Fraunhofer IML

Fraunhofer IOSB

SWAP-IT: A Scalable and Lightweight Industry 4.0 Architecture for Cyber-Physical Production Systems

An Integration Guide

https://github.com/iml130/swap-it-integration-guide

@INPROCEEDINGS{9926665,
author={Lünsch, Dennis and Detzner, Peter and Ebner, Andreas and Kerner, Sören},
booktitle={2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)},
title={SWAP-IT: A Scalable and Lightweight Industry 4.0 Architecture for Cyber-Physical Production Systems},
year={2022},
volume={},
number={},
pages={312-318},
doi={10.1109/CASE49997.2022.9926665}}

Abstract: In recent years, various abstract and practical architectures have been proposed in the context of Industry 4.0. While these architectures focus on different aspects, their common goal is to facilitate the transformation of a static production into a flexible, resilient, interconnected cyber-physical production system (CPPS). However, reviewing those reveals that a general procedure for applying those architectures is missing. In this paper, we propose a modular, scalable and lightweight architecture utilizing simplified semantic information models. We also present an integration guide that helps factory owners and process engineers to apply the proposed architecture. Furthermore, we also show how the factory operators can make architectural decisions according to their needs. This approach will help speed up the application of the architecture for the realization of a modular and scalable CPPS.

Authors

Dennis Lünsch, Fraunhofer IML
Peter Detzner, Fraunhofer IML
Andreas Ebner, Fraunhofer IOSB
Dr.-Ing. Sören Kerner, Fraunhofer IML

Contact

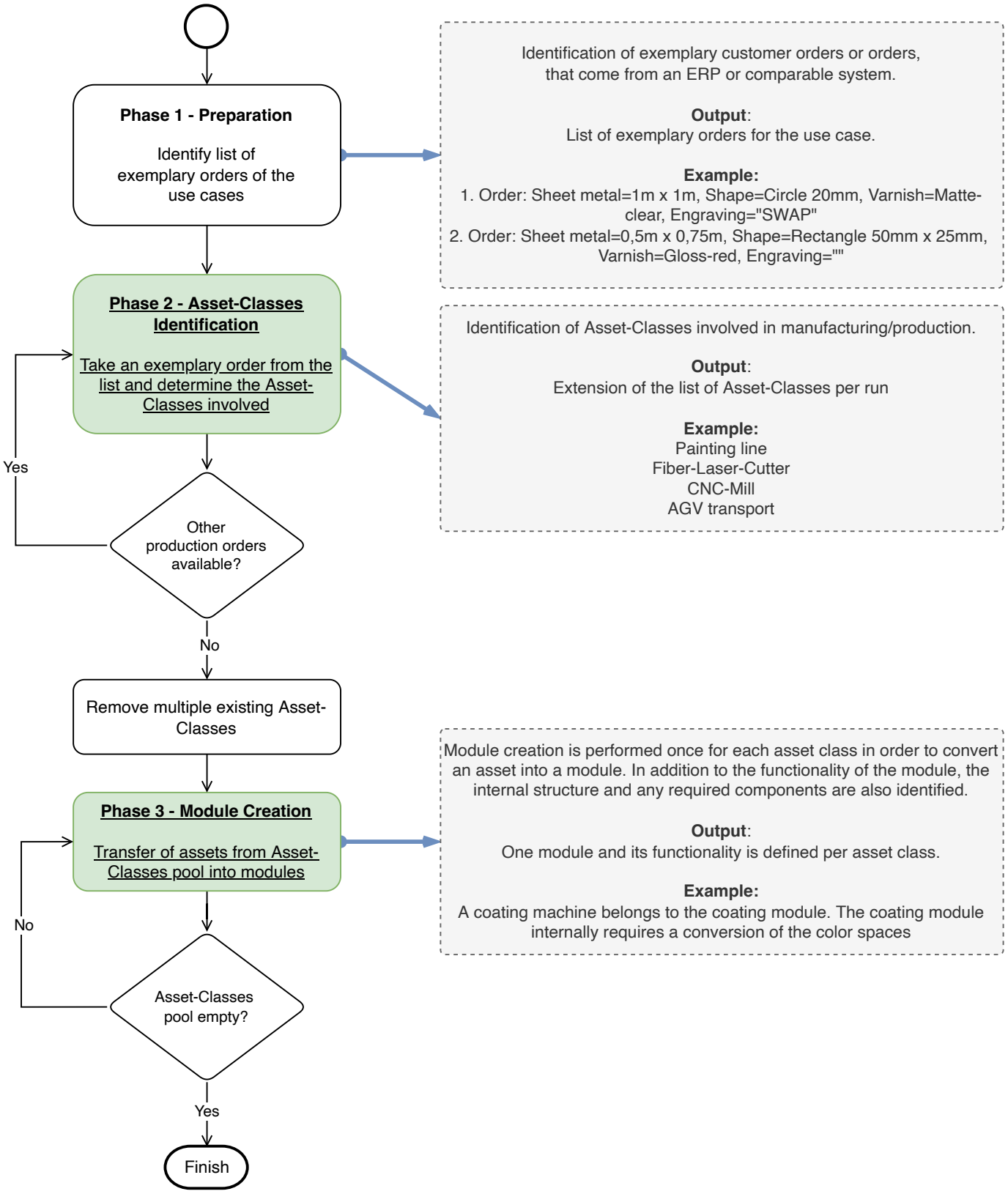
dennis.luensch@iml.fraunhofer.de
peter.detzner@iml.fraunhofer.de
andreas.ebner@iosb.fraunhofer.de
soeren.kerner@iml.fraunhofer.de

