a recipe download, MES fails the request with `Failed to get recipe list. No reply to get recipe list request.` or `Failed to get recipe body. No reply to get recipe body request.`

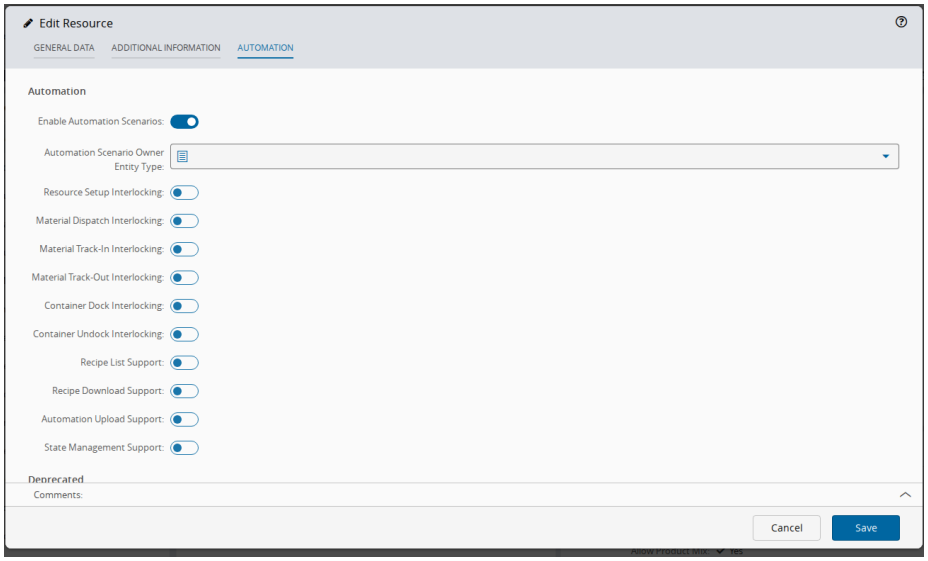
To keep MES running while the equipment or its instance is down, set the property of the scenario to `false` on the **Resource**. The transaction then runs without a request until you enable the property again.

For how each designer turns a value into a reply, see Connect IoT - Talking to MES Tutorial.

## Enable the Scenarios on the Resource

The scenarios are configured on the Automation step of the **Resource** wizard. For the full description of the automation properties, see Create Resource.

1. Go to **Business Data > Resource** and select the **Resource** to integrate.
2. Select **Edit**.
3. Go to the **Automation** step.
4. Set Enable Automation Scenarios to `true`.
5. Enable the properties of the scenarios that the equipment supports. For this tutorial, enable Material Dispatch Interlocking.
6. Select **Save**.




> **Note**
>
> Recipe List Support, Recipe Download Support, and Automation Upload Support all require the Recipe Management Enabled property of the Resource.

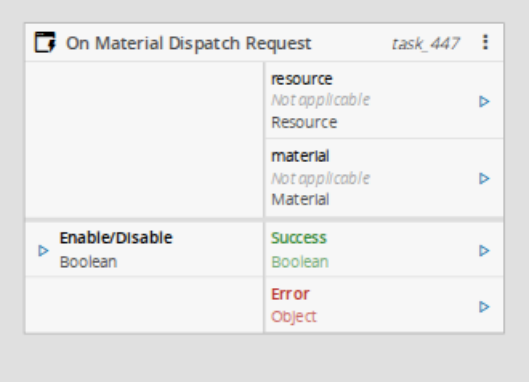
If the **Automation Controller** instance that handles the requests belongs to another entity, for example an **Area** that groups several **Resources**, define the Automation Scenario Owner Entity Type and the Automation Scenarios Owner Entity properties. The requests are then handled by the controller instance of that entity instead of the controller instance of the **Resource**.

> **Info**
>
> The previous automation properties (Mode, Address, and the automation capabilities) are deprecated and grouped in the Deprecated section of the Automation step. Use the new properties for all new integrations.

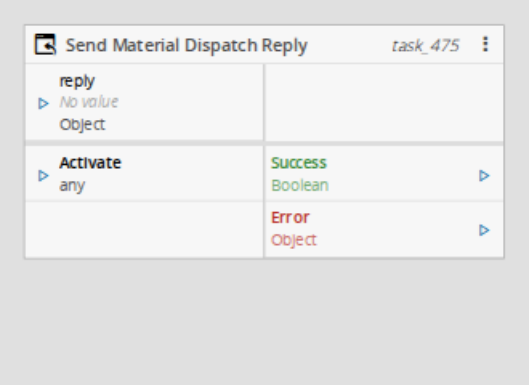
## Build the Equipment-Side Workflow (Data Flow)

With the scenario enabled, the MES blocks the dispatch transaction until the workflow replies. Build a workflow that receives the request, performs the equipment-side handling, and sends the reply.

1. Open the **Automation Controller** associated with the **Resource** and create a new Data Flow workflow.
2. Add the **On Material Dispatch Request** task, available in the `System / MES` category, under `Triggers`. This task emits the `resource` and the `material` involved in the operation when a request arrives.

