System-of-Systems Description (SoSD) – Autonomous workflow Use case VTC

**Abstract**

This document defines the System-of-Systems Description of the Use Case developed at Volvo Truck Company inside the Far-Edge project.

It will contain an abstract high-level view, describing the System-of-Systems main functionalities and architecture. This is an ongoing work, so it is subject to further changes as the project evolves.

The architecture presented will enable a smooth transition from a traditional manufacturing ISA-95 organization to an industry 4.0 structure in a factory. The architecture contains part of the MES layer used in ISA-95 and presents a new concept for the workflow where the product becomes the main actor.

Table of contents

[1. System of Systems Overview 3](#_Toc377099507)

[2. Systems 5](#_Toc377099508)

[3. Use-cases 5](#_Toc377099509)

[4. Diagrams 7](#_Toc377099510)

[5. Non-functional requirements 10](#_Toc377099511)

[6. Security 10](#_Toc377099512)

[6.1. Security Objectives 11](#_Toc377099513)

[6.2. Assets 11](#_Toc377099514)

[6.3. Non-technical Security Requirements 11](#_Toc377099515)

[7. References 11](#_Toc377099516)

[8. Revision history 11](#_Toc377099517)

[8.1. Amendments 11](#_Toc377099518)

[8.2. Quality Assurance 12](#_Toc377099519)

1. System of Systems Overview

The main objective of the SoS is to enable a workflow in which the factory workstations can perform their operations autonomously, without connection to a MES layer.

An overview of the proposed architecture for this workflow can be seen in figure 1. It is part of a use-case in the FAR-EDGE project.

