Ecosystem for COllaborative Manufacturing PrOceSses – Intra- and Interfactory Integration and AutomaTION
(Grant Agreement No 723145)

D3.1 Process Modelling Representation and Monitoring Framework

Date: 2018-02-28

Version 1.0

Published by the COMPOSITION Consortium

Dissemination Level: Public

Co-funded by the European Union's Horizon 2020 Framework Programme for Research and Innovation under Grant Agreement No 723145

Document control page

Document file: COMPOSITION_D3.1_v1.0.docx
Document version: 1.0
Document owner: FRAUNHOFER

Work package: WP3 – Manufacturing Modelling and Simulation
Task: T3.1 – Process Modelling and Monitoring Framework
Deliverable type: OTHER

Document status: ☒ Approved by the document owner for internal review
☒ Approved for submission to the EC

Document history:

Version	Author(s)	Date	Summary of changes made
0.01	Junhong Liang (Fraunhofer)	2018-01-15	Initial TOC
0.02	Junhong Liang, Daniela Fissler (Fraunhofer)	2018-02-10	Initial content added to the deliverable
0.03	Alexandros Nizamis, Vagia Rousopoulou (CERTH)	2018-02-12	Added contribution from CERTH about BPMN and DFM integration
0.10	Junhong Liang (Fraunhofer)	2018-02-15	Small fixes for wording and format; Released for internal review
0.11	Junhong Liang (Fraunhofer)	2018-02-19	Addressed comments from ATL;
0.12	Junhong Liang (Fraunhofer)	2018-02-22	Addressed comments from ATOS;
1.0	Junhong Liang (Fraunhofer)	2018-02-22	Final version submitted to the European Commission

Internal review history:

Reviewed by	Date	Summary of comments
Ifigeneia Metaxa (ATL)	2018-02-16	Approved with minor comments
Javier Romereo (ATOS)	2018-02-22	Approved with minor comments

Legal Notice

The information in this document is subject to change without notice.

The Members of the COMPOSITION Consortium make no warranty of any kind with regard to this document, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Members of the COMPOSITION Consortium shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Possible inaccuracies of information are under the responsibility of the project. This report reflects solely the views of its authors. The European Commission is not liable for any use that may be made of the information contained therein.

Index:

1 Executive Summary

This deliverable is a report of the result of T3.1 - Process Modelling and Monitoring Framework. This task involves two main activities: manufacture process modelling with Business Process Model and Notation (BPMN) and the development of a BPMN Monitoring Framework for real-time process monitoring.

To help readers understand the topic better, the document first introduces related background knowledge, such as the concept of BPMN, the advantages of BPMN, software tools related to BPMN, etc.

Following the background introduction, the applicable use cases and the working principle of the BPMN Monitoring Framework is explained. A concrete example from COMPOSITION is used for demonstration.

The relationship between the BPMN Monitoring Framework and the COMPOSITION system is then described. The connection between the monitoring framework and the Digital Factory Model (DFM) as well as the Decision Support System (DSS) is explained. This helps the readers to gain more knowledge about the integration of the monitoring framework in the COMPOSITION system.

After that, the modelling of a manufacture process is presented. The model is based on a Printed Circuit Board (PCB) production line from Boston Scientific Limited (BSL). Details regarding the process and the modelling principles are also introduced.

Finally, the implementation of the BPMN Monitoring Framework is presented in detail. It includes the requirements for the development, tools and technologies used, the implementation details as well as exposed interface.

T3.1 starts at M1 and finishes at M18, which is also the time for submission of this deliverable. As a result, this document serves as a final report of T3.1. Never the less, this does not limit the future activities of adjusting the implemented software to accommodate the new changes and requirements of other COMPOSITION components.

2 Abbreviations and Acronyms

Table 1: Abbreviations and acronyms are used in this deliverable

Acronym	Meaning
BPMN	Business Process Model and Notation
BPD	Business Process Diagram
BSL	Boston Scientific Limited
MES	Manufacture Execution System
SCADA	Supervisory Control and Data Acquisition
API	Application Programming Interface
OMG	Object Management Group
MOF	Meta Object Facility
XML	Extensible Markup Language
XMI	XML Metadata Interchange
GUI	Graphical User Interface
DFM	Digital Factory Model
SOP	Standard Operation Procedures
IIMS	Integrated Information Management System
PCB	Printed Circuit Board
PCBA	Printed Circuit Board Assembly
DSS	Decision Support System
REST	Representational state transfer
HTTP	Hypertext Transfer Protocol
AJAX	Asynchronous Javascript And XML
HMI	Human Machine Interface
ICT	Information and Communication Technology
IoT	Internet of Things

3 Introduction

3.1 Purpose, context and scope of this deliverable

The purpose of this deliverable is to present the result of T3.1 - Process Modelling and Monitoring Framework. T3.1 includes the definition of manufacturing process and domain models and the implementation of a process-oriented monitoring framework. The manufacturing process model will follow the Business Process Model and Notation (BPMN) standard to provide a graphical notation for specifying manufacturing processes in Business Process Diagrams (BPD) and at the same time they will be expressed logic-based. The framework will monitor and process data coming from sensor networks, through the respective automation and control system (e.g. MES, SCADA), and other information management or computer aided tools and relate the data to the process. This way, context-aware reactions to certain (unusual) events or combination of events will be possible. Effective modelling will also enable better designing, engineering, and planning of processes.

3.2 Content and structure of this deliverable

In this deliverable, the activities related to T3.1 are described. In order to give readers a more thorough understanding of the work done in this task, the following chapters are included:

In chapter 4 the basic concept of BPMN as well as the advantages of using BPMN are first introduced. Afterward, available tools for working with BPMN are described. These tools can be classified into three categories: BPMN modeller, BPMN viewer and BPMN engine. As typical example, two toolsets, Activti and Camunda are further introduced.

In chapter 5, the applicable use case for the BPMN Monitoring Framework is first discussed. The working principle of BPMN Monitoring Framework is then illustrated with a detailed example. The purpose of this chapter is to prepare the readers with better understanding of the targeted scenario as well as requirements of the BPMN Monitoring Framework.

In chapter 6 the Digital Factory Model (DFM), a core component in COMPOSITION system developed in T3.2, as well as the Decision Support System (DSS), which is developed in T3.4 are introduced. The connection between the BPMN Monitoring Framework and the DFM and DSS is then presented to the readers for a better understanding of the integration of BPMN Monitoring Framework in COMPOSITION.

In chapter 7, the BPMN model of a manufacture process, namely the PCB production line in BSL, is presented. It also offers detailed explanation for the whole workflow as well as the modelling principles behind.

In chapter 8 the implementation of the BPMN Monitoring Framework in COMPOSITION is presented. In this section, implementation requirements are first summarised. Regarding these requirements, suitable tools and technologies for development are described. Finally, the architecture as well as exposed API of the BPMN Monitoring Framework are presented.

4 Introduction to BPMN

In this chapter, the basic definition and concept of BPMN is first introduced to provide readers background information. Available software tools are then introduced to present the readers an overview of the state-of-the-art development in BPMN domain.

4.1 Concept of BPMN

Business Process Model and Notation (BPMN) is a standard for business process modelling that provides a graphical notation for specifying business processes in a Business Process Diagram (BPD). BPMN is based on a flowcharting technique similar to a state machine diagram.

The BPMN was developed by the Object Management Group (OMG). The primary goal of BPMN is to provide a notation that is readily understandable by all business users, from the business analysts that create the initial drafts of the processes, to the technical developers responsible for implementing the technology that will perform those processes, and finally, to the business people who will manage and monitor those processes. Thus, BPMN creates a standardized bridge for the gap between the business process design and process implementation (BPMN Specification, 2011).

In other words, the objective of BPMN is to support business process management, for both technical and business users, by providing a notation that is intuitive to business users, yet able to represent complex process semantics. Figure 1 shows an example of BPMN diagram.

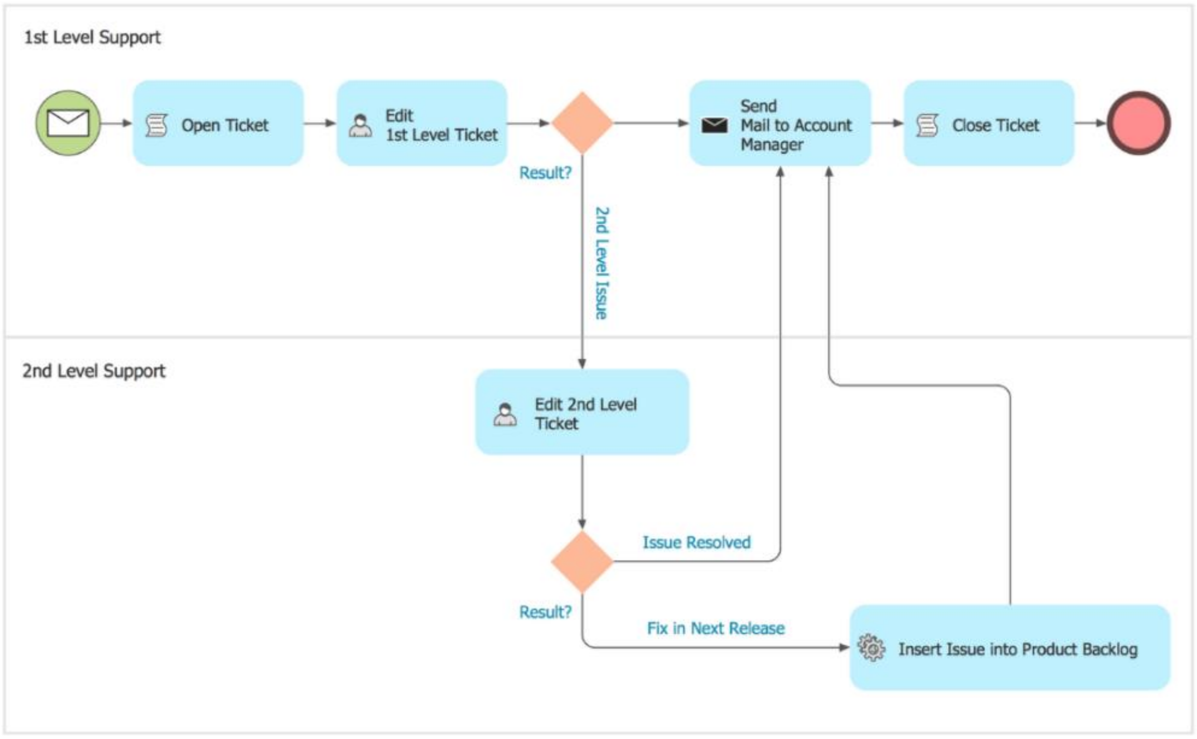


Figure 1 Example of a BPMN diagram describing a trouble ticket system[1]

There are several advantages of using BPMN for model business and manufacture processes:

- Standardisation: BPMN defines MOF-based meta-models for not only process modelling but also diagram interchange (BPMN DI), which makes it very easy to serialize and interchange a BPMN model between application programs using XMI. Besides, BPMN also defines standards for visual representation, which makes it easy to be interpreted consistently by people in different domains. As a result, BPMN has been a favourable choice both in describing processes involving cross-domain knowledge.

