

Standard Production Cell Interface



Revision history

Version	Date	Author	Description
2022.12.Beta	21-12-2022	PAI AEF PPO	Initial layout, domain model and draft for contents.
2023.1.Beta	24-01-2023	AEF SML PPO	Added communication, telegrams and flows sections. Changes to domain model.
1.1	04-05-2023	PPO AEF	Changes for better support PLCs <ul style="list-style-type: none"> - id as strings - dashes changed to underscore in JSON keys Changes to parameters for orders. Changed from position_id to location_id Updated a few telegram titles
1.2	03-07-2023	AEF	Added "load_carriers" to production manifest. Added examples for Kitting Cell in appendix.
2.0	14-12-2023	SML AEF	Updated to use v2.0 of the communication protocol. Split up the document into two parts: Standard Equipment Interface & Standard Production Interface. Added telegrams for handling access to resources and production scenarios. Updated scenarios & examples. Several corrections throughout the document.
2.1	06-02-2024	AEF	Made adjustments to the following telegrams: <ul style="list-style-type: none"> - Item/Container Transport Request - Load Carrier Ready for Pickup - Production Manifest
2.2	02-04-2024	AEF	Added the option to use sub-locations for "Item/Container at Location". The cell can report an error if insufficient space is available to start a new batch.
2.3	10-10-2024	MIV	Added data types in section Telegram Format and removed them from telegrams. Changed definition of Load carrier object to match differences in telegrams. Added QA-passed property to batch manifests.
2.4	13-03-2025	SML	Updates to types and telegrams. The communication protocol has been made more agnostic.
	26-06-2025	PDA	Release

Review history

Version	Date	Reviewer	Notes / description
2022.12.Beta	21-12-2022	PPO	Review
2023.1.Beta	24-01-2023	PPO	Review
1.2	03-07-2023	PKI KVN	Review
2.0	14-12-2023	AEF SML	Review
2.1	13-02-2023	PPO	Review
2.3	30-10-2024	AEF	Review
2.4	13-12-2025	AEF	Review
	24-06-2025	PDA	Release

Table of Contents

1 Introduction	5
1.1 References	5
1.2 Glossary	5
1.3 Copyright and right to use under Apache License 2.0	5
1.4 Background / 5G-Robot	6
2 Terms and Concepts	7
2.1 Terminology	7
2.2 Domain Model	7
2.3 Production Cells	7
2.3.1 Input Item Management	8
2.3.2 Output Item Management	8
2.3.3 Location Management in Container	8
3 Communication	10
3.1 Using the MQTT Protocol	10
4 Common Data Types	11
4.1 Segment Parameters	11
4.2 Segment	11
5 Error Types	13
6 Production Cell Telegrams	14
6.1 Batch Instructions (HLC -> Cell)	14
6.2 Batch Instructions Reply (Cell -> HLC)	15
6.3 Batch Instructions Complete (HLC <-> Cell)	16
6.4 Production Manifest (Cell -> HLC)	17
7 Scenarios and Examples	19
7.1 Production	19
7.1.1 Start Production Batch	19
7.1.2 Replenish with Materials	19
7.1.3 Item(s) Produced and Container Ready for Pickup	20
7.1.4 Production Completed	21
7.2 Production Error Scenarios	22
7.2.1 Batch instructions are sent when a cell is in invalid state	22

1 Introduction

This document is a specialization of the standard equipment interface [1] that describes the standard production cell interface. This interface is used for communication between a high-level control (HLC) system and a production cell.

Detailed descriptions of the communication protocol, telegram format, and common telegrams used by this interface can be read in the standard equipment interface document [1]. Readers are advised to become familiar with that document before implementing the standard production cell interface.

Note that, in case the production cell has capabilities for transportation of containers, e.g., moving a container to/from the production cell itself and other equipment, the production cell may also implement the standard transport interface [2].

1.1 References

ID	Document	Description
[1]	Title: Standard Equipment Interface v2.4	The base interface description used for all equipment / robot cell open interfaces.
[2]	Title: Standard Transport Interface v0.1	Specification for moving items or containers between a robot cell and external equipment/locations.
[3]	Title: Domain Model v1.3	Defines various terms and concepts used for the domain.
[4]	Title: MQTT Protocol for Standard Interfaces v1.0	Protocol specification for using MQTT with the standard open interfaces.