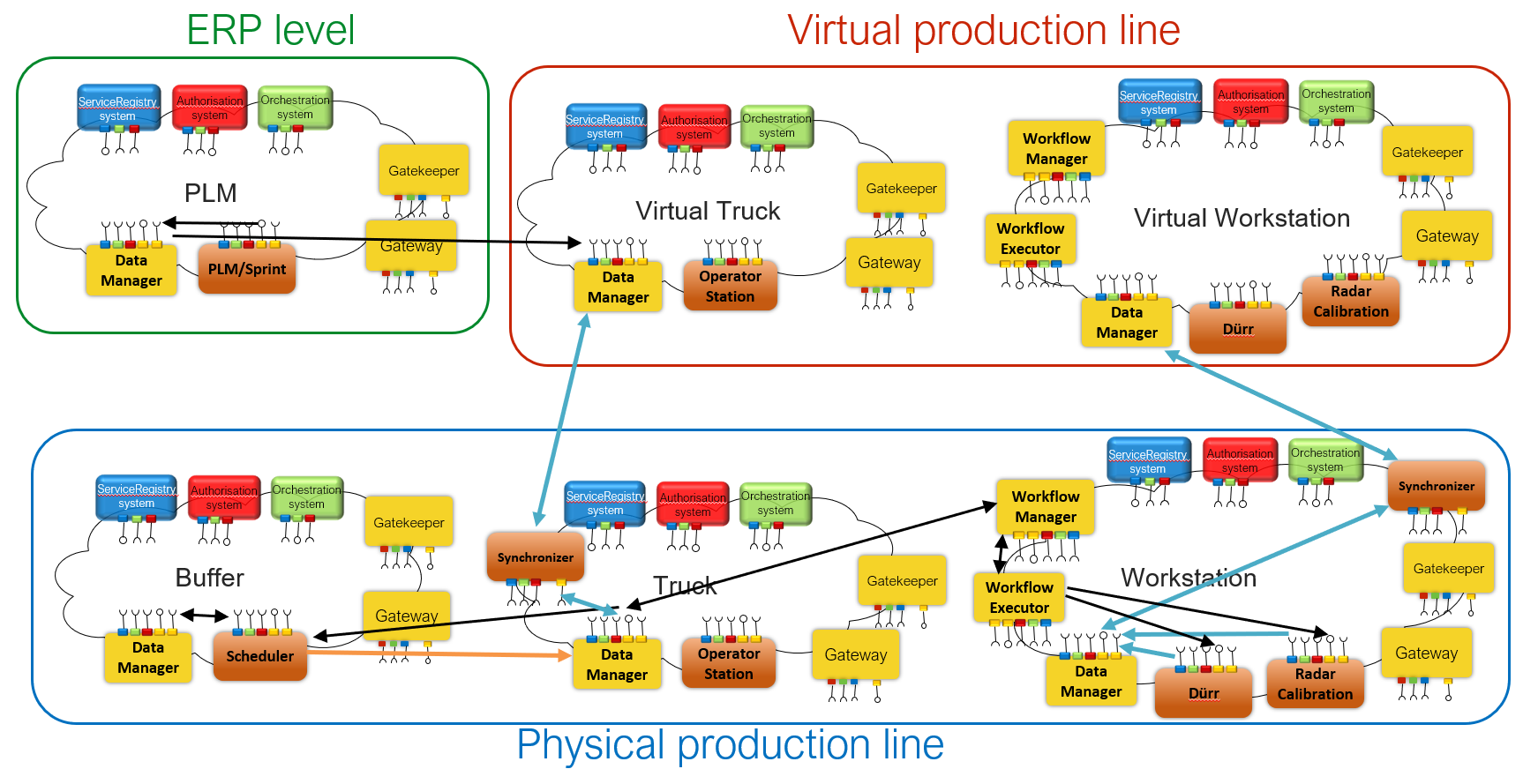


Figure 1. SoS Architecture

The data to carry out the operations in the product, production order, comes from the ERP/MES layer. The production order will be received by the PLM/Sprint system, inside the PLM Local cloud, which works as an adapter to the factory MES layer. Inside the local cloud, the Data Manager will be the system in charge of storing the production order, and then transmitting it.

The production order data is then passed to the Virtual Product Local Cloud, to its Data Manager system. The set of Virtual Product and Virtual Workstations Local Cloud form what is called the Virtual Production Line. The Virtual Product will synchronize with the Product (Truck) Local Cloud once it physically exists and is scheduled for manufacturing, through the Synchronizer system. In this way, the Virtual Product local cloud operates as the virtual twin and it has the same systems as the real one, but it will be physically deployed in the factory network.

To carry out the operations, the workstations need to have access to the production order. Every time the product enters a Workstation Local Cloud, the production order can be transmitted to the Workflow Manager system, which will transform it into a State Machine, needed by the Workflow Executor to command the different equipment of the workstation. Once all operations are finished, the Workflow Manager will inform of the results to the Truck Local Cloud, so it can update the production order stored in the Data Manager system and continue to the next workstation. In this architecture the product is the main actor, responsible of the production order. As the Truck must be driven by an operator, it has an Operator Station system that will present the information in a human readable format.

When the manufacturer of the machines used in the workstation is interested in data from their use, the system corresponding to that machine, Dürr or Radar Calibration, can store its operations values in the Data Manager system of the corresponding Workstation, which will eventually be synchronized with the Virtual Workstation. The manufacturer can request the data to the Virtual Workstation without affecting the real workflow at the factory.

Before the product enters the workstations, it is usually placed in a Buffer. The Buffer local cloud contains a system called Scheduler, in charged of optimizing the order in which the products enter the workstation. The Truck will transmit his production order when he arrives to the Buffer, and the Scheduler is then able to change this production order when he obtains a better scheduling of the product workflow.