[1] http://www.conceptdraw.com/samples/business-process-diagrams-business-process-model-notation

- Simplicity and flexibility: BPMN offers very simple and intuitive notations for describing business processes. Even people who have limited knowledge in domain information and communication technology (ICT) can easily model and interpret BPMN. This makes BPMN very appealing to both business users and technical users. Despite its simplicity, BPMN also offers high flexibility, which makes it suitable to model both simple and complex processes.

- Rapid development and integration: There are already many tools and libraries which make it easy to work with BPMN. These tools offer functionalities such as modelling Business Process Diagrams (BPD), executing BPMN logic and visualizing BPMN dynamically. With these tools, a real life process could be turned into BPMN and further integrated into software application logic in a very rapid and simple manner. A more detailed introduction of some of the popular tools for BPMN is presented in the next chapter.

4.2 Available Tools for BPMN

There are many software tools available which help users deal with BPMN. In general, these tools could be classified into three categories: modeller, viewer and BPMN engine.

BPMN Modeller
BPMN modeller is a category of software tools, which help user to draw BPMN diagrams. These tools offer a graphical interface, so that user can use simple operations to add and modify elements in a BPMN diagram. Software such as Microsoft Visio[2] is a typical example of such tools (Figure 2). Besides that, some modellers, such as LucidChart[3], draw.io[4], etc., are even offered as web interface, which further eliminates the need for software installation and configuration (Figure 3).

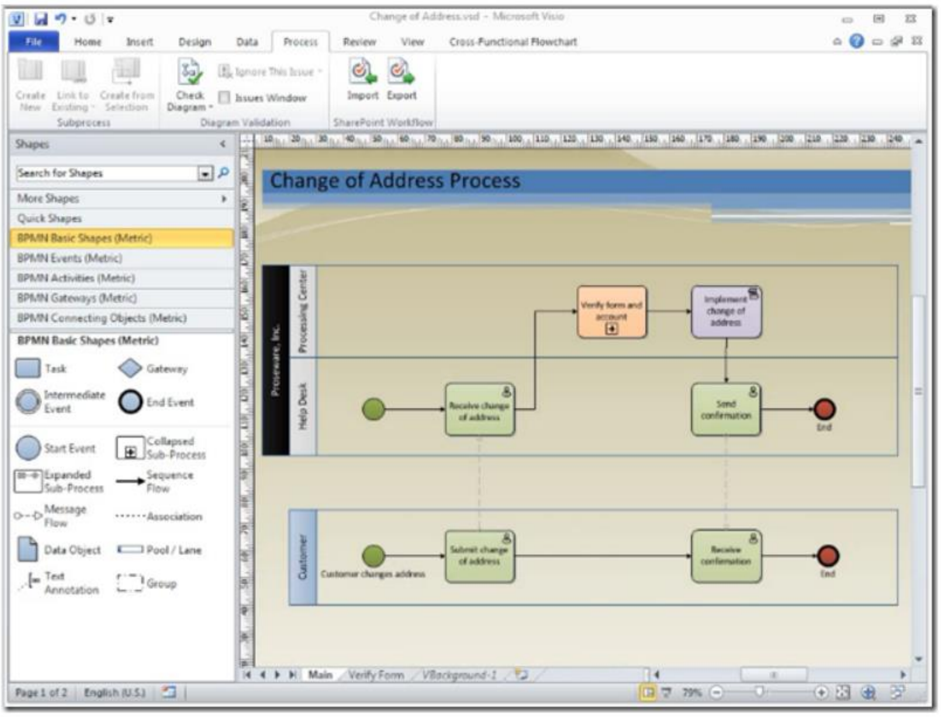
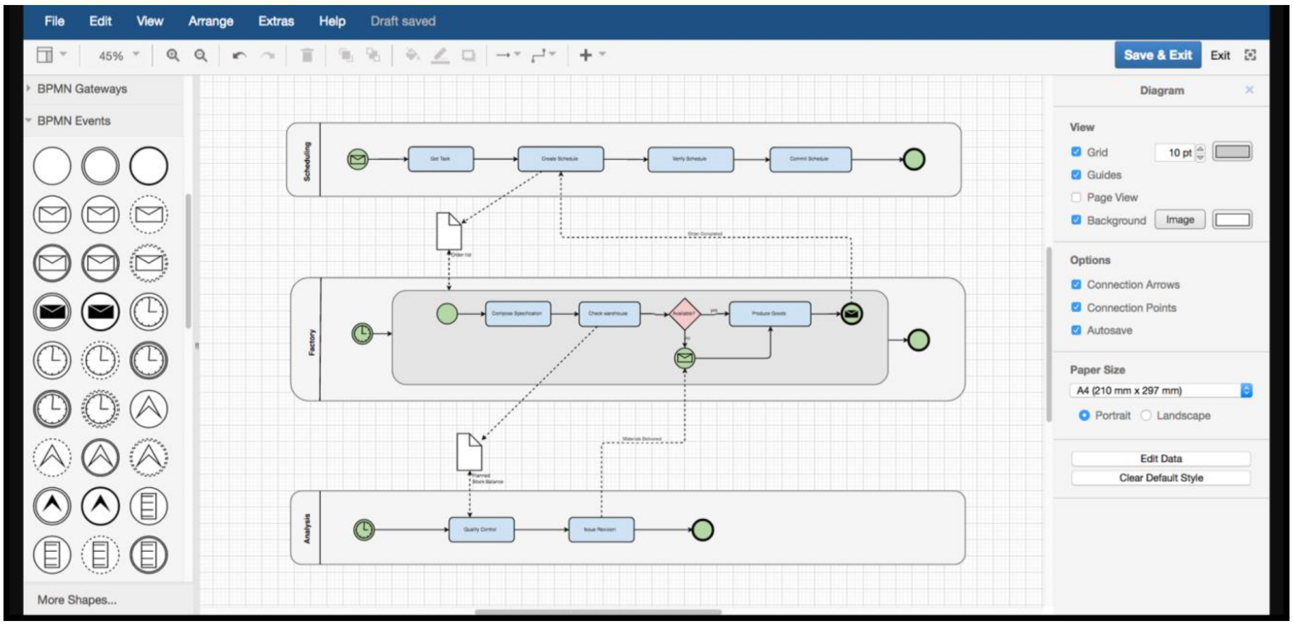


Figure 2 BPMN modelling with Microsoft Visio[5]

[2] https://products.office.com/en/visio/flowchart-software?tab=tabs-1
[3] https://www.lucidchart.com
[4] https://www.draw.io/
[5] https://blogs.msdn.microsoft.com/visio/2011/08/19/bpmn-diagramming-basics-course-available/



View
- [✓] Grid 10 pt
- [✓] Guides
- [ ] Page View
- [✓] Background Image

Options
- [✓] Connection Arrows
- [✓] Connection Points
- [✓] Autosave

Paper Size
A4 (210 mm x 297 mm)
[✓ Portrait] Landscape

Figure 3 Web-based BPMN modeller draw.io

BPMN Viewer

BPMN viewer is a category of software tools, which visualise the underlying BPMN model with a graphical interface. In general, BPMN modellers could also partially fulfil the role as BPMN viewer by presenting the BPMN to users through its GUI. However, BPMN models visualised in this way are static. However, in some use cases, especially when BPMN is used as monitoring interface, dynamic visualization of the process is required. Dynamic visualization of a BPMN model means changing the appearance of the BPMN according to real-time state of the process. Fortunately, a library called bpmn-js[6] offers a good solution for web-based BPMN dynamic visualisation. Bpmn-js is one of the most popular BPMN rendering kit for web-based application. It is written in JavaScript and it could be easily integrated and extended to fit different requirements. It offers API for user to manipulate the appearance of the BPMN model to highlight process status, such as turning the colour of a task to red to indicate error. Besides, it could also be easily extended to build interactive GUI based on the BPMN model that is presented at Figure 4.

[6] https://github.com/bpmn-io/bpmn-js

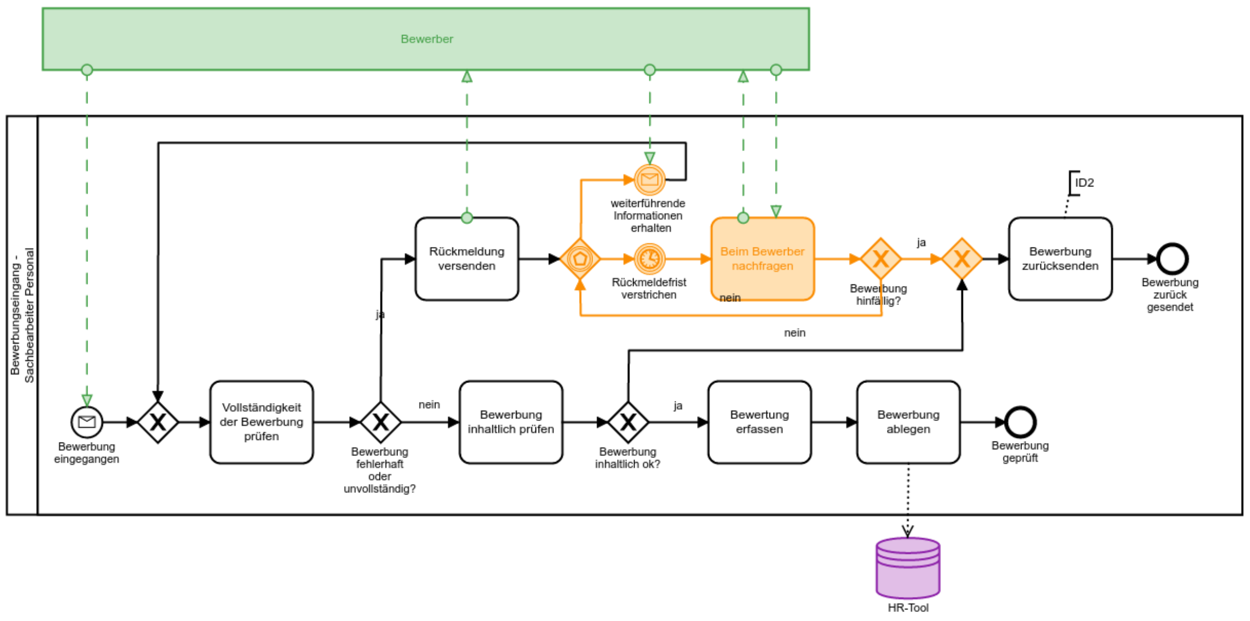

Bewerber
Bewerbungseingang - Sachbearbeiter Personal
Bewerbung eingegangen
Vollständigkeit der Bewerbung prüfen
Bewerbung fehlerhaft oder unvollständig?
ja
nein
Rückmeldung versenden
weiterführende Informationen erhalten
Rückmeldefrist verstrichen
Beim Bewerber nachfragen
nein
Bewerbung hinfällig?
ja
ID2
Bewerbung zurücksenden
Bewerbung zurück gesendet
nein
Bewerbung inhaltlich prüfen
Bewerbung inhaltlich ok?
ja
Bewertung erfassen
Bewerbung ablegen
Bewerbung geprüft
HR-Tool

Figure 4 BPMN model visualized with Bpmn-js with different colors

BPMN Engine

BPMN engine is a category of software tools, which execute the logic behind a BPMN model. When modelling BPMN, developers can specify what actions should be taken, during a certain activity or when the process reaches a certain state. The execution of the actions and the transition to the next state are managed by the BPMN engine. With the help of BPMN engine, users can express reliable service orchestration, human task flows, event handling and much more in diagrams that are technically executable yet easy to understand for everyone.