1.2 Glossary

Abbreviation	Description
Cell	Production Cell - robot controller or PLC controlling the cell.
HLC	High-Level Control system - e.g. WCS/WMS, MES or MFS.

1.3 Copyright and right to use under Apache License 2.0

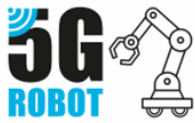
The copyright on this document and any contained specifications, designs, etc. belongs to the company: Intelligent Systems A/S, Havnevej 11, DK-9560 Hadsund, DENMARK.

Any referenced 3rd party designs, technologies and or other IP shall remain with the original owners.

The reference architecture, designs and the included open standard integration interfaces are open and free to use under Apache License 2.0. For details please see

<https://www.apache.org/licenses/LICENSE-2.0>.

1.4 Background / 5G-Robot



Parts of this document / release was made in the **5G-Robot** project also known under the long name **5G-ENABLED AUTONOMOUS MOBILE ROBOTIC SYSTEMS** - the largest innovation project that has been launched under the Innovation Fund Denmark's (IFD) Grand Solutions program.

The groundbreaking project united Denmark's leading robot, automation and factory digitalization companies as technology vendors, research partners and industry-leading end-user companies.



Illustration: Project partner logos.

The aim of the project was to revolutionize manufacturing - paving the way to smart production and smart factories and the application of a number of new technologies in production and manufacturing including 5G wireless communication, cloud and edge computing and digital twin.

Intelligent Systems played a leading role in the project, providing the glue that ties the robotic solutions of the partners together making the work as one - i.e. one connected integrated intelligent manufacturing system.

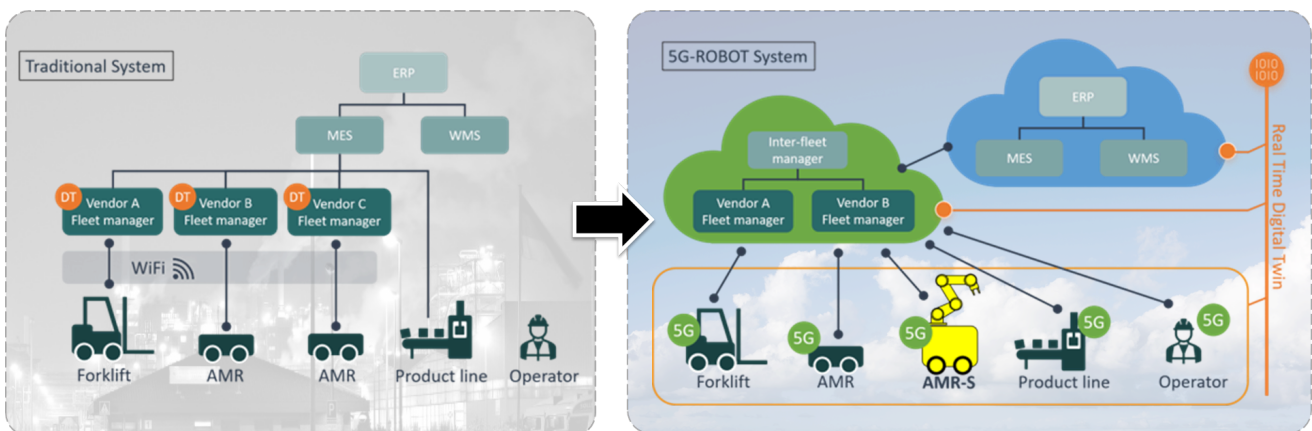


Illustration: The aim is to revolutionize manufacturing paving the way to smart production and smart factories.

Read more about the 5G-Robot project here: <https://www.5gsmartproduction.aau.dk/5g-robot>

2 Terms and Concepts

This section describes the terms that are used throughout the document, their respective explanations and how they relate to one-another.

2.1 Terminology

Term	Description
Equipment Concept	
Equipment	A piece of or a group of physical hardware, such as robot arms, actuators, and barcode scanners.
Location	A place where an object exists. This can either correlate with a physical location or can be used for logical grouping. Typically used for specifying where a container or items are located.
Production Cell	A piece of equipment used for manufacturing items.
Manufacturing Concept	
Product Recipe	Provides the operations/product definition for a given product, i.e. the recipe for how to produce a given product. There can be multiple recipes for the same product (e.g. by using different pieces of equipment). Can also be referred to as the Operations- or Production Definition.