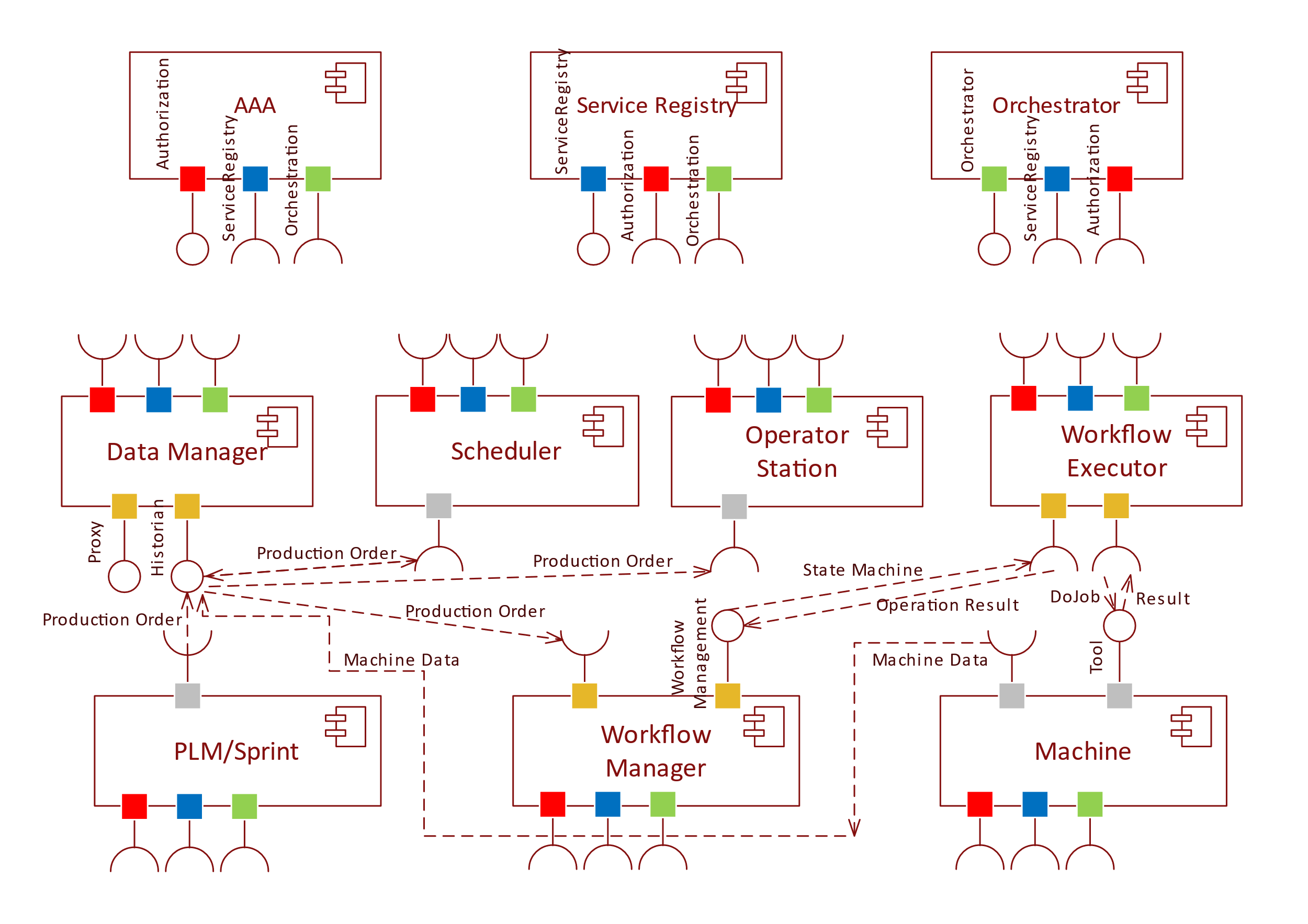


Figure 2. UML component diagram

Figure 2 includes a formal diagram containing the Systems inside our System of Systems, and their structural relationships. For ease of view, the connection between the mandatory core systems and the application or support core systems has not been drawn, but a color schema has been used to show these relations. Also, the intercloud communication through the gateway are not included, as we expect them to be transparent to the application systems.

1. Systems

In Table 1 are the pointers to the SysD documents of the systems used. The documents which include Arrowhead in their title correspond to support core systems of the Arrowhead Framework, which could be accepted or in the state of proposal. The rest of the systems are Application Systems part of this type of SoS.