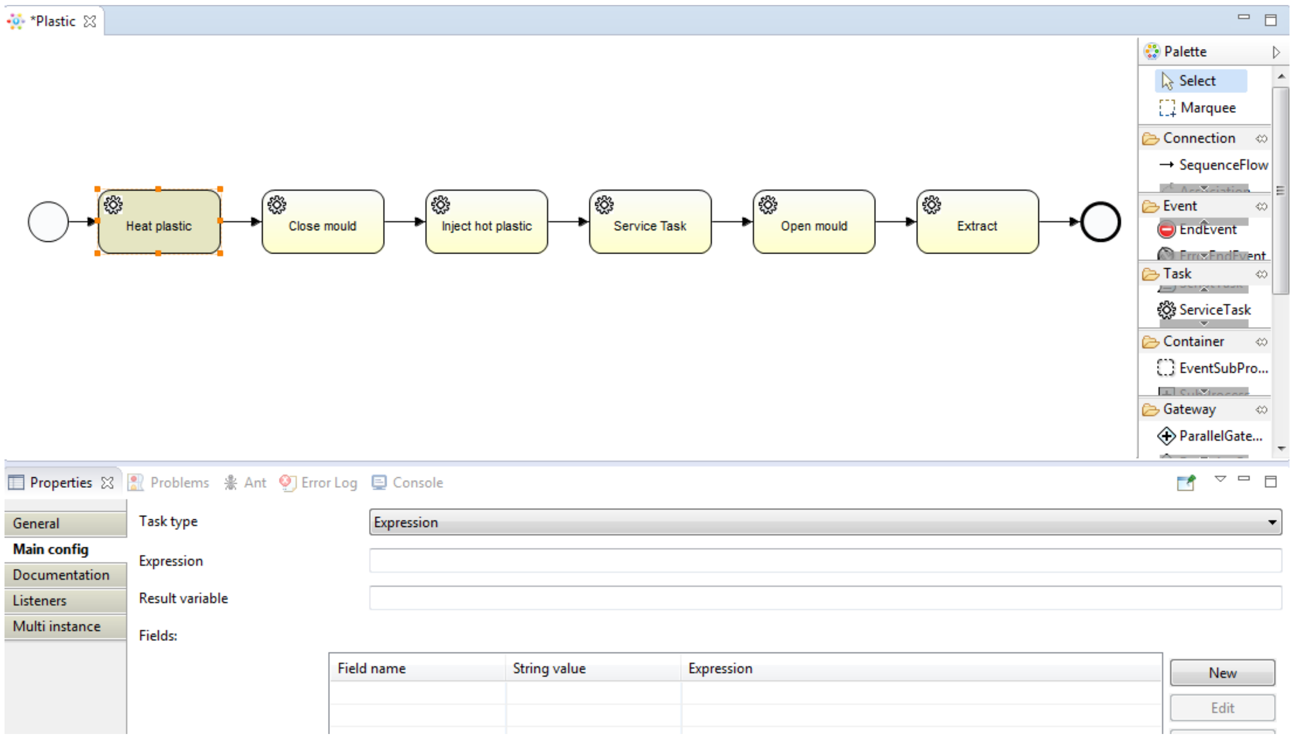
BPMN engines are typically bundled with their own specific BPMN modeller for better compatibility between models and the engine. Typical examples of BPMN engine tools are Activiti[7], Camunda[8], Workflow Engine[9], etc. In the following text, a brief introduction of Activiti and Camunda is given to help readers understand the concept.

Activiti

Activiti is an open source platform for executing and creating BPMN models. Activiti is based on Java, XML, and common database technologies. It consists of several components:

- A web-based interface called Modeller for modelling workflows;
- An Eclipse IDE[10] plugin for developing workflows, including the possibility to add all the data necessary for making it executable in the engine called Activiti Designer;
- An Engine, which is a workflow processor for executing BPMN model logic;
- A web tool for deploying workflows and creating instances called Explorer;
- A collaboration tool for user called Cycle.

[7] https://www.activiti.org/
[8] https://camunda.org/
[9] https://workflowengine.io/
[10] https://www.eclipse.org/ide/



Task type: Expression

Expression:

Result variable:

Figure 5 GUI of Activiti Designer

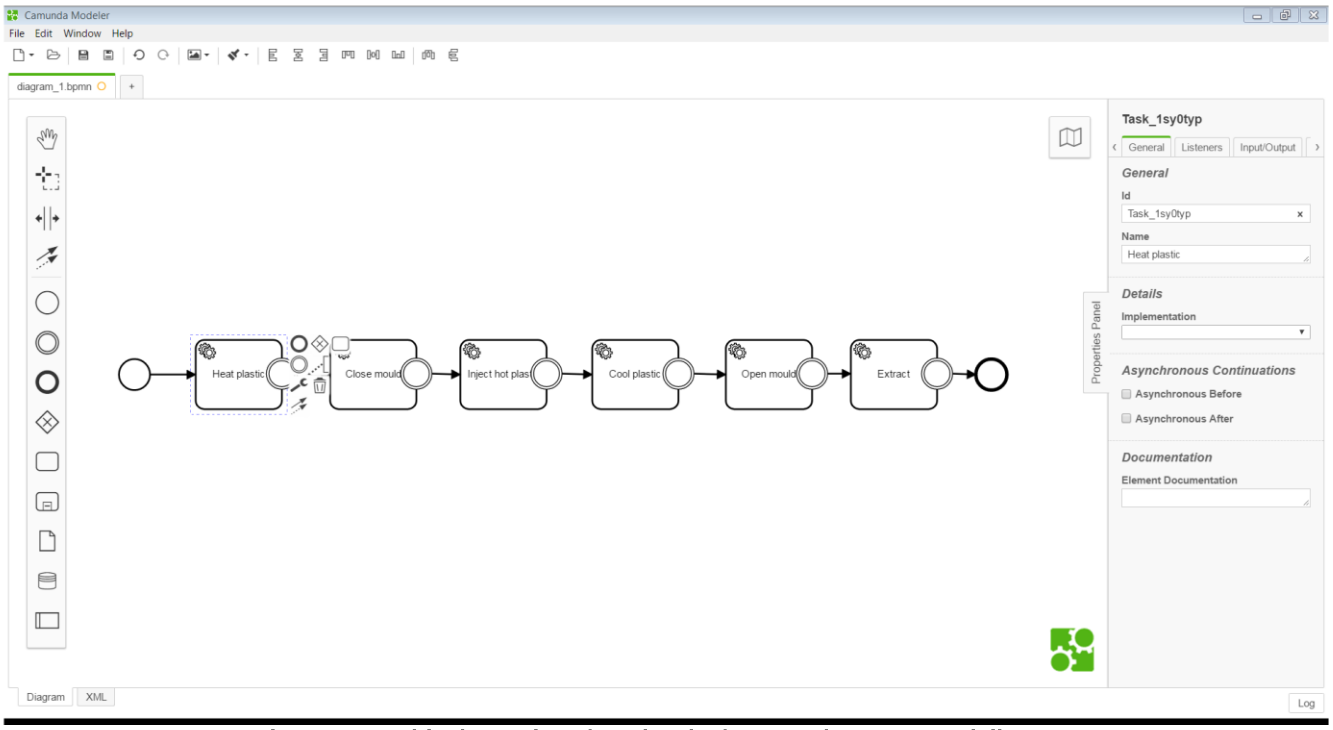
The Activiti Designer has a simple GUI presented at Figure 5. Models are created via drag-and-drop. At the bottom the user can make configurations such as changing the id of an element or adding information needed for the execution of the process model such as the java class to be executed in case of a service task or which event is triggered by which message. If the configuration is filled properly, the Activiti engine is able to execute the process model. This model can be integrated into software program logic for functionalities such as process status tracking, abnormal detection, etc.

Camunda

Camunda is another open source platform for workflow and BPMN management. It was first developed as a fork of Activiti. Since then, many features and performance improvement has been added to expand the usability to a wider range. The Camunda toolset includes the following components:

- Camunda BPMN Modeller – a graphical tool for BPMN diagram modelling;
- Camunda BPMN Workflow Engine – a workflow engine responsible for executing BPMN process logic;
- Camunda Tasklist – a web application that allows end user to work on the tasks assigned to them. It provides additional visibility when using the Camunda Workflow Engine for human task management;
- Camunda Cockpit – a web tool for monitoring workflows and decisions in production to discover, analyze and solve technical problems;
- Admin – a tool for managing Camunda web application or REST API users, organize them in groups and grant permissions;
- Optimise – a tool for creating beautiful reports and arrange them in a dashboard for business monitoring;

The following diagram at Figure 6 shows the GUI of the Camunda BPMN Modeller:



Task_1sy0typ

General

Id: Task_1sy0typ

Name: Heat plastic

Details

Implementation:

Asynchronous Continuations

☐ Asynchronous Before

☐ Asynchronous After

Documentation

Element Documentation:

Figure 6 Graphical User interface (GUI) of Camunda BPMN Modeller

The modeller has a simple, intuitive GUI, which makes it easy to use even for first-time users and those who do not have specialized knowledge in BPMN. Per drag-and-drop user can easily add and modify elements of the BPMN diagram. On the right-side panel, users can also configure settings related to BPMN execution for different tasks and events. These settings include, for example, which java class to run if a specific task is active, or which message would trigger a specific event, etc. By configuring these settings in the modeller, the output BPMN could be directly used in the Camunda BPMN Engine without extra processing.

5 BPMN Monitoring Framework

This section describes the concept of BPMN Monitoring Framework in COMPOSITION. First, the general suitable use cases for the BPMN Monitoring Framework are described to explain what kind of problem the monitoring framework is aiming to solve. After that, the working principle of the Monitoring Framework is further demonstrated with a concrete example.

5.1 Application Scenarios for BPMN Monitoring Framework

This chapter describes how a use case should look like so that the monitoring framework can be used for monitoring the process.

Companies have different reasons why they need to monitor and control their business processes: official regulations, the planning of the production or logistic processes, finding and correcting errors during the activities' execution, using the monitoring data to optimize the process or to increase resource-efficiency. There are commercially available solutions to address most of these reasons. The BPMN Monitoring Framework goes one step further. Its goal is to monitor processes and enrich the process data with relevant data from other sources to gain knowledge beyond simple process or sensor monitoring. Questions like how much energy does a single item need in order to be produced or where do irregularities in the production occur, can be answered with this framework.