2.2 Domain Model

This section contains the domain- and object models used for the interface.

Refer to the following concepts in the domain model [3]:

- Equipment Concept
- Items Concept
- Production Order Concept
- Manufacturing Concept

2.3 Production Cells

A production cell is a specialized piece of equipment that can perform a process on one or more input items (materials) resulting in one or more output items (materials or finished items). Part of the process may result in scrap that must be discarded.

The process performed by a production cell is described in a *Product Recipe* and a list of properties. These settings may be managed on and by the HLC or supplied by an external system, but it should be sufficient in most cases so that the equipment can perform an action.

This definition of a production cell is very broad and will as such encompass many different types of equipment. The equipment may be an autonomous mechanical device and in other cases simply be an operator, but in all cases the process to perform is sent by the HLC and the equipment or the operator is able to understand the *product recipe*. In this document, equipment can implicitly also be an operator.

This interface focuses on equipment that either creates new items from the input materials (manufacturing or assembly cells), performs a specific action on a single item to modify (manufacturing cells) or verifies an item (quality assurance cells).

For more specialized setups where the properties are more specific to the purpose or the flow of telegrams is more complex, the expectation is that specialized standard interfaces will be created for such setups. One such example is *palletizing equipment*, which in essence uses a *product recipe* to process items, but here the flow of items to the equipment is more complex and therefore has its own standard interface.

2.3.1 Input Item Management

Input items come in many shapes and sizes, making it impossible to provide one general description that suits all input items. For instance, injection molding equipment uses plastic pellets directly, and therefore, their input materials are not directly managed, i.e., with containers that include a discrete number of input items.

The design of containers for the production cell may also vary. Some containers are specialized for the individual equipment whereas others are identical and used for all equipment in the factory.

2.3.2 Output Item Management

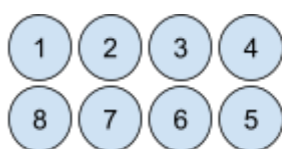
Output items also come in many shapes and sizes. Since the output may both be materials and finished items, or it may be scrap materials, it is recommended to keep these separated in different locations. This way the container content is not mixed and reduces the need for manual operations to split materials again.

2.3.3 Location Management in Container

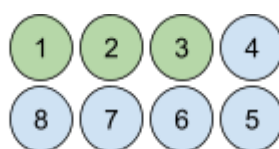
In some containers, the materials are fixed in place with a fixture with a recognizable or distinct pattern. If using such a pattern, equipment can calculate the individual item's placement. For containers without a fixture, the items may move around during transport and the method described here is not applicable.

If a given container has location management, the number of items in the container is known. As the container does not have to be full when delivered to the equipment, it is necessary to know where the items are placed within the container. A simple method for this is to numerate each location within the container and then pick them backwards (largest numerical location first). This ensures that no matter the number of items in the container, the items will always be placed from numerical location 1 to the location matching the number of items in the container.

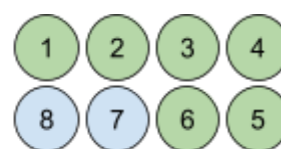
The following example shows that using a container with a fixture and the aforementioned location management strategy, it is possible to determine the location of the items by only knowing the number of items in the container.



Pattern example with 8 locations in the container

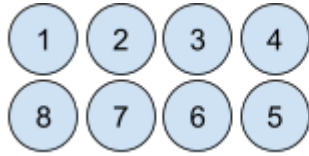


Example with 3 items

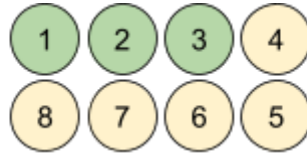


Example with 6 items

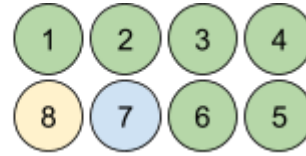
The same method also works if a container contains two distinct kinds of materials. The second material will work on location numbers reversed. This is useful for having both raw items and finished items in the same container during processing.