Table 1 Pointers to the SysD documents

|  |  |
| --- | --- |
| **System name** | **Path** |
| Data Manager | \Arrowhead DataManager G4.0 SysD.docx |
| Workflow Manager | \Arrowhead WorkflowManager G4.0 SysD.docx |
| Workflow Executor | \Arrowhead WorkflowExecutor G4.0 SysD.docx |
| PLM/Sprint | \PLM\_Sprint G4.0 SysD.docx |
| Scheduler | \Scheduler G4.0 SysD.docx |
| Operator Station | \OperatorStation G4.0 SysD.docx |
| Dürr | \Dürr G4.0 SysD.docx |
| Radar Calibration | \Radar Calibration G4.0 SysD.docx |

1. Use-cases

The main use-cases identified for this SoS are:

* The normal behavior in which a Production Order goes all the way from the PLM/Sprint System to the Workstation.
* The change in Production Order when the Scheduler optimizes the process.
* The synchronization of the data of the Real production line, the local clouds of the Truck and the Workstations, with the Virtual production line, which are the local clouds of the Virtual Truck and Virtual Workstations.

They are represented in a UML use-case diagram in Figure 3. The main actors of these use-cases are identified as the different local clouds. Such topology enables communications between all the actors following the Arrowhead framework, while there is only one actor able to communicate with the PLM/Sprint.