Our approach makes it necessary to model the whole process to be monitored. So there needs to be very good knowledge about it and ideally it is documented somewhere. However, it also enables people monitoring the process not only to see the process as a whole, but also to analyse single process steps. Different sensors and actuators are means of interaction in the process and can be integrated in the monitoring framework. One aim of this framework is to be easily understandable and to be configurable by non-engineers. Therefore, we use BPMN to model the process as well as to visualize the process for the monitoring task, as BPMN shows good potential to bridge the understanding of people in different domains.

In order to use the full potential of the framework, the use cases should fulfil certain criteria.

First of all, there should be a real workflow which needs monitoring. This could be a production line or an order process of an online shop. The monitoring of a single machine on the other hand is not a good use case in most cases as there is no real workflow involved.

Secondly, the so-called **signal sensor data** should be available. Signal sensor data are generated by signal sensors, such as an infrared sensor which detects the arrival of a workpiece, or a barcode reader which reads the ID of a workpiece. The data provided by these sensors could help us to synchronize the virtual BPMN process with the real production process, so that at any given time, information such as which workpiece is being processed in which step is known to the virtual BPMN process.

Finally, the so-called **measurement sensor data** should also be available. Measurement sensor data are produced by sensors which constantly measure the state of a single machine on the production line. One typical example is the energy consumption sensor on a specific machine. The measurement sensor data will be enriched and annotated according to the process state determined using signal sensor data. In the case of energy consumption sensor, the measurement data at a specific time point will be annotated with the ID of the workpiece, which is being processed at the same time point. In this way, context is given to the meaningless measurement data and it would be easy to find out how much energy is consumed on producing that particular workpiece.

For a more detailed illustration with concrete example, please see the next chapter.

5.2 Working Principle of BPMN Monitoring Framework

In this chapter, the working principle of the BPMN Monitoring Framework is further elaborated through an example to provide the readers a better understanding.

As mentioned in the previous chapter, the BPMN Monitoring Framework requires a real workflow that needs monitoring. As an example, a simplified subset of the production line in BSL is examined. Figure 7 illustrates the process workflow.

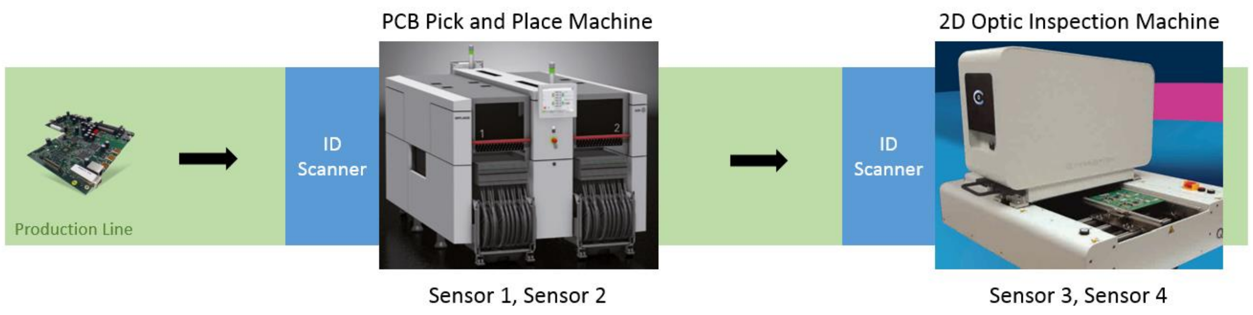


Figure 7 A simplified subset of BSL production line

Figure 7 shows the partial production line in making printed circuit boards (PCB). PCBs are transported along the production line. They will first be processed in a Pick-and-Place machine, in which small electronic components are put on the board. Then the PCB will be inspected in an optic inspection machine to check if error happens during pick and place. Both machines are equipped with ID scanner to read the ID of PCB before processing them. Besides, there are energy consumption sensors installed in each of the machine, which monitor the real-time state of the machine. The goal is to know how much energy is consumed in producing each PCB.

The use case above fulfils the criteria mentioned in the previous chapter. With the process workflow in hand, the BPMN Monitoring Framework can then be utilized to enrich real-time data. The following text describes the simplified workflow for developing a BPMN Monitoring Framework for this use case:

- Design and establish a BPD, modelling the process in question with the help of BPMN modelling tool, such as Activiti or Camunda Modeller.

- Expose the real-time sensor data, including the signal sensor data and the measurement sensor data. Typically, sensor data are published to different IoT channels such as MQTT topics so that the Monitoring Framework can get those data easily.

- Establish a mapping between the signal sensor resources and the events in the BPD, so that each event in the BPMN will be configured to listen to a specific IoT channel. This way, a real life sensor signal can be mapped to trigger a specific event in BPMN, making tracking of products along the process possible. The ID scanners in this example are the signal sensors and they should be mapped to the message event in the BPD (Figure 9).

- Establish a mapping between the measurement sensor resources and the tasks in the BPD, so that the measurement data could be correlated to a specific process task. This way, the measurement data can then be annotated regarding to the actual active products in that task in real-time. In our example, the energy sensors installed on each machine are measurement sensors. The measurement sensor data will be annotated with the currently in process PCB ID (Figure 9).

- Deploy the BPD in BPMN Monitoring Framework. The BPMN Monitoring Framework will take care of managing the process and tracking the products with the help of incoming sensor signals.

Figure 8 shows the BPMN model of the BSL production line subset in the example. The two machines are modelled as two tasks (shown as rectangles) in the model. There are also multiple events modelled (circle with envelop symbol inside). A transition of state will happen, if an event is triggered.

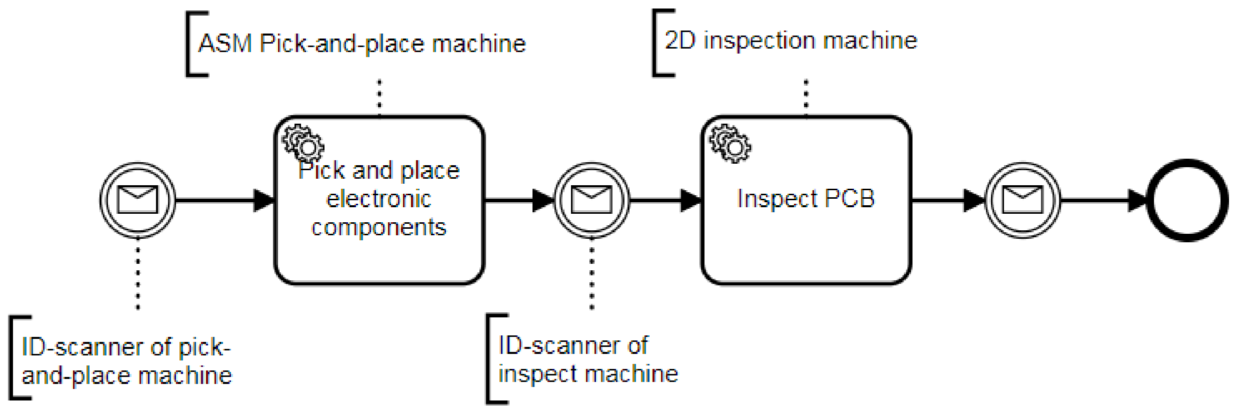
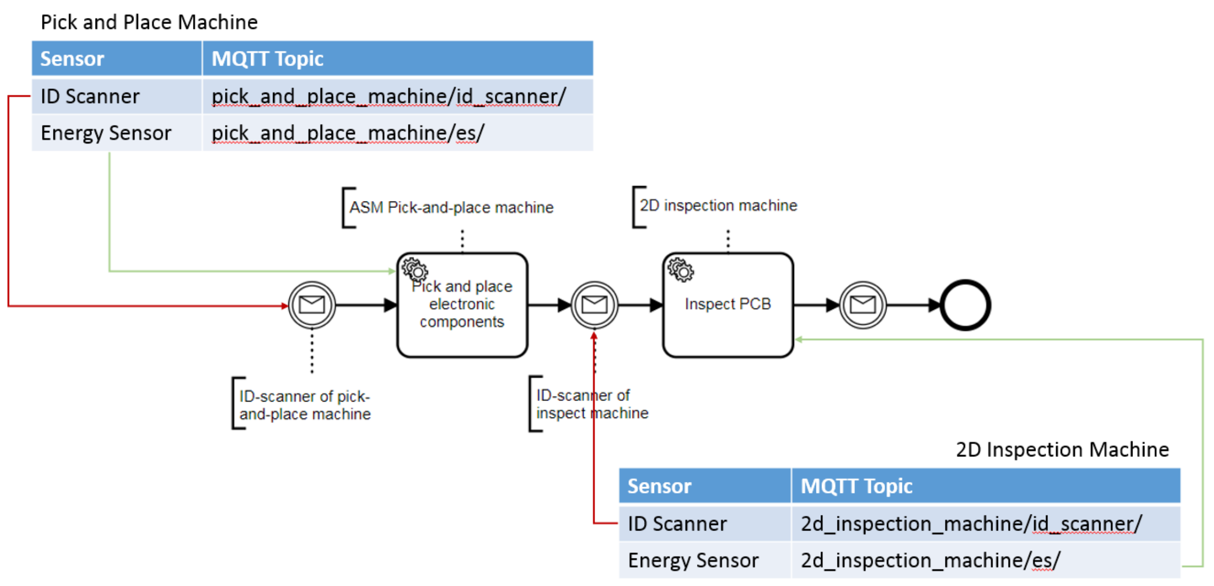

ASM Pick-and-place machine
2D inspection machine
Pick and place electronic components
Inspect PCB
ID-scanner of pick-and-place machine
ID-scanner of inspect machine

Figure 8 BPMN model of the exemplary BSL production line

Figure 9 illustrates the mapping of MQTT topics to the elements in BPD.


ASM Pick-and-place machine
2D inspection machine
Pick and place electronic components
Inspect PCB
ID-scanner of pick-and-place machine
ID-scanner of inspect machine

Pick and Place Machine

Sensor	MQTT Topic
ID Scanner	pick_and_place_machine/id_scanner/
Energy Sensor	pick_and_place_machine/es/

2D Inspection Machine

Sensor	MQTT Topic
ID Scanner	2d_inspection_machine/id_scanner/
Energy Sensor	2d_inspection_machine/es/

Figure 9 Mapping between a Deployment Model and a BPD

6 Integration of BPMN Monitoring Framework with COMPOSITION System

This chapter describes how the BPMN Monitoring Framework is fit into the COMPOSITION system. The BPMN Monitoring Framework is a component of the Integrated Information Management System (IIMS). On one hand, the monitoring framework requires a source for BPMN model data. The Digital Factory Model (DFM), as a central repository for modelling data, provides the perfect container for BPMN models. On the other hand, the monitoring framework also requires access to real-time data, which could be provided by the Decision Support System.