Pattern example with 8 locations in the container

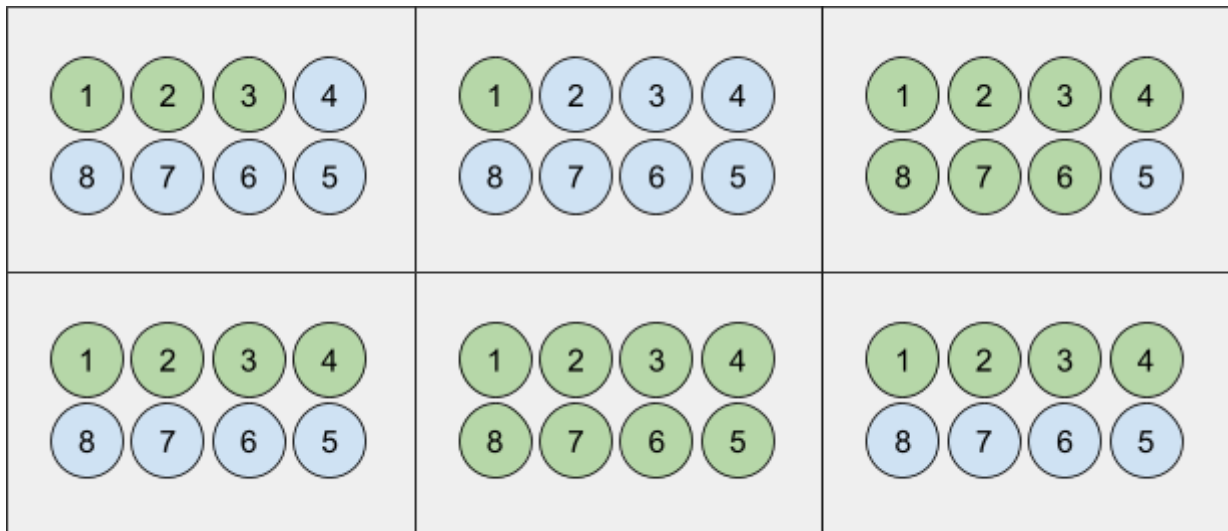


Example with 3 raw items (green) and 5 finished items (yellow)



Example with 6 raw items and 1 finished item (yellow). The remaining spot could be a raw item being processed leaving a spot for the finished item

Using this location management strategy to place and remove items in containers within the whole factory (where relevant) makes it possible to introduce automated equipment that can use the containers even with future upgrades of the factory.



Container with six sub-containers that utilize the location management strategy.

If a container has many raw items (i.e., more than two kinds of materials), this method will only work if each item is placed in a container within the container. For example, a container with six different kinds of raw items used by a production cell to create a finished item. In such a scenario, the container should be split into six containers within. Each of these six containers should then utilize the location management strategy allowing the equipment to pinpoint each individual input item's position at any given time. See the image above for an illustration of the scenario.

3 Communication

See the standard equipment interface [1] for a description of the protocols used for communication with the interface.

3.1 Using the MQTT Protocol

Refer to [4] to see the details for implementing the interface using the MQTT protocol.

Note that this production cell interface overrides the MQTT Protocol with the following values:

- Prefix is set to *production-cells/{type}* where *type* refers to the specific type of production cell. Valid values are *manufacturing*, *assembly*, and *QA*.
- Publisher is set to 'cell' for clients and 'hlc' for the HLC

Example topic for [Batch Instructions Reply](#) for an assembly cell:

production-cells/assembly/cell-124/cell/production/batch-instructions-reply

4 Common Data Types

This section extends the data types presented in the standard equipment interface document [1].

4.1 Segment Parameters

Data type: **SegmentParameters**

A JSON object literal of parameters for the given segment. Written as key-value pairs where the key and value are the parameter's name and value, respectively. Both the key and value must be of type string. Note that the usage is subject to definition in the specific implementations.

Examples

```
{
  "key_1": "value_1",
  "key_2": "value_2",
  "key_3": "value_3",
  ...
}
```

4.2 Segment

Data type: **Segment**

This data type contains information about segments of production orders.

Properties

Property	Type	Description
recipe_id	String	ID of the recipe that should be followed for the given segment.
serial_number	List of Strings or null	Optional. ID for the produced item. This ID can be supplied by the HLC and should be used by the cell when referring to the finished product. Otherwise, the ID can be null. The format of the serial number is chosen by the HLC.
parameters	<i>SegmentParameters</i> or null	Can be empty. The usage is subject to definition in the specific implementations.

Examples

```
{
  "recipe_id": "1234",
  "serial_number": [
    "ABC-1567-001"
  ],
  "parameters": {
    "phone_color": "white"
  }
}
```

```
{
  "recipe_id": "4321",
  "serial_number": [
    "ABC-1567-002"
  ],
  "parameters": {
    "phone_color": "red",
    "screen_size": "12"
  }
}
```

5 Error Types

This section extends the error types presented in the standard equipment interface document [1].