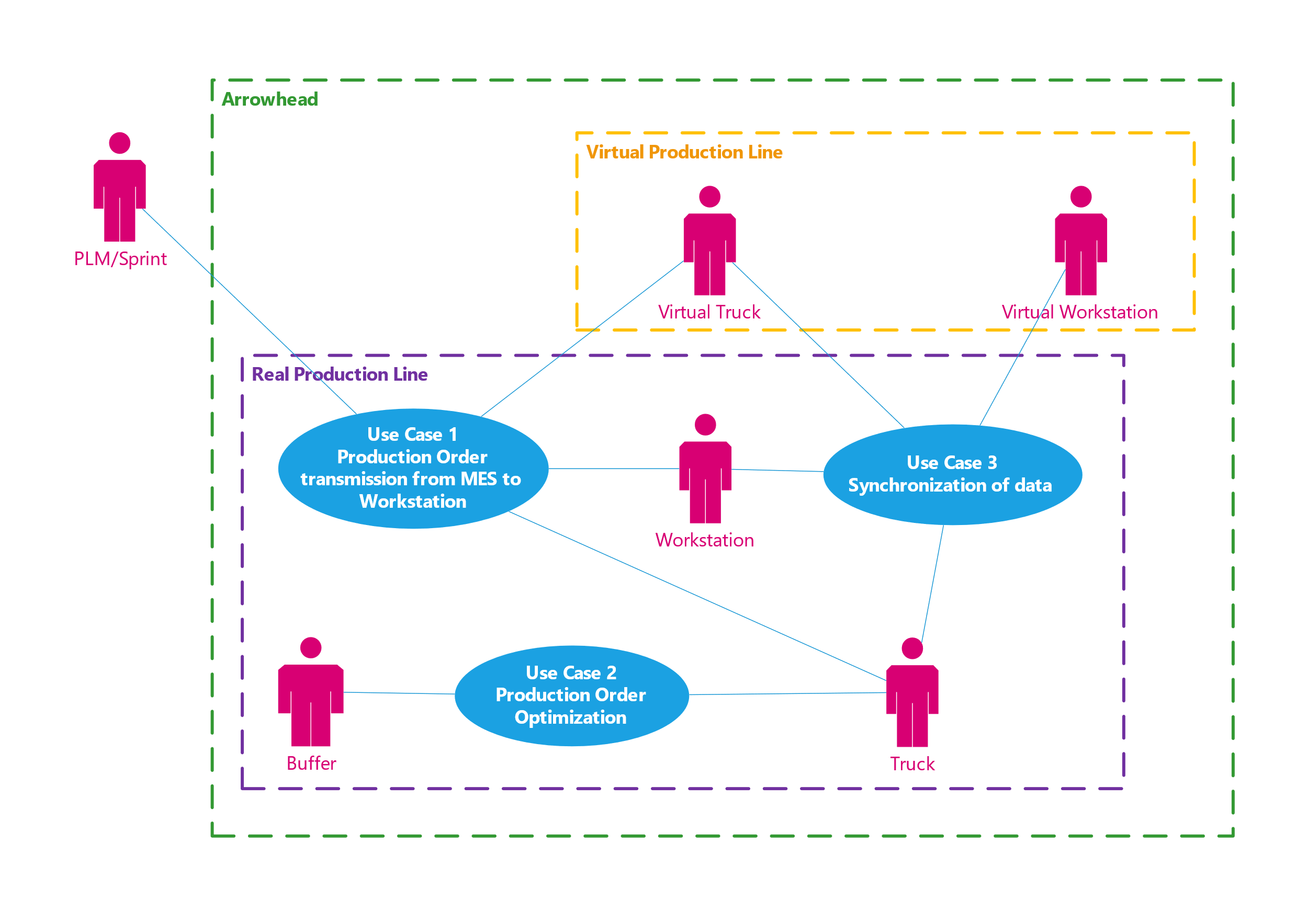


Figure 3. UML use-case diagram

The use-case 1 is described in Table 2. This is the basic flow of data in the SoS, and it fulfills the basic necessary functionalities.

Table 2 Use-case description table

|  |
| --- |
| **Name of the Use-case**: Production Order from MES to Workstation |
| **ID**: 1 |
| **Brief description**:  In this use-case a production order will go from the factory network, PLM/Sprint, to the Workstation by means of intermediate actors, which will transmit the information without changes and will enable the autonomous operation of the assembly line in case there is a disconnection from the factory network. |
| **Primary actors**: PLM/Sprint |
| **Secondary actors**:  Virtual Truck, Truck, Workstation |
| **Preconditions**:  Initial connection between Factory network, PLM/Sprint, and Virtual Production Line. |
| **Main flow**:  1- The factory PLM/Sprint, executing in the Factory Network, commands a new Production Order for a specific Truck, and it is received at the local cloud by the PLM/Sprint system and saved in its Data Manager.  2- A Virtual Truck local cloud is created, which will receive the production order and store it in its own Data Manager.  3- The real Truck is scheduled to production. The Truck local is created and it will synchronize with the Virtual Truck correspondingly.  4- The Truck moves through the production line and arrives to a Workstation.  5- The Workflow Manager in the Workstation local cloud request the production order, which is converted to a State Machine and sent to the Workflow Executor.  6- The Workflow Executor commands the machines at the workstation to perform the operations according to the State Machine and reports back the results.  7- The Workflow Manager updates the production order accordingly to the results and transmits back to the Truck.  8- The Truck can continue in the production line to the next workstation. |
| **Postconditions**:  At the end of the production line, synchronization with the Virtual Production Line is necessary. |
| **Alternative flows**:  If workstation reports failure in operation, Truck must go to the Repair Workstation. |

The use-case 2 is presented in Table 3. It can be seen as an improvement to the normal flow, with the addition of a different type of workstation that performs scheduling optimizations. This provides additional functionalities to the SoS and becomes an integration point with other use-cases of the FAR-EDGE project.

|  |
| --- |
| **Name of the Use-case**: Production Order optimization |
| **ID**: 2 |
| **Brief description**:  In this use-case a product will enter a Buffer, a location where the products are stored between different workstations. In the buffer, scheduling optimization can be done in real time, making modifications to the production order. |
| **Primary actors**: Buffer |
| **Secondary actors**:  Truck |
| **Preconditions**:  Truck already waiting in buffer and next workstation busy. |
| **Main flow**:  1- A new Truck arrives to the Buffer.  2- The Scheduler connects to the Truck and retrieves the production order, that will be stored in the Data Manager of the Buffer.  3- When two Trucks are waiting for the next workstation, the Scheduler will execute its optimization algorithm.  4- If the result Schedule is different from the one planned in the production order, it will modify the production order.  5- The new production order will be saved in Data Manager and transmitted to the Truck, so the operator can see which Truck to take in the Operator Station system.  6- Once the next workstation is free, the corresponding Truck will be taken in. |
| **Postconditions**:  None |
| **Alternative flows**:  If scheduling algorithm results in the original schedule, the steps 4 and 5 will be skipped. |