6.1 Integration of BPMN Monitoring Framework with DFM

In this chapter a brief description of Digital Factory Model from Task 3.2 and its connection with the BPMN diagrams and Monitoring Framework are presented.

The Digital Factory Model is a core component of the COMPOSITION system. The DFM enables the digitalization of industrial aspects. Data which are provided from different system's parts in a heterogeneous format finally are described in a common format using DFM schema. This means that all the data are modelled and provided with the same format to all related components. The Digital Factory Model is able to describe all the information related to a factory such as buildings, assets, actors, processes and measurements.

As depicted in the picture below, the DFM structure has three root components:

Factory ID: represents the unique ID of the factory

Information Model: contains all the static information of the factory

Events: contains all the dynamic information about the factory

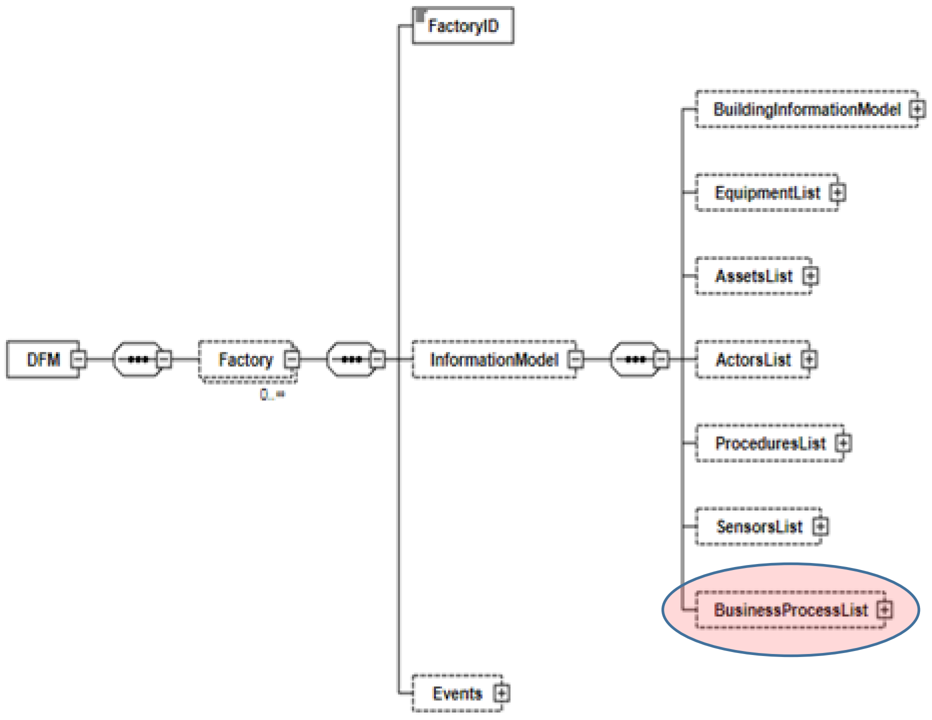


Figure 10: High Level Structure of DFM

The BPMN diagrams from Task 3.1 are part of the Information Model element. More precisely, the Business Process List of the DFM contains all the BPMN diagrams related to factory processes. The BPMN diagrams are designed at the Modeller interface of the BPMN Monitoring Framework (see section 8.4.2). These diagrams will be exported to XML format which is valid with the DFM schema. After that, they will be imported to the Business Process List of the DFM instances. The Business Process List element of the DFM was covered by OMG's BPMN XML package (BPMN, 2018). This package provides schemas which offer all the necessary means for the representation of a BPMN diagram in a DFM instance.

The next figure presents a high level view of the structure of DFM's Business Process List based on BPMN schema:

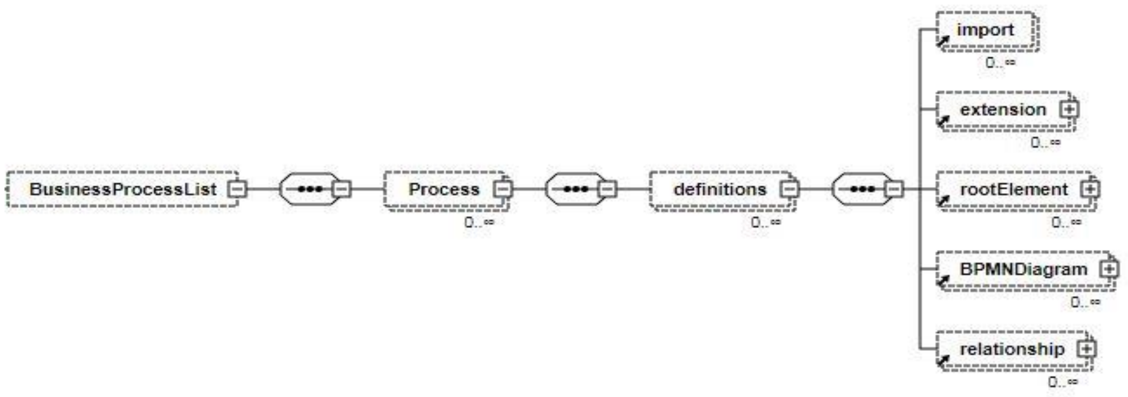


Figure 11: Business Process List XML Schema Representation

The BPMN diagrams will be available to the rest of IIMS components via DFM API. By using the DFM API and its supported interfaces/services, the IIMS components are able to store, retrieve or delete data from a common database. The data exchange format should be valid against the DFM schema. The Monitoring Framework will be able to load BPMNs using the DFM API's services which are related to Business Process List. Furthermore, the BPMN information will be available to DSS component using the DFM API services as well. Based on D3.2 (COMPOSITION D3.2, 2017), the supporting interfaces of DFM API which are related to BPMNs are the following:

- String *setBusinessProcessList* (String bpmnXML)

  It sends an XML file compatible with the DFM's schema structure (actually BPMNL structure) for storing to a back end database. The DFM API reads the factory ID from the posted XML file, and the total information about the factory's business processes (BPMN diagrams) is stored or updated (if exists) in the corresponding DFM instance. A message is returned. This message informs the component which sends the request if the operation was successful or not.

  It is called by a corresponding POST request. The XML file which contains the complete list of the business processes diagrams of the factory is the body of the request.

- String *setBusinessProcess* (String bpmnXML)

  It sends an XML file compatible with the DFM's schema structure (actually BPMN structure) for storing to the back end database. The DFM API reads the factory ID from the posted XML file, and the business process information (BPMN diagram) which is described in the XML file is stored or updated (if exists) in the corresponding DFM instance. A message is returned. This message informs the component which sends the request if the operation was successful or not.

  It is called by a corresponding POST request. The XML file which contains the information of a factory's BPMN diagram is the body of the request.

- String *getBusinessProcessList*(String factoryID, boolean zip)

  It retrieves from the database all the BPMN diagram's information stored in a DFM instance based on the factory ID. If the value of the boolean parameter zip is true, then the response will return a zipped file which contains the XML file with the complete list of the factory's business processes. Else, the returned information will be an XML file in text form.

  It is called by a corresponding GET request. The factoryID and zip are URL parameters.

- String *getBusinessProcessByID* (String factoryID, String bpmnID, boolean zip)

  It retrieves from the database the information of a specific business process diagram related to a DFM instance, based on its ID and the factory ID. If the value of the boolean parameter zip is true, then the response will return a zipped file which contains the XML file with the corresponding BPMN diagram's information. Else, the returned information will be an XML file in text form.

It is called by a corresponding GET request. The factoryID, bpmnID and zip are URL parameters.

- String *deleteBusinessProcessList* (String factoryID)

  It deletes based on the factory ID, the complete list of the business processes of a factory which are stored in a DFM instance. A text message is returned. This message informs the component which sends the request if the operation was successful or not.

  It is called by a corresponding GET request. The factoryID is URL parameter.

- String *deleteBusinessProcess*(String factoryID, String bpmnID)

  It deletes the information from the database related to a specific business process diagram, based on its own ID and the corresponding factory ID. A message is returned. This message informs the component which sends the request if the operation was successful or not.

  It is called by a corresponding GET request. The factoryID and bpmnID are URL parameters.

Besides that, a wide catalogue of supported interfaces is provided by DFM. The Monitoring Framework can use these interfaces in order to get information about the actors, assets and equipment which are related with the BPMN diagrams. A complete list of the supporting interfaces is presented at D3.2 - Digital Factory Model I.

6.2 Integration with DSS

The DSS (Decision Support System) is another component in the IIMS. The main aim of the DSS will be to make a step forward towards a better understanding of the involved manufacturing processes and operations, the contribution of individual links of the supply chain, the effect of process monitoring in productivity, to facilitate communication and knowledge sharing among departments with different roles and responsibilities, the maintenance requirements and procedures and the detection of daily production details and flaws. The DSS will receive data provided from the partners involved in the ongoing processes and they will be processed in real time. They will be coupled with the associated requests to certain parts of the supply chain, SOP (standard operating procedures) and response strategies, in order to offer feedback to the involved internal or external suppliers, in terms of actionable knowledge and recommendations, including maintenance operations and schedules.

The DSS receives data coming from the production line and does the necessary processing on those data. The data are then exposed to the BPMN Monitoring Framework through web services. This provides a source of real-time data for the monitoring framework. More details about the DSS will be documented in deliverable 3.8 – Manufacturing Decision Support System I in M20.

7 Manufacture Process Modelling

As mentioned in the task description, T3.1 includes the modelling of manufacture processes with BPMN. The PCBA production line from BSL has been chosen as the prototype for modelling. The manufacture process starts by printing solder paste on unprocessed PCBs and then proceeds to place electronic components on board. Then the PCBs are put into a reflow oven, where the solder paste hardens and the placed components are stabilized. Eventually the PCBs are cooled down and examined for later production usage. Figure 12 shows the complete BPMN model of the PCBA production line.