The following list of errors are the ones referenced in this interface. The relevant Error IDs are listed for each telegram of which it is relevant.

Error ID	Error Message	Reason
BATCH_NOT_FOUND	Batch ID not found	The <i>batch_id</i> was not recognized.
INVALID_PARAMETERS	Invalid Parameters	The parameters associated with a segment have invalid entries or some expected properties are missing.
OUTPUT_BUFFER_FULL	Output buffer full	There is not sufficient space in the output buffers for the produced item(s).
PRODUCTION_EXECUTION	Cannot execute production	The cell cannot produce a batch using the provided instructions. This error type should be used if other types do not apply.
RECIPE_NOT_FOUND	Recipe not found	One or more <i>recipe_ids</i> were not recognized. The ID of the unrecognized recipe should be added as a <i>resource_id</i> . If there are multiple, any unrecognized recipe id can be added.
UNKNOWN_REQUEST	Unknown Request	The <i>request_id</i> was not recognized.

6 Production Cell Telegrams

6.1 Batch Instructions (HLC -> Cell)

Telegram Type ID: production.batch-instructions

Description: Telegram containing the information required to execute the next production.

The request is sent on a per-batch basis. The batch contains one or more segments with associated recipes that the production cell will follow. In addition to the recipe, a list of parameters is supplied. The number of parameters and their properties are defined by the recipe.

The recipe and the associated parameters are defined in the configuration of the production cell and HLC, and are not part of this specification. Recipe IDs simply refer to a specific recipe already known by the cell. The recipe itself may cover any action that a production cell may perform. In some cases, a recipe is used to determine how to perform QA on an item, which parts to add to an item or how to manufacture an item.

A list of containers to be worked upon may also be supplied. This is relevant in cases where the production is carried out on supplied items.

Note the production cell must be in state *idle* to receive batch instructions. If the production cell is in any other state, it must reject the request with an *INVALID_STATE* error telegram. See [Batch instructions are sent when a cell is in invalid state](#) for an example.

Properties

Property	Type	Description
batch_id	String	ID of the batch. Variable number of characters.
segments	List of Segment	An array of segments for a receipt.
containers	List of Containers	Optional. Container information for the batch. If this is not provided, the cell should consider all containers on its infeed buffers.

Errors

The following error types can be returned by the cell for this request:

- INVALID_STATE
- INVALID_PARAMETERS
- OUTPUT_BUFFER_FULL
- PRODUCTION_EXECUTION
- RECIPE_NOT_FOUND

Example

```

{
  "header": {...},
  "batch_id": "1951",
  "segments": [
    {
      "recipe_id": "1234",
      "serial_number": [
        "ABC-1567-001"
      ],
      "parameters": {
        "phone_color": "white"
      }
    },
    {
      "recipe_id": "4321",
      "serial_number": [
        "ABC-1567-002"
      ],
      "parameters": {
        "phone_color": "red",
        "screen_size": "12"
      }
    },
    {
      "recipe_id": "1122",
      "serial_number": [
        "ABC-1567-003"
      ],
      "parameters": {
        "phone_color": "white"
      }
    }
  ]
}

```

6.2 Batch Instructions Reply (Cell -> HLC)

Telegram Type ID: production.batch-instructions.reply

Description: Reply sent by the production cell when it has accepted the [Batch Instructions](#).

If an error occurred while processing the batch instructions, an error message should be sent as described in [Batch Instructions](#).

Properties

Property	Type	Description
batch_id	String	ID of the batch. Variable number of characters.

Errors

The following error types can be returned by the HLC for this request:

- BATCH_NOT_FOUND

Example

```
{
  "header": {...},
  "batch_id": "1951"
}
```

6.3 Batch Instructions Complete (HLC <-> Cell)

Telegram Type ID: production.batch-instructions.completed

Description: Once the batch instructions have been completed, this telegram should be sent to the requester to signal that processing has been completed.

The completion telegram should use the same request id as the one provided in the original request (see [Batch Instructions](#)).

Properties

Property	Type	Description
batch_id	String	ID of the batch. Variable number of characters.
timestamp	Date	Timestamp for when the batch was completed.