The use-case 3 is described in Table 4. It shows the process of synchronization between the Virtual Production and the Physical (real) production line. This is always necessary for products to do at least before they leave the assembly line, but the more often the better. For workstations, this process is necessary if the manufacturer of the machine/tool used in the workstations wants to know data from it. It will probably be executed once a day when the workstation is idle.

|  |
| --- |
| **Name of the Use-case**: Synchronization of data |
| **ID**: 3 |
| **Brief description**:  This use-case describes the process of synchronization between the Physical Production Line, Truck or Workstation local cloud, and the Virtual Production Line, Virtual Truck or Virtual Workstation local cloud. It will merge the data in the Data Manager of the corresponding clouds, so that their information can be retrieved form outside the physical production line. This use-case enables the concept of digital twin inside a factory with the use of the Arrowhead Framework. |
| **Primary actors**: Truck, Workstation |
| **Secondary actors**:  Virtual Truck, Virtual Workstation |
| **Preconditions**:  Connection between Physical production line and Virtual production line. |
| **Main flow**:  1- A synchronization request arrives to the local cloud.  2- The Synchronizer will try to find the corresponding Virtual cloud in the network.  3- When a match is found, the Synchronizer will connect to the Data Manager in the virtual cloud and retrieve the information it has.  4- Then it will compare with the information from the Data Manager in its own cloud. It will update the Virtual cloud with the last version of the data from the physical production line.  5- Finally it will reply to the synchronization request with the result of the synchronization. |
| **Postconditions**:  None |
| **Alternative flows**:  If connection with Virtual cloud is not possible, skip directly to step 5 and answer with an error message. |

1. Behaviour diagrams

The behavior of the SoS can be seen in the UML activity diagram presented in Figure 4.

The UML Activity diagrams are used to model a higher-level business process or processes.

It includes the normal flow of the production order, explained in use-case 1 in the previous chapter, and it adds the other use-cases to expose the complete behavior.