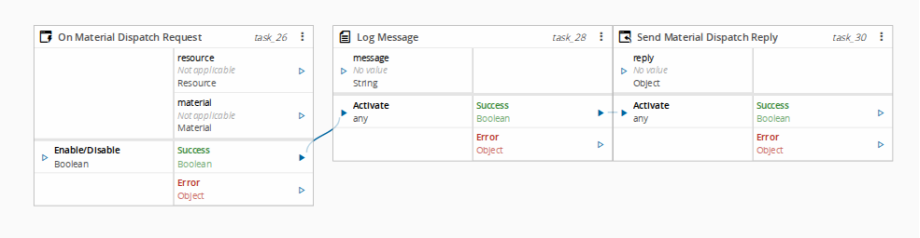



3. Connect the task outputs to the logic that prepares the equipment, for example writing the material identifier to a device variable and waiting for an acknowledgment.
4. Add the **Send Material Dispatch Reply** task, available in the `System / MES` category, under `Actions`, and activate it when the equipment-side handling completes.



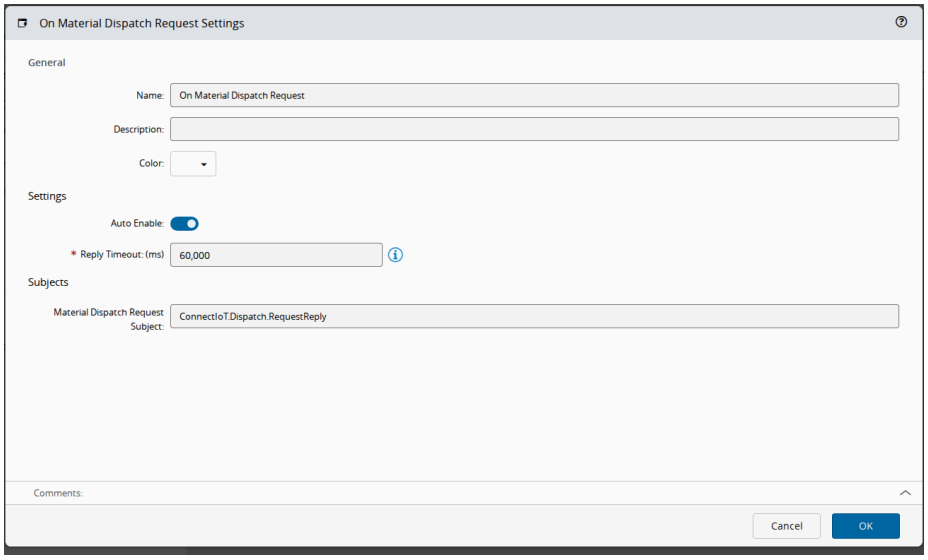

5. Deploy the workflow changes and start the **Automation Manager**.

The resulting workflow follows this shape, with a **Log Message** task standing in for the equipment-side logic:




> **Warning**
>
> The reply task must run in the same workflow branch as the request it answers. The request and reply tasks share context through a structure called `zone`; a reply task activated outside the zone of the request emits an error instead of unblocking the MES. See Zones for more information.

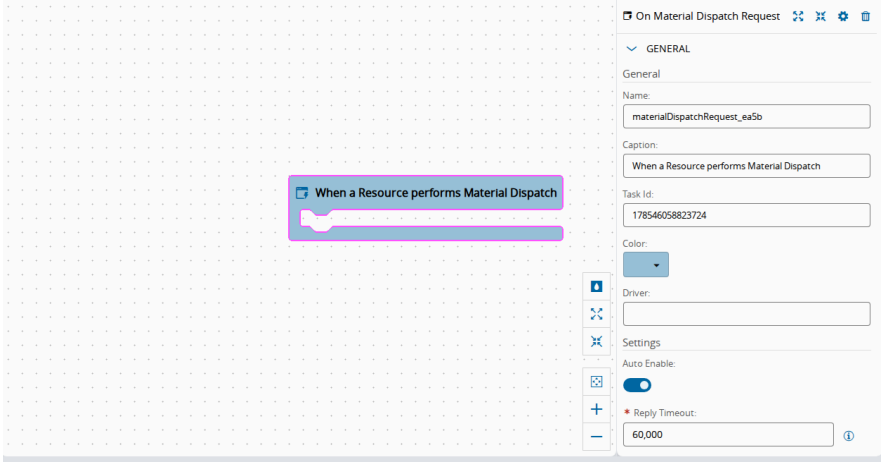
The trigger task has a `Reply Timeout` setting, with a default value of 60000 milliseconds. If the workflow does not reply within this time, the trigger emits `Error` and the MES transaction fails. Adjust the timeout to the time the equipment needs to confirm the operation.




## Build the Equipment-Side Workflow (Control Flow)

In a Control Flow workflow, the request and reply pair is represented by a single `Start` trigger, named `When a Resource performs Material Dispatch` for the dispatch scenario. The result of the handler branch is sent back to the MES as the reply, so no reply task is needed.

1. Open the **Automation Controller** and create a new Control Flow workflow.
2. Add a `Start` trigger and select `When a Resource performs Material Dispatch`.




3. Implement the handler with the equipment-side logic. When the handler finishes, the reply is sent automatically.

## Test the Scenario

1. Go to a **Material** that can be dispatched to the integrated **Resource** and perform Dispatch Material.
2. Observe the running **Automation Manager**. The workflow receives the request, executes the equipment-side logic, and sends the reply.
3. The dispatch transaction completes in the MES after the reply arrives.

To confirm the interlocking behavior, stop the **Automation Manager** and dispatch another **Material**. The transaction stays blocked and fails when the reply timeout expires, showing that the MES only completes the operation with the equipment confirmation.

## Next Steps

- Enable the remaining scenarios that the equipment supports and add the corresponding trigger and reply tasks to the workflow. All scenarios follow the pattern shown in this tutorial.
- See the reference pages listed in the scenarios table for the inputs, outputs, and settings of each task.

# 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.

Copyright © 2023 Critical Manufacturing. All rights reserved.