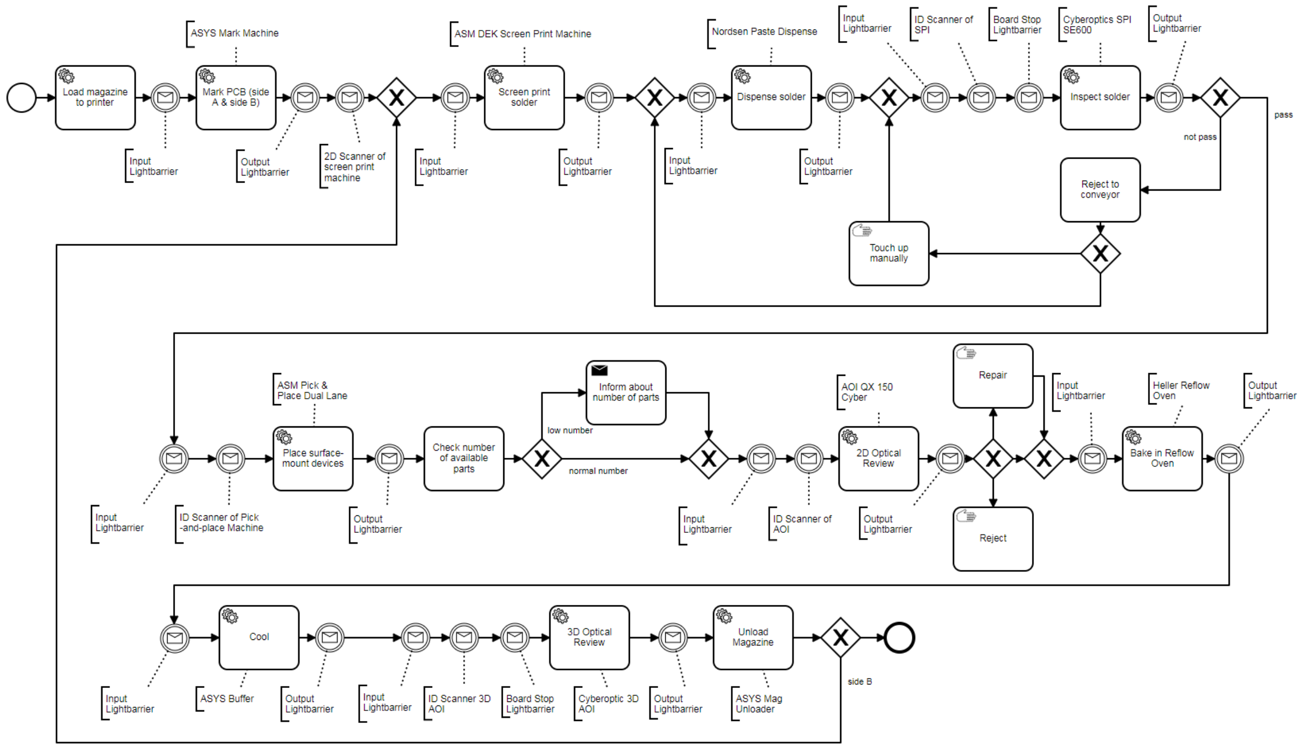


Figure 12 BPMN model of the PCBA production line in BSL

The PCBs are processed in the following sequential steps:

1. The magazines full of PCBs will be loaded to ASYS mark machine, where the PCBs will be marked with unique ID with laser.
2. The PCBs will be brought to a solder screen printing machine (DEK Neo-Horizon from ASM), where solder paste will be printed on board with stencil.
3. The PCBs goes through a solder paste dispenser machine (Paste Dispenser from Nordsen) in pass through mode, which means that if the screen printing has no problem, then the board goes through directly. If the solder paste is undervolume (that means not enough paste has been applied on the board), the dispenser will put additional paste to the board. If it is overvolume (that means too much paste has been applied on the board), the board will have to be taken out and manually touched up.
4. The PCBs will be transferred to a pick-and-place machine (Pick & Place Dual Lane from ASM), where small electronic components are placed onto the board.
5. The board with placed components will then go through a 2D optical reviewing machine (AOI QX 150 Cyber), where the correctness of the placement will be examined. If any problem occurs, the board may need manual repair, or may be even removed completely out of the production line.
6. PCBs are transported to a reflow oven, solder melts and then hardens, hence stabilizing the placed electronic components.
7. PCBs are cooled down in a buffering machine (Buffer machine from ASYS).

8. PCBs are inspected in a 3D optical reviewing machine to check if everything's OK.
9. The PCBs are unloaded from the magazine. If required, repeat the previous steps for side B of the PCBs.

In order to make the model compatible with the COMPOSITION BPMN Monitoring Framework, the following principles are followed during model time:

1. Each activity involving equipment on the production line should be modelled as a service task in the BPMN model (represented as a rectangle with gear symbol on the top left corner). This way, the monitoring framework could tell that this is where you want the machine status to be displayed.
2. Each of the service tasks should be assigned a unique ID, which the monitoring framework could use to map data getting from external components to the service task.
3. Signal sensors are modelled as message events (circles with envelop symbol inside). The signal sensor data should be mapped to those message events.
4. Other activities should be modelled with other types of tasks, such as manual task (rectangle with a hand symbol on top left corner), send task (rectangle with an envelope symbol), etc. according to actual situation.

8 Implementation of BPMN Monitoring Framework in COMPOSITION

This chapter describe the implementation of the BPMN Monitoring Framework in details. The technical requirements derived from WP2 as well as from partner components are extracted and formulated in the first part of this chapter. Then suitable tools and technologies for fulfilling the requirements are chosen and introduced in the second part. In the following part, the details in implementation including software architecture, exposed interfaces and deployment are presented. Finally, the extensibility and reusability of this framework is also discussed in the last part of this chapter.

8.1 Requirements for the BPMN Monitoring Framework

The BPMN Monitoring Framework will be implemented mainly for the use case BSL-5, Equipment Monitoring and Line Visualisation. The main goal of this use case is to keep track of equipment issues, such as the amount of actually processed units compared to the target amount and the up and down state of each machines. Users can be informed instantly on changes in equipment status so that they can take quicker actions to adjust production.

Compared to the example use case described in section 5.2, there is no workpiece tracking as well as no data annotation requirements are involved in this use case. As a result, the BPMN Monitoring Framework is mainly used as a visualization tool. The whole production line will be visualized as a BPMN diagram and real-time state of each machine will be reflected dynamically on this diagram. Compared to a regular real-time dashboard, a real-time line visualization can provide the users not only the current information, but also the whole process as a context for digesting such information. Users can have a more intuitive feedback regarding where the bottleneck occurs and how it could affect the whole production line.

From an implementation point of view, there are the following requirements for the BPMN Monitoring Framework:

- **Connection with DFM:** The BPMN Monitoring Framework must be able to get BPMN model from the DFM. The DFM is essentially a repository, in which all relevant model data are stored. These also include the BPMN process model that the user wants to monitor. As a result, it is necessary to establish a connection between the BPMN Monitoring Framework and the DFM, so that the BPMN model in question could be retrieved from the DFM to the monitoring tool.
- **Connection with DSS:** The BPMN Monitoring Framework must be able to get real-time data from the DSS. The DSS is responsible for providing real-time data indicating the state of each machine. Since these data have to be shown in the monitoring tool, it is also necessary to establish a connection between the monitoring tool and the DSS, so that real-time data could flow in.
- **Dynamic line visualization:** The BPMN Monitoring Framework must be able to visualize the whole production line as a BPMN diagram to provide user context of the monitored process. Besides, dynamically display real-time information on top of it to provide user intuitive feedback of the production line's current state.
- **Interface exposure for third party tools:** In some use cases, other programs may want to retrieve data from the BPMN Monitoring Framework for their own usage. In order to fulfil such use cases, the BPMN Monitoring Framework should also expose other interface besides the web interface to allow interaction with other programs.
- **Conformation of communication and security standard:** As part of the COMPOSITION system, the BPMN Monitoring Framework should conform to the communication as well as security standard used within the system to ensure interoperability and security.

8.2 Tools and Technologies

The tools and technologies used for the BPMN Monitoring Framework development are chosen according to the following criteria:

- Tools and technologies should address the requirements described above;
- When possible, open and free technologies and tools are preferred;

- Tools and technologies with well-established standards are preferred to ensure interoperability and extensibility.

The main selected technologies and tools are the following:

Web Services as defined by World Wide Web Consortium is a system designed to support interoperable Machine to Machine interaction over a network. Web services are server applications which can process and exchange data. As the components in COMPOSITION are developed as distributed services, it was decided that the BPMN Monitoring Framework should also follow the same principle, which means it is a standalone service and it communicates with other components through web service interface.

REST or Representational State Transfer was selected as the architectural style of web services. REST offers good performance, modifiability and scalability to enable web services to work better on the Web. The REST architecture style is a client/server architecture where clients and servers exchange representations of resources by using a standardized interface and protocol. Resources are accessed using Uniform Resource Identifiers (URIs) which are the typical links on the Web.

HTTP stands for HyperText Transfer Protocol. It is the selected protocol to be used by the RESTful API. This application protocol is used to link pages of hypertext and it is a way to transfer files. HTTP is the foundation of data communication for the Web. HTTP protocol is a supported protocol by COMPOSITION system architecture.

Java was selected as the implementation language for the back end component. It is a general purpose, object oriented programming language. Java is one of the most popular programming languages in use, especially for client server web applications.

8.3 Implementation Details of the BPMN Monitoring Framework

Starting from this standpoint, the BPMN Monitoring Framework in COMPOSITION is designed as following:

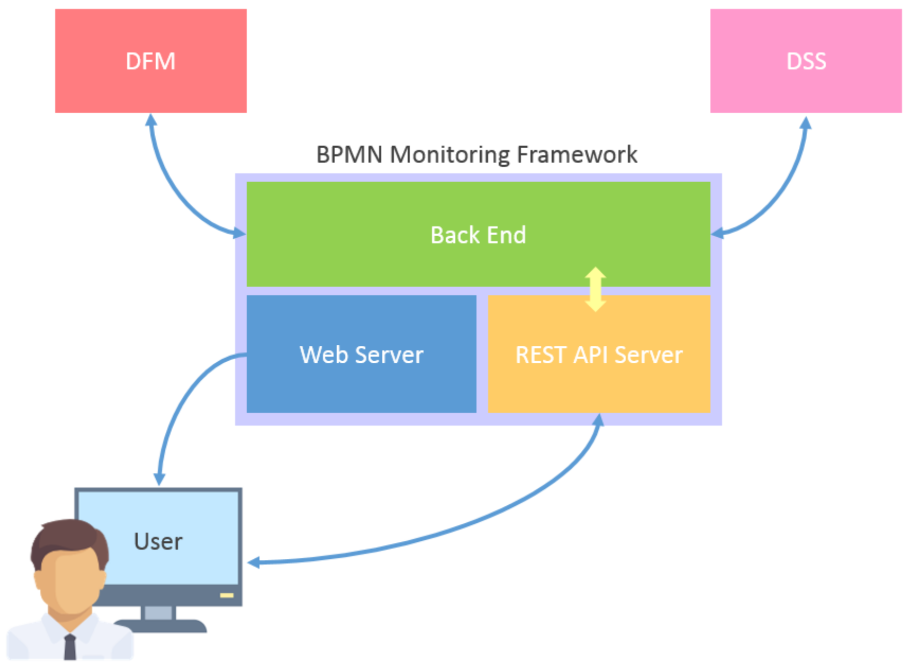


Figure 13 Structure of the BPMN Monitoring Framework and its interaction with other components

Figure 13 shows the structure of the BPMN Monitoring Framework as well as its interaction with other components in COMPOSITION. The BPMN Monitoring Framework consists of three internal components:

Web Server

The Web Server has the responsibility of serving web pages and resources to the users. The web pages serve as the GUI for both the interaction with the monitoring framework and the visualization of the process. Notice that a web interface has been chosen as the visualization solution for the following reasons:

- Mature JavaScript library (Bpmn-js) is available for dynamic display of BPMN diagram. With the help of Bpmn-js, development speed could be largely accelerated, as there is no need to implement the underlying basic functionalities.
- The web interface could be easily supported across all platforms, as long as the machine has a modern browser installed. No more installation is required to use the monitoring tool.
- The web interface is easily extensible, as it utilizes commonly used web technology. New developers can get their hands on quickly to start customizing the tool to fit their use case.