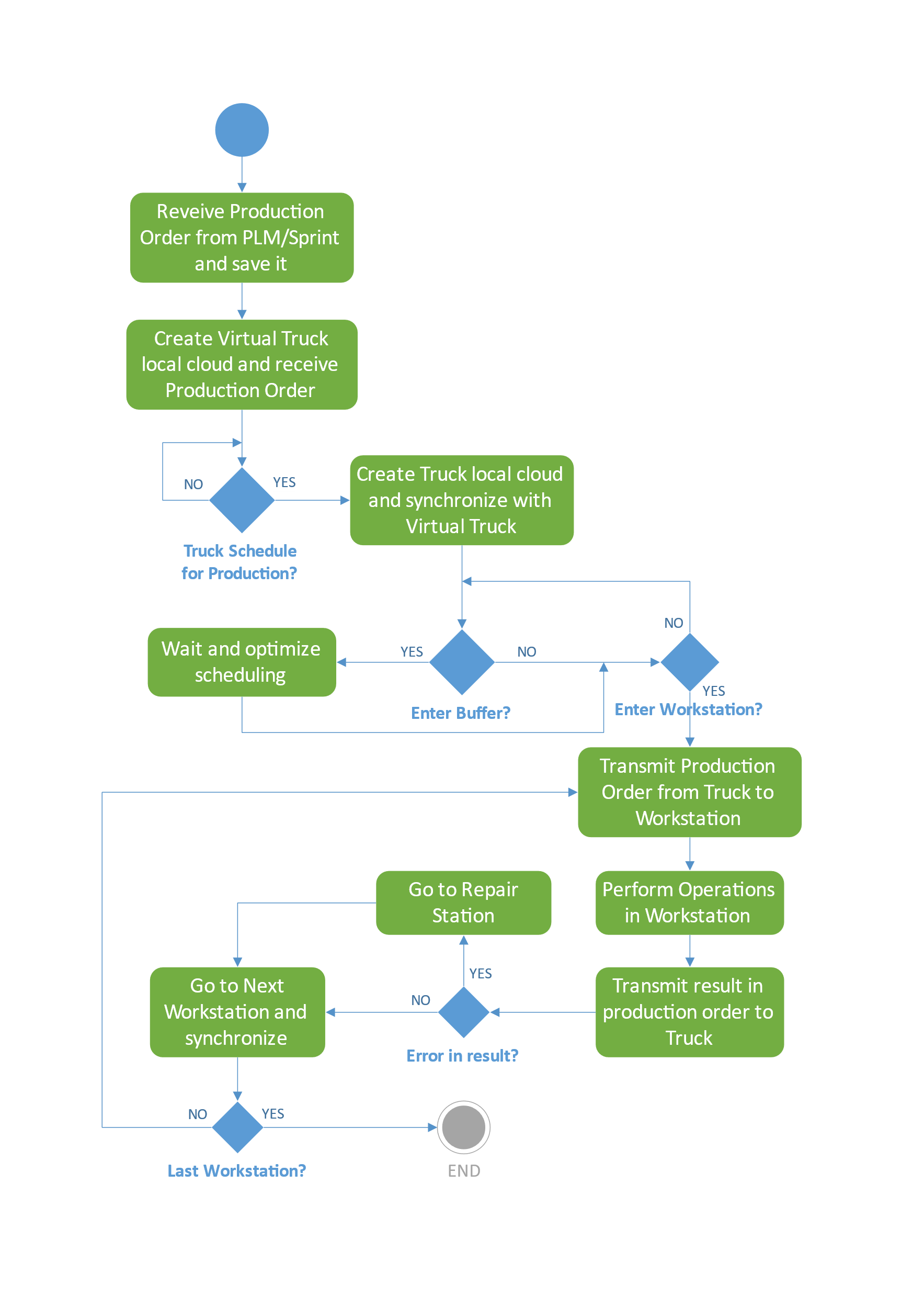


Figure 4. UML Activity diagram

1. Non-functional requirements

As non-functional requirements in this System of Systems, we can have the existing requirements of a production line in a factory, in addition to the ones imposed by Arrowhead.

In the case of a production line, the response time and reliability are key factors, as there are real-time performance requirements that must be met. Security will be explained in the next chapter. Scalability and interoperability are some of the properties that this SoS tries to improve in the existing architecture, as there should be no limit to the number of local clouds that could run in parallel inside a factory.

In our scenario, we expect to only have one local cloud who connects to the MES (PLM/Sprint) network. From the production order, it will create the Virtual local clouds, at an unknow rate depending on the MES configuration. It is expected to have a Truck local cloud being created every few minutes, at the pace the production line can work. This means that we can have many Truck local clouds in the factory, and even a greater number of Virtual Truck local clouds depending on the network back-up rate. Workstation local clouds could be created at any moment, but once the factory set-up is organized, it is not expected to have big changes. It could be more common to see changes inside the Workstations, as they reconfigure for new machines.

1. Security

At this moment of the project the security requirements for this SoS are still unclear. As part of the systems, the Virtual Production line, will be deployed in the factory network, there are probably stablished requirements from the company. There could be more security requirements for the product and workstations, but this have not been stablished yet. Probably a certificate hierarchy will be used [1].

At a system level, the security tools provide by the arrowhead framework, through the Authorization core system will be used [1].

# Security Objectives

In this aspect, for the moment only the basic objectives are required, the *AIC-*triad (availability, integrity, confidentiality). The attribute *availability* ensures that information is available when it is needed. *Integrity* refers to the protection of data against unauthorized modification within a given system. *Confidentiality* seeks to ensure that information can only be read by authorized subjects.

# Assets

As the main asset to protect in this SoS, we have the final product. This local cloud will leave the factory floor and continue through the life of the product, so it presents a big security risk.

1. References

[1] Arrowhead G4.0 generic SoSDD - Core System architecture

1. Revision history

# Amendments

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Date | Version | Subject of Amendments | Author |
| 1 | 2019-02-01 | 0.1 | Creation | Jaime Garcia Represa |

# Quality Assurance

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Date | Version | Approved by |
| 1 | YYYY-MM-DD | 1.0 | Nnnnn Nnnnnnn |
| 2 |  |  |  |