DOI

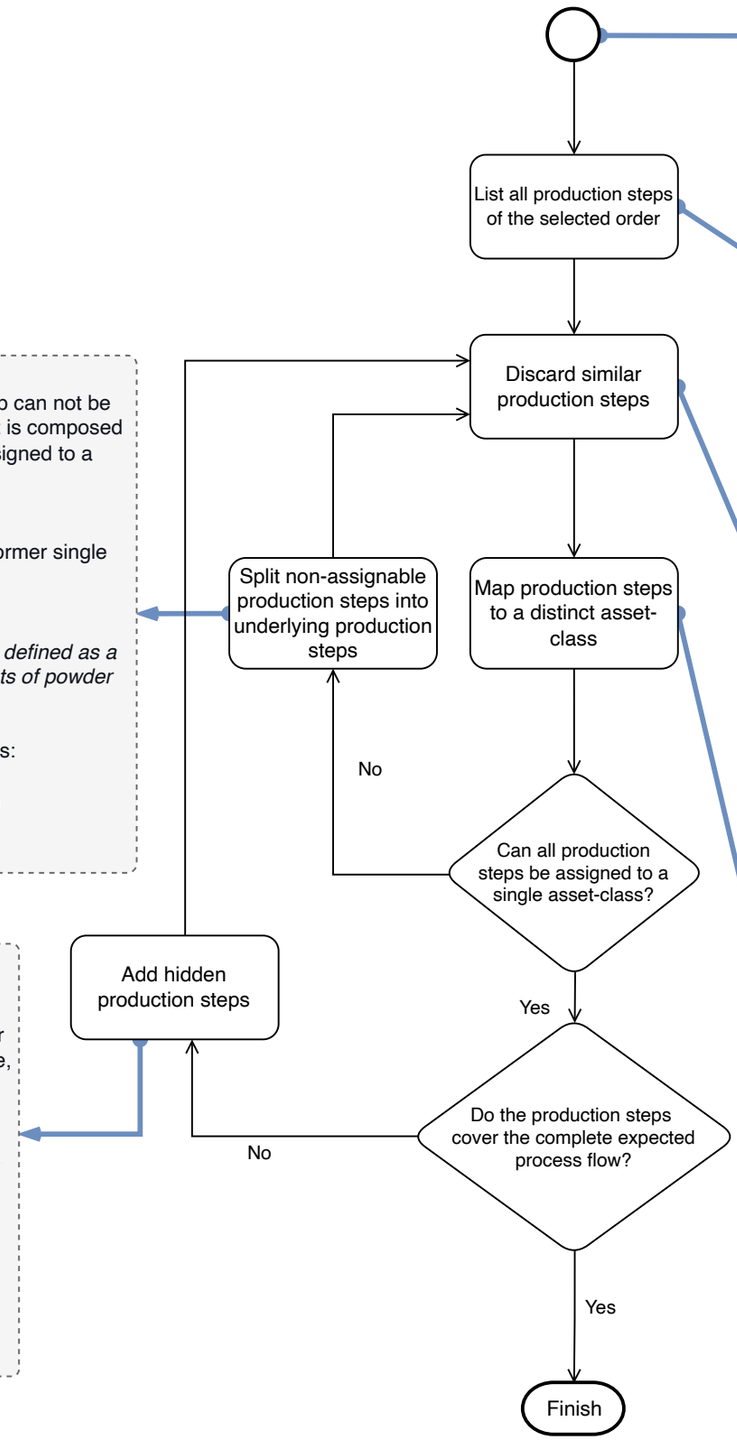
10.1109/CASE49997.2022.9926665

1.0.1 • October 2022

Overview


Phase 1 - Preparation
Identify list of exemplary orders of the use cases
Identification of exemplary customer orders or orders, that come from an ERP or comparable system.
Output:
List of exemplary orders for the use case.
Example:
1. Order: Sheet metal=1m x 1m, Shape=Circle 20mm, Varnish=Matte-clear, Engraving="SWAP"
2. Order: Sheet metal=0,5m x 0,75m, Shape=Rectangle 50mm x 25mm, Varnish=Gloss-red, Engraving=""
Phase 2 - Asset-Classes Identification
Take an exemplary order from the list and determine the Asset-Classes involved
Identification of Asset-Classes involved in manufacturing/production.
Output:
Extension of the list of Asset-Classes per run
Example:
Painting line
Fiber-Laser-Cutter
CNC-Mill
AGV transport
Yes
Other production orders available?
No
Remove multiple existing Asset-Classes
Phase 3 - Module Creation
Transfer of assets from Asset-Classes pool into modules
Module creation is performed once for each asset class in order to convert an asset into a module. In addition to the functionality of the module, the internal structure and any required components are also identified.
Output:
One module and its functionality is defined per asset class.
Example:
A coating machine belongs to the coating module. The coating module internally requires a conversion of the color spaces
No
Asset-Classes pool empty?
Yes
Finish

Phase 2 - Asset-Classes Identification



Identification of exemplary customer orders or existing orders from e.g. an ERP system.

Input:
Exemplary order from the use case.