The web interface will be displayed in the browser on the users' computer. By utilizing AJAX (Asynchronous JavaScript And XML) technique, the web interface could interact with the REST API Server to retrieve data from the BPMN Monitoring Framework.

REST API Server:

The REST API Server expose a REST interface for both interaction with web interface as well as third party programs. REST-compliant Web services allow requesting systems to access and manipulate textual representations of Web resources using a uniform and predefined set of stateless operations. The REST interface has been chosen, as it is commonly used in IoT domain to ensure application interoperability and connectivity. A more detailed description of the interface provide by this server can be found in the next chapter.

Back End:

The Back End is responsible for communicating with other external software components and providing the retrieved data for the REST API Server. As mentioned in the previous chapter, the Back End on one hand connects to the DFM to retrieve BPMN model from it, on the other hand connects to the DSS to retrieve real-time data. All the data retrieved will be prepared for consumption through REST API Server.

8.4 Interface of the BPMN Monitoring Framework

8.4.1 REST Interface

In this chapter, the REST API available on REST API Server is documented in details.

Table 2 details of REST API "Get local BPMN model"

Get BPMN model uploaded to the monitoring framework	
URL	/rest/bpmn/
Method	GET
Request data content	None
Response data content	XML file of the BPMN model uploaded to the monitoring framework
Comments	This API returns the BPMN model which is currently being used in the BPMN Monitoring Framework.

Table 3 details of REST API "Upload BPMN model to BPMN Monitoring Framework"

Upload BPMN model to BPMN Monitoring Framework	
URL	/rest/bpmn/
Method	POST
Request data content	The XML file of the to be uploaded BPMN model
Response data content	none

Comments	Through this API, user can upload their BPMN model to the BPMN Monitoring Framework. By doing so, the newly uploaded model will be used to display the process.

Table 4 details of REST API "Get BPMN model list from DFM"

Get BPMN model list from DFM	
URL	/rest/bpmn/remote/
Method	GET
Request data content	None
Response data content	List of BPMN models available in DFM
Comments	By calling this API, user will get the list of available BPMN models stored in DFM. For this API, the backend of BPMN Monitoring Framework functions as a proxy to fetch the model list from DFM and then return to the users.

Table 5 details of REST API "Get BPMN model from DFM"

Get BPMN model from DFM	
URL	/rest/bpmn/remote/{id}
Method	GET
Request data content	None
Response data content	XML file of the BPMN model in DFM, specified by {id}
Comments	By calling this API, the XML file of the BPMN model in DFM will be returned. The model to return is specified by the parameter {id} in the URL. For this API, the backend of BPMN Monitoring Framework functions as a proxy to communicate with DFM.

Table 6 details of REST API "Upload BPMN model to DFM"

Upload BPMN model to DFM	
URL	/rest/bpmn/remote/{id}
Method	POST
Request data content	XML file of the BPMN model, which is to be uploaded to DFM
Response data content	None
Comments	By calling this API, users can upload the list of available BPMN models stored in DFM. The backend of BPMN Monitoring Framework functions as a proxy to post the model to DFM.

8.4.2 Web Interface

The BPMN Monitoring Framework provides two web interfaces to the user: the Modeller and the Viewer interface. The Modeller offers a simplistic and easy-to-use graphical interface for users to build new BPMN models or to tweak the existing ones for better visualization in the Viewer interface. Within this interface, user can add elements per drag-and-drop to the models and wire them up in an intuitive fashion. The Viewer interface is the place where users can monitor the current state of the whole production line. The modelled BPD will be shown in this interface, with all different state variables dynamically represented on the diagram.

In the following part, detailed description of both interfaces' usage will be presented. Please notice that as the HMI designing task in COMPOSITION is still ongoing and BSL-5 is not the prioritized use case, the COMPOSITION interface design hasn't been applied to the interfaces yet. As a result, the following screenshots reflect only the temporary look, which will be adapted to fit in COMPOSITION's interface styling guidelines later. However, the description below remains valid even after the adaptation, as only the styling would be changed and the functionalities of the interface will remain the same.

Modeller

The user can visit the Modeller interface under the URL http://<host IP address>/modeller/. Upon visiting, the user will be brought to the main menu of the Modeller, in which he or she needs to choose the source of the BPMN model to edit (Figure 14). There are the following options:

- **Uploaded Model:** upon choosing this option, the currently in the framework uploaded BPMN model will be opened for editing.
- **DFM:** upon choosing this option, a list of available BPMN models stored in DFM will be presented to the user. By choosing any one of those, the selected model will be retrieved from DFM and opened for editing.
- **Hard Drive:** upon choosing this option, the user will be prompted to select a local BPMN model file from the hard drive. The selected file will be opened for editing.
- **Create New Diagram:** upon choosing this option, the user will be brought to the editor interface with an empty canvas.

After selecting the source of BPMN model, the user will be brought to the editor interface with the chosen model displayed on the canvas (Figure 15). The editor consists of multiple components:

- **Toolbox panel:** the toolbox panel contains different elements, which users can per drag-and-drop add to the current BPMN model.
- **Property panel:** the property panel reflexes the properties of the chosen element. By clicking on any element in the diagram, the property panel will show what properties are available for the selected element.
- **Control panel:** in the control panel, users can execute different operations, such as save model to your hard drive as XML file, save as SVG file, upload the model to BPMN Monitoring Framework and upload the model directly to DFM.

It is mandatory for the user to upload a BPMN model to the monitoring framework using the Modeller interface, so that the Viewer interface knows what models to visualize.

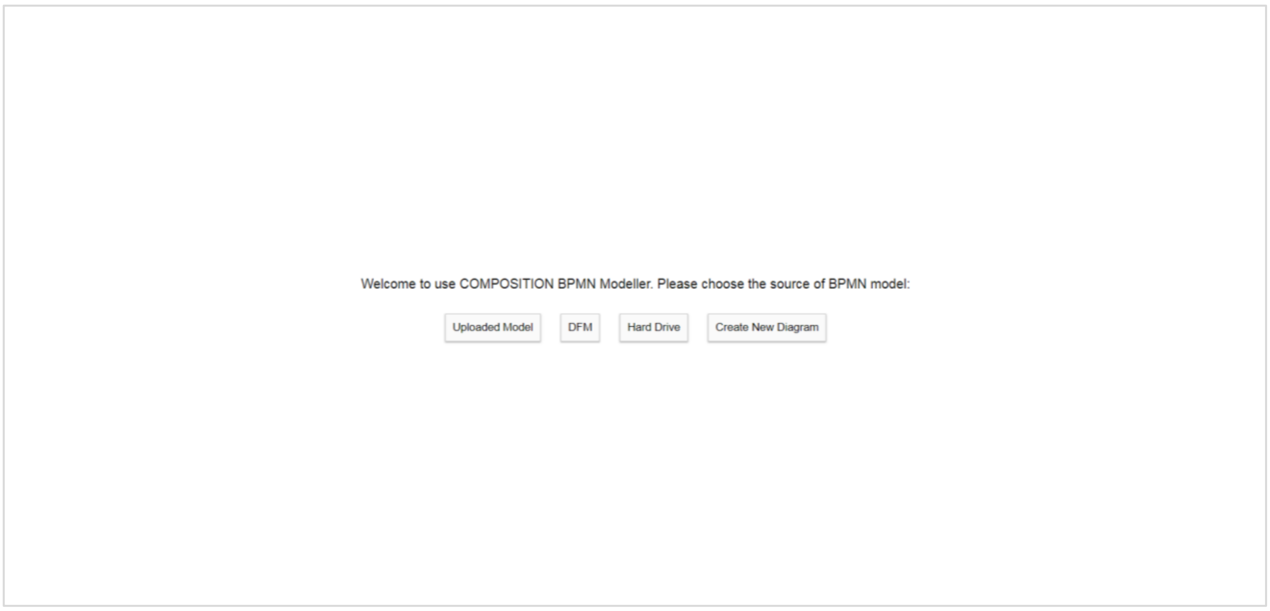
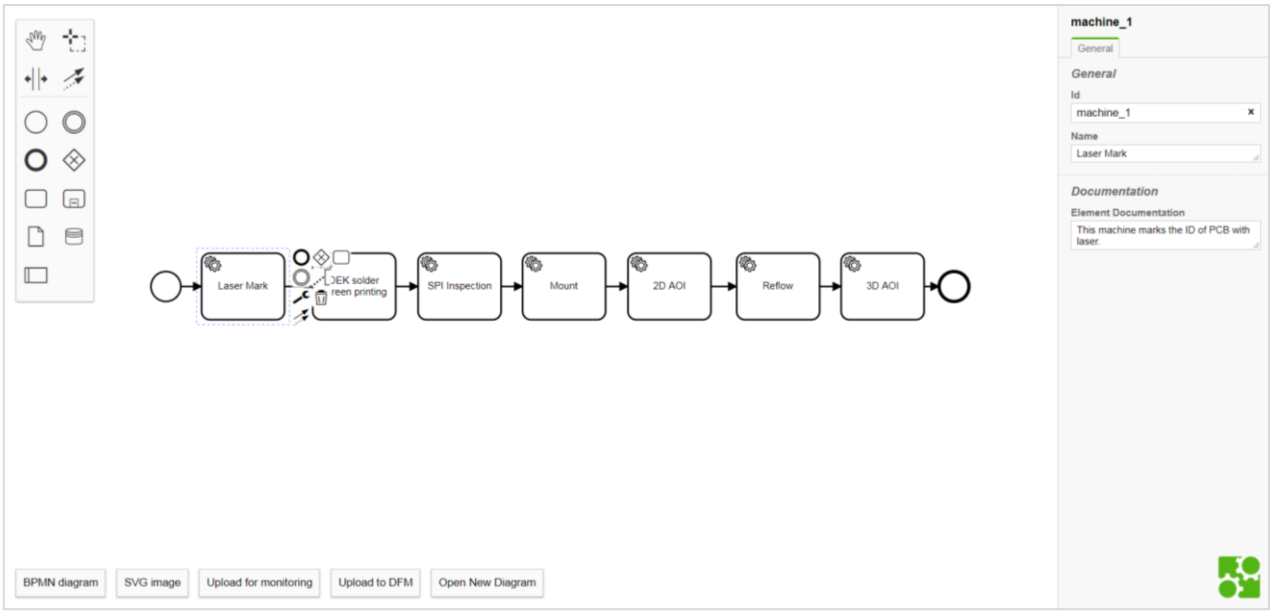

Welcome to use COMPOSITION BPMN Modeller. Please choose the source of BPMN model:

Figure 14 the Modeller main menu interface



machine_1

General

General

Id

machine_1

Name

Laser Mark

Documentation

Element Documentation

This machine marks the ID of PCB with laser.