Errors

The following error types can be returned by the HLC or cell for this request:

- UNKNOWN_REQUEST

Example

```
{
  "header": {...},
  "batch_id": "1951",
  "timestamp": "2023-01-19T09:01:50Z"
}
```

6.4 Production Manifest (Cell -> HLC)

Telegram Type ID: production.manifest

Description: Sent by the production cell when a set of items in the batch has been produced. Note that this telegram may be sent multiple times per batch, and do not indicate that the production cell has finished producing the batch. The production cell must send a [Batch Instructions Complete](#) telegram once the batch is completed, after sending the last manifest.

If multiple manifests are sent for a single batch, information for a container included in a previous manifest does not need to be sent in subsequent telegrams. If the production is stopped manually, a manifest with any finished items must be sent.

Properties

Property	Type	Description
batch_id	String	ID of the batch.
containers	List of Container	Summary of containers in which the products were placed for this batch.

Errors

The following error types can be returned by the HLC or cell for this request:

- BATCH_NOT_FOUND

Example

```
{
  "header": {...},
  "batch_id": "1951",
  "containers": [
    {
      "grai": "01234567890123456789",
      "container_type": "TRAY",
      "contents": [
        {
          "item_id": "10243-001",
          "quantity": "1",
          "properties": {
            "place": "1"
          }
        },
        {
          "item_id": "10243-004",
          "quantity": "1",
          "properties": {
            "place": "2"
          }
        }
      ]
    }
  ],
  "grai": "01234567890121212121",
}
```

```
"container_type": "TRAY",
"contents": [
  {
    "item_id": "1024123001",
    "quantity": "1",
    "properties": {
      "place": "1"
    }
  }
]
}
```

7 Scenarios and Examples

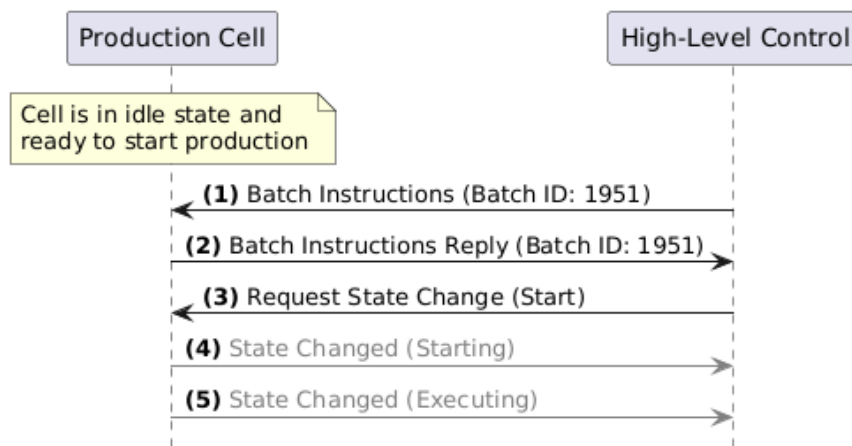
This section provides sequence diagrams which define the order in which telegrams are sent between the HLC and production cell in various scenarios. These sequences must be followed by the HLC and production cell.

Note that the scenarios may include telegrams that are not directly specified in this document. Those telegrams can be found in the standard equipment interface [1].

7.1 Production

7.1.1 Start Production Batch

This scenario illustrates the flow for instructing the production cell what to produce in a batch and starting the production.