For reference, see "Integration guide overview"

Example:
Order: Metal sheet=1mx1m, Form=circle 20mm, Paint=matt clear, Engraving "SWAP"

A production process consists of one or many production steps. A production step can be, among others, an assembly, a transport, or an optimization step.

Output:
List of production steps

Example:

1. Paint
2. Cut (Cutting segments from sheet metal)
3. Engrave (Add engraving to the segments)
4. Paint

An initially defined production step can not be mapped to a single asset-class but is composed of two substeps that can be assigned to a separate asset-class.

Output:
List of production steps (from a former single production step)

Example:
Assuming that "coating" has been defined as a partial step, but in practice consists of powder coating + oven

Splited production steps:

Paint-Powder coating
Paint-Oven

Production steps can occur more than once in a process. In this step, the previous production step list is condensed. The resulting list contains each production step type only once.

Output:
List of production steps

Example:
In the example scenario, the substeps "Paint" occurs two times and is therefore removed.

1. Paint
2. Cut (Cutting segments from sheet metal)
3. Engrave (Add engraving to the segments)

In the first analysis, only "direct" production steps were considered. Hidden production steps required for transport, handling, quality assurance, etc., are maybe still missing.

Output:
An extended list of production steps

Example:
Transport metal sheet
Measurement of metal sheet
Pick & Place operations
Surface inspection
...