Figure 15 the Modeller editor interface

As mentioned in section 7, each machine to be monitored should be modelled as a service task element in the diagram. A service task element (represented as rectangle with gears in the diagram) has three properties: ID, name and documentation. The most important property is the ID, which serves as a unique identifier to locate the data resources in the DSS. The BPMN Monitoring Framework will look for data resources from the DSS according to the ID specified here and map it to the correspondent rectangle. Therefore, users need to map the ID correctly so that the data could be shown in the desired location.

The property “name” represents the text shown within the rectangle and serves only for visualization purpose. The property “Documentation” only serves as a holder for comments on the selected element(add button to upload to DFM).

Viewer

The Viewer interface can be visited under the URL http://<host IP address>/viewer/. Notice that the Viewer interface could only be shown, when a BPMN model has already been uploaded to the framework. In the absence of an uploaded model, the user will be brought to another page, which urges the user to use the Modeller to upload a model first.

If a model has been successfully uploaded, the user will be brought to the Viewer interface (Figure 16). This interface visualizes the whole BPMN models. At the same time, the current state of the production line will be shown on top of the diagram. Notice that according to the requirement from BSL, the visualized model is simplified to only reflect the seven machines in the production line.

On the bottom-left corner is the control panel. It contains multiple buttons, with which users can execute different operations. The “Setting” buttons will open a setting side panel, in which users can configure the interface’s visual representation, data polling interval, etc. The “Stop/Start monitoring” button can toggle the continuous polling of data from the backend.

The current state of each machine is displayed on the matching rectangle in the BPD. The colour of the rectangle represent whether a machine is running or stopped. Some important indicators are shown both above and below the rectangle. When user clicks on one of these rectangles, a detail side panel will be brought up, which contains more detailed information about the selected machines.

For better usability, the interface also allows user to zoom in and zoom out per mouse scrolling. User can adjust the diagram to the perfect size for visualization.

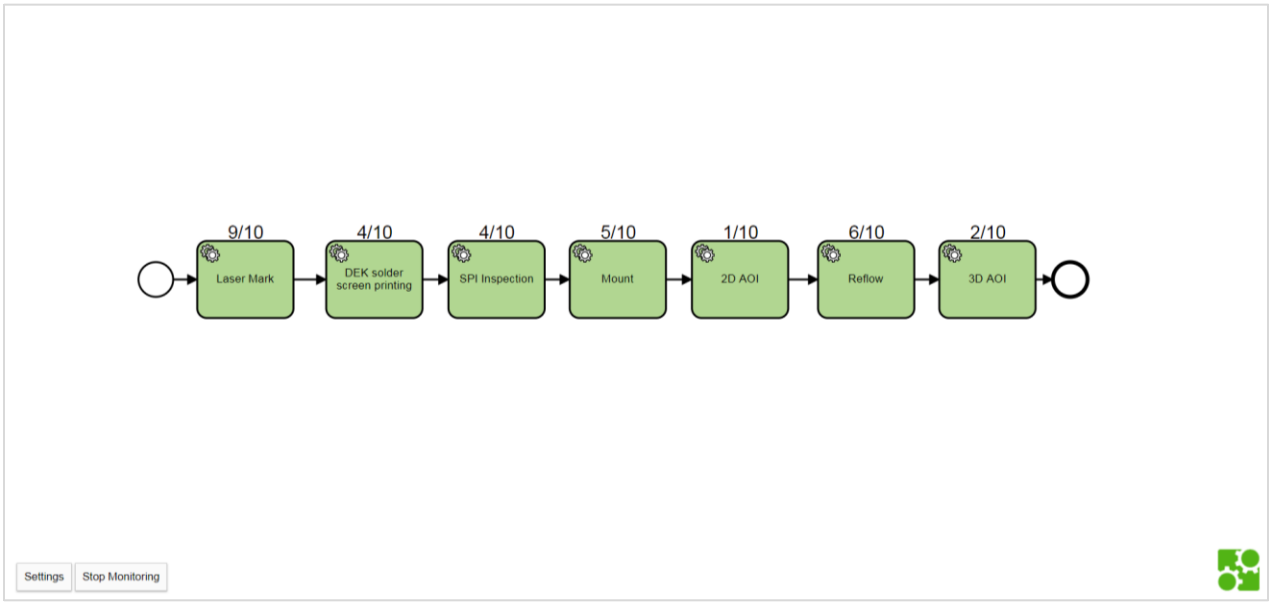
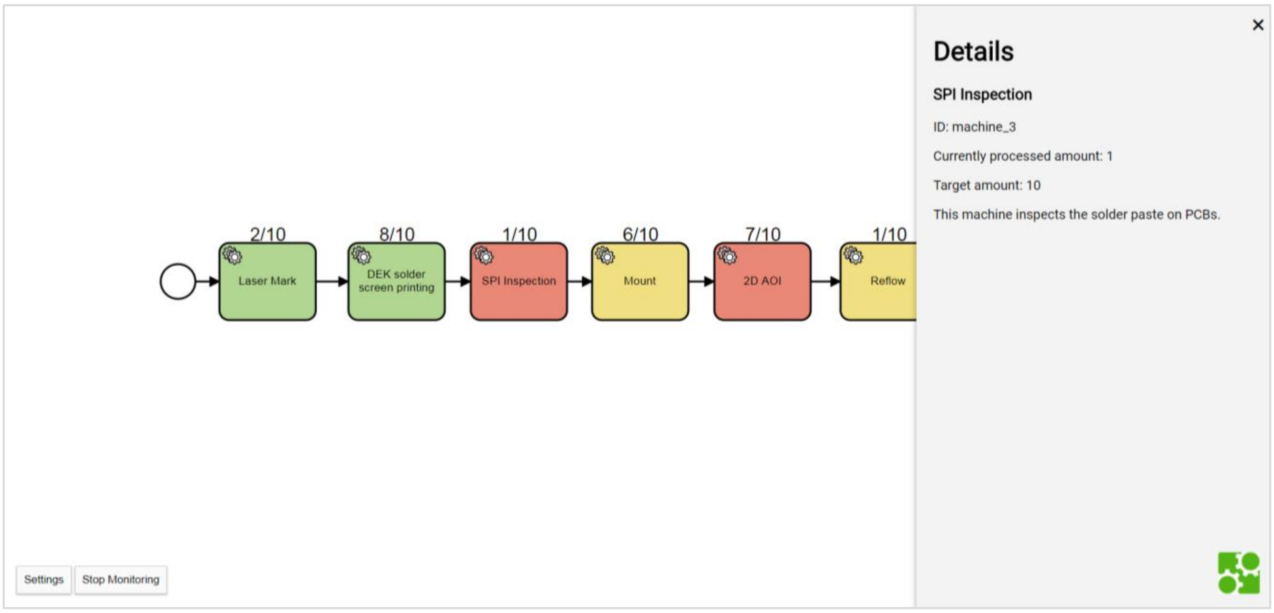


Figure 16 The Viewer interface



Details

SPI Inspection

ID: machine_3

Currently processed amount: 1

Target amount: 10

This machine inspects the solder paste on PCBs.

Figure 17 The Viewer interface with detail side panel

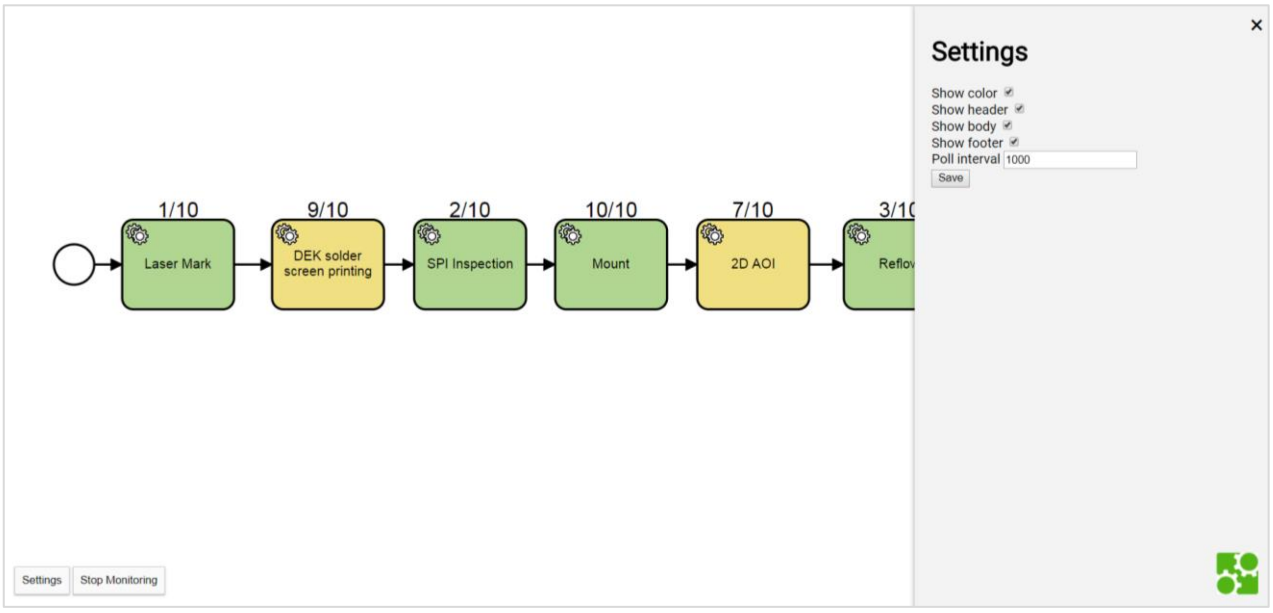


Figure 18 The Viewer interface with setting side panel

8.5 Deployment

COMPOSITION utilizes Docker widely for deployment. Docker is a software technology providing operating-system-level virtualization also known as containers. A container image is a lightweight, stand-alone, executable package of a piece of software that includes everything needed to run it: code, runtime, system tools, system libraries, settings. Containerized software will always run the same, regardless of the environment. Containers isolate software from its surroundings, helping to reduce conflicts between teams running different software on the same infrastructure.

Because of the above mentioned features, Docker containers are widely used for clean and easy software deployment. Following the COMPOSITION guidelines, the BPMN Monitoring Framework will be shipped with a Docker image for deployment.

8.6 Extensibility and Reusability

Although the BPMN Monitoring Framework is mainly developed for the specific BSL-5 use case, extensibility and reusability are never compromised during the development. In fact, the BPMN Monitoring Framework could be easily extended and reused for different use cases, as long as they fulfil certain basic criteria.

As shown in BSL-5, the BPMN Monitoring Framework relies on two external components: a source of model data (the DFM in BSL-5) and a source of real-time data (the DSS in BSL-5). The existence of a real-time data source is mandatory, as it provide data representing the current state of the production line that are interesting for the user to monitor. The source of model data, however, is not a mandatory requirement, as the BPMN Monitoring Framework also allows user to manually create or upload BPMN model with its Modeller interface.

As long as the use case fulfils the above requirements, the BPMN Monitoring Framework could be extended easily to fit in. In fact, to reuse the monitoring framework for other use cases, the only thing the developers have to do is to adjust the back