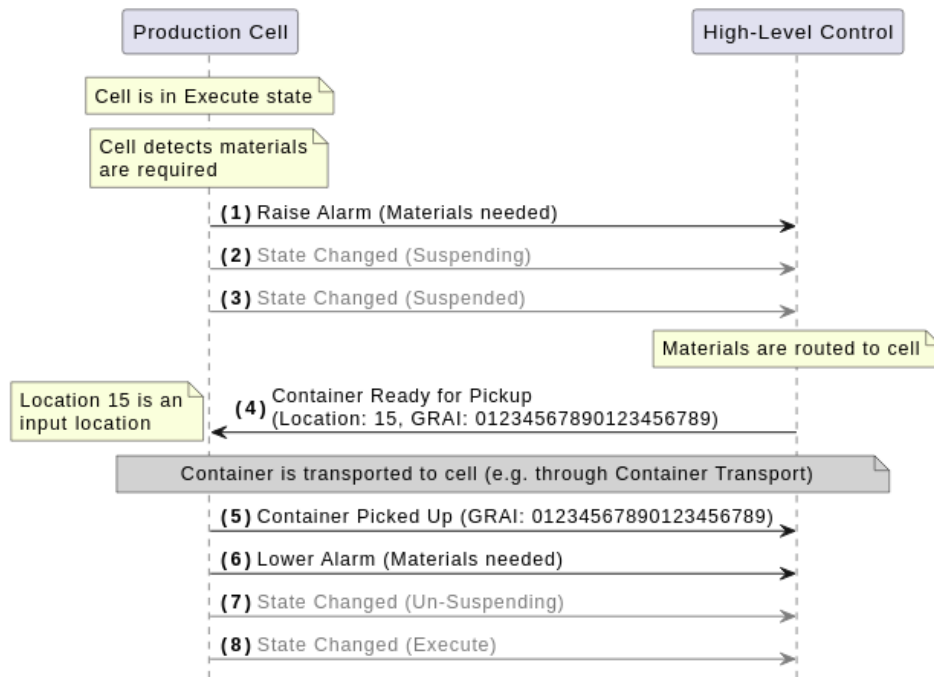


- (1) [Batch Instructions](#) are sent to the production cell detailing what should be produced, the batch id, expected sequence numbers and more.
- (2) The production cell sends a [Batch Instructions Reply](#) to acknowledge that the instructions have been accepted.
- (3) The HLC sends a *Request State Change* to start the production of the batch.
- (4) Production cell starts production.
- (5) Production cell has started the production of the batch.

7.1.2 Replenish with Materials

This scenario illustrates the flow for the production cell requesting materials used for the production of a batch. In this example, the production cell raises an alarm that triggers the HLC to route the materials to the cell. Note that the cell has already started the production of the batch.

Note that the *Item/Container at Location* messages have been omitted from this example.



- (1) An alarm is raised by the production cell after it detects that insufficient materials are present for the production.
- (2) The production cell suspends the current production, as it requires more materials to continue.
- (3) The production cell has been suspended.
- (4) The HLC sends a *Container Ready for Pickup* telegram to inform the cell that the required materials are available for pickup.

The container(s) can be transferred to the production cell through several means (see *Container Transport* and *Resource Access*). The details for these operations are not described in this example. After the transfer is complete, the flow shown above continues.

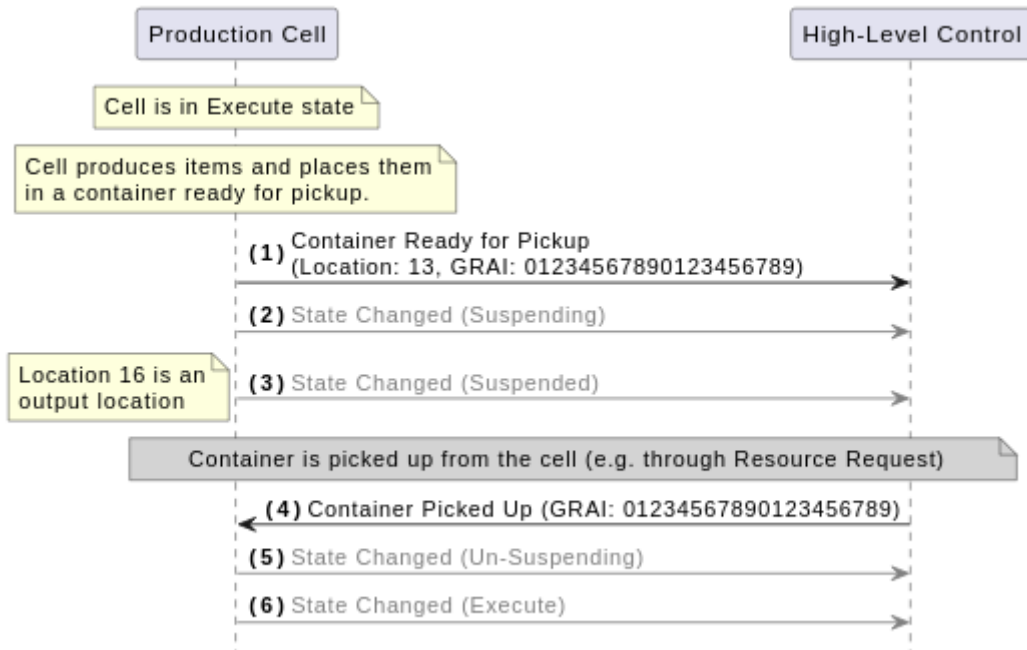
- (5) The production cell sends a *Container Picked Up* telegram to signal that the container has been picked up and is being used by the cell.
- (6) The production cell lowers the alarm it raised earlier as it can continue with the production of the batch.
- (7) The production cell resumes the production of the batch.
- (8) The production cell is producing the items in the batch.