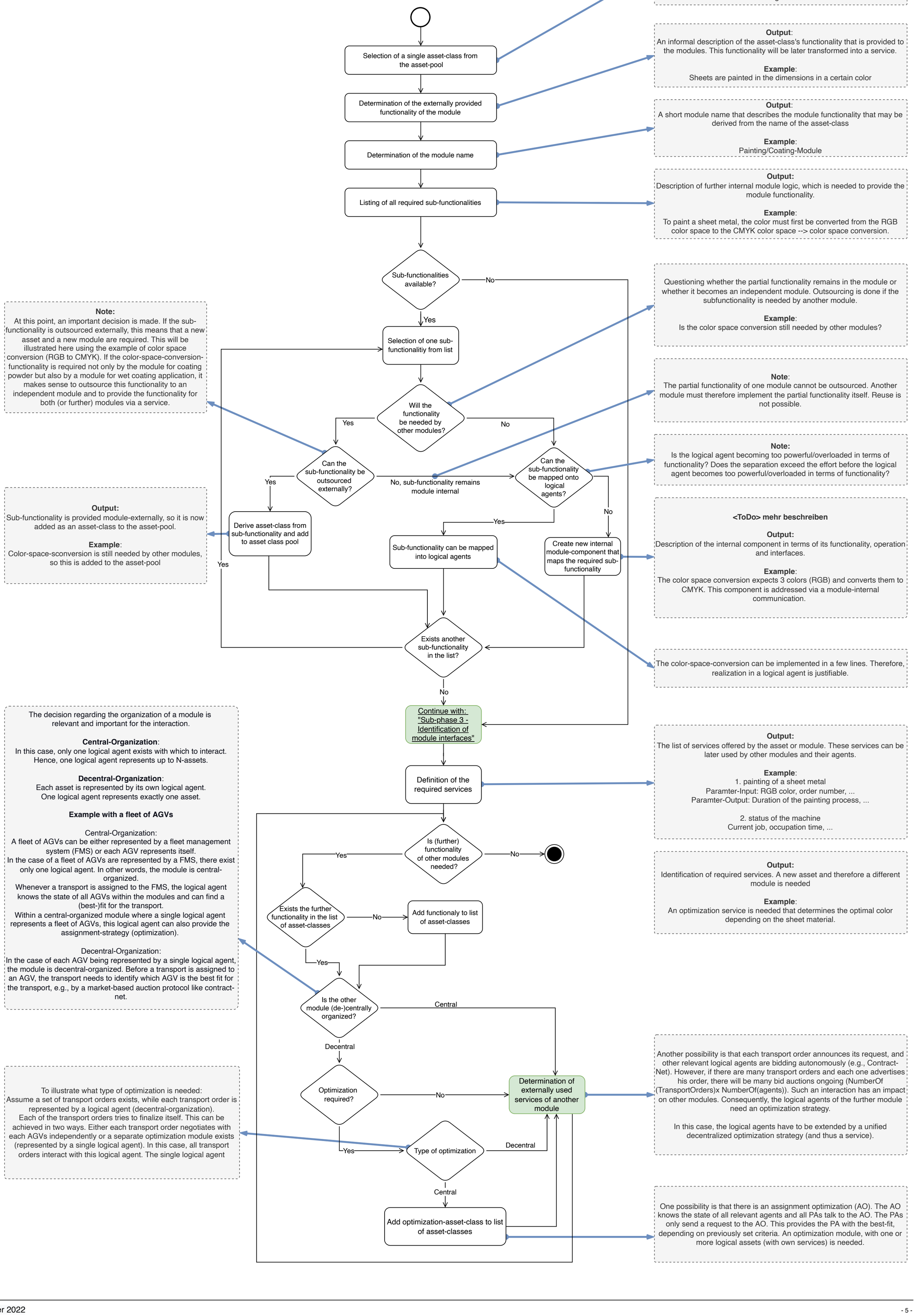
During the order execution, each production step must be executed. In this step of the integration guide, the production steps are assigned to types of the real world production resources.

Asset-classes: Production or manufacturing assets are required for the execution of the production steps. Asset-classes summarize all assets with the same functionality, possibly from different manufacturers.

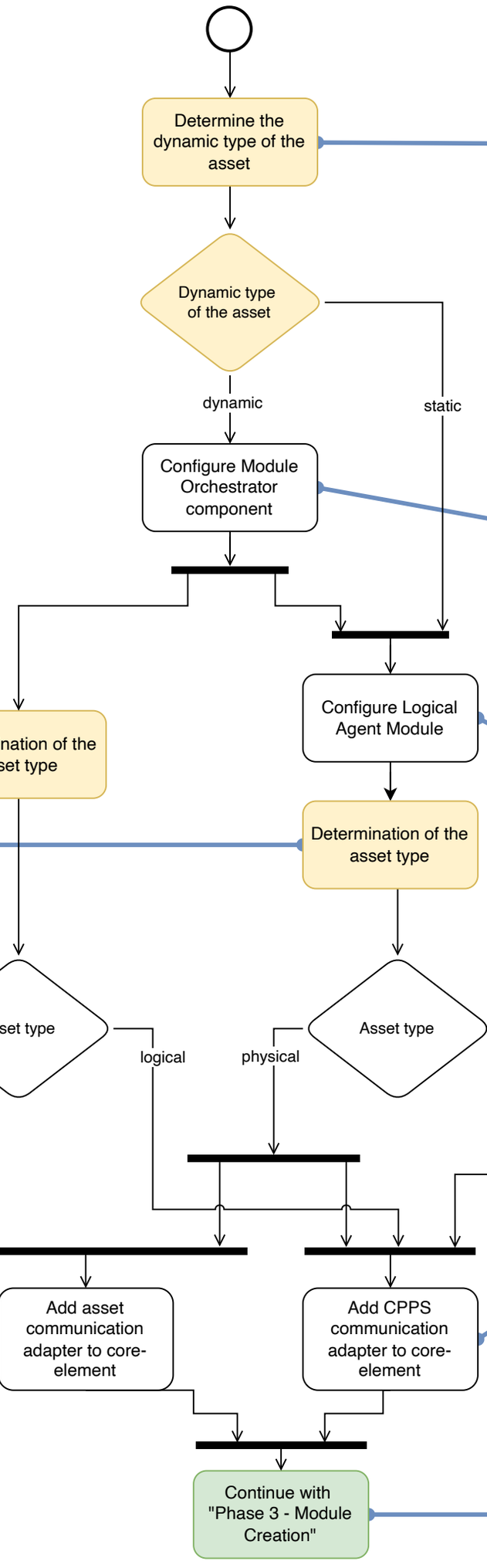
Output:
Assignment of production steps to asset classes.

Example:
Asset-Class | Asset
Paint | Painting Line
Cut | Fiber-Laser-Cutter
Engrave | CNC-Mill

Phase 3 - Module Creation



Sub-phase 3 - Identification of module interfaces



Determination of the dynamic type of an asset:

Definition "dynamic":

The following decision tools can be applied to identify a dynamic asset:

- The number of assets is unknown before startup and may change dynamically at system runtime.
- Runtime-dependent parameters are necessary to create the logical agent.
- The lifetime is limited to one operation (e.g., execution of an order). After this time, the asset is no longer needed by the system and is removed.

Definition "static":

The following decision tools can be applied to identify a static asset:

- The Logical Agent can be started from the outside with the asset without any configuration. No runtime-dependent parameters are required.
- The number of assets is known before the system is started and/or will not change at runtime.
- The asset's life is not limited or tied to individual operations (e.g., machine).

Output:
The determined dynamic type of the asset

Dynamic assets require a module orchestrator that spawns the Logical Agent based on the Orchestrator Information Model (OIM). An OIM consists of a set of parameters needed to launch the Logical Agent.

Output:
Module Orchestrator added as a component to the Module

Logical Agents serve as an abstraction and manage one or many assets.

Output:
Logical Agent added as a component to the Module

Determining the type of asset provides information about which interfaces are required. In the case of physical assets, an asset communication interface to the respective asset is required besides the interface for CPPS communication. An identification of the communication paths via the logical agent and, if necessary, via the required orchestrator is the goal of this and the subsequent steps. According to RAMI 4.0, the type of an asset can be either physical (e.g., an AGV) or logical (e.g., a transport order)

Physical Asset:
A physical asset can be both material and immaterial. Thus, a physical asset can be for e.g., a machine or a software component. However, every physical asset has a possibility for communication and interaction.

Logical Aasset:
Logical assets cannot communicate by themselves, and the
representation needs to be realized by a LA. Nevertheless, the asset can represent a physical entity, for e.g., a product as a production order that is currently in the manufacturing process.

Output:
Added communication interface(s) to the component

An asset communication interface is added to the module to communicate with the asset. The type of communication is specified by the asset. This can be different from the CPPS communication.

Output:
Added asset communication adapter to the component

A CPPS communication adapter is added to the module for communication with other modules. Services can be offered via the CPPS connection and other services of other modules can be used.

Output:
Communication adapter for CPPS communication added to the module.

Result:

1. Need for Module Orchestrator to launch Logical Agents identified.
2. Communication interfaces of the Logical Agent (and Module Orchestrator) are determined and added to the Module.