7.1.3 Item(s) Produced and Container Ready for Pickup

This scenario illustrates the flow for the production cell completing the production of one or more items, which are placed in a container. This container is then ready for pickup by the HLC.

The production cell enters a suspended state in this example while it is waiting for the container to be picked up. However, the cell can continue the production of items while the container is pending pickup, and multiple containers can be prepared if possible (e.g. through temporary buffers or multiple output buffers).

Note that the *Item/Container at Location* messages have been omitted from this example.



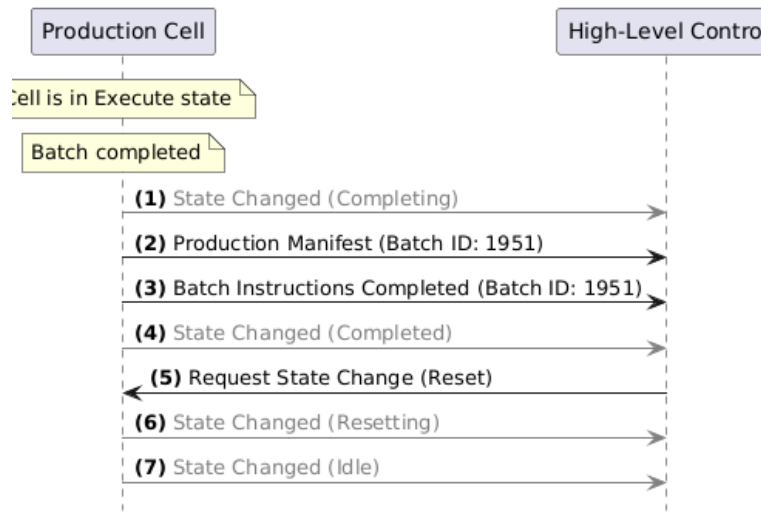
- (1) The production cell signals to the HLC that a container is pending pickup through the *Container Ready for Pickup* telegram.
- (2) Because the cell cannot continue the production of items until the container is removed, it enters a suspended state. Note that this is not required for all production cells if they are capable of continuing.
- (3) The cell enters the suspended state.

The container(s) can be transferred from the production cell through several means (see *Container Transport* and *Resource Access*). The details for these operations are not described in this example. After the transfer is complete, the flow shown above continues.

- (4) The HLC sends a *Container Picked Up* telegram to signal that the container has been picked up and is being transported away from the cell.
- (5) As the container has been removed, the cell can continue its production.
- (6) The cell continues the production of the batch.

7.1.4 Production Completed

This scenario illustrates the flow for completing a batch after the final container has been picked up. The HLC will in this example immediately request the cell to restart to prepare it for a new batch.

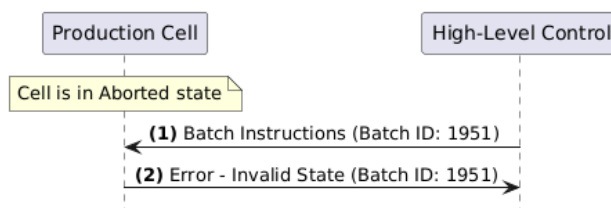


- (1) The cell has finished the production of all items in the batch and enters the Completing state.
- (2) A [Production Manifest](#) is sent to the HLC with details about the batch, including which items were produced and which containers were used.
- (3) A [Batch Instructions Complete](#) is sent to the HLC to notify that all segments in the batch have been completed.
- (4) The cell enters the Completed state.
- (5) The HLC requests the cell to reset in order to prepare it for new batches.
- (6) The cell resets.
- (7) The cell is idle and ready for new batches.

7.2 Production Error Scenarios

7.2.1 Batch instructions are sent when a cell is in invalid state

This scenario illustrates what happens if batch instructions are sent to a production cell when the cell is not in a state where it is able to receive such a request.



- (1) The HLC sends [Batch Instructions](#) to the cell.
- (2) The cell responds with an *invalid state* error as it is not in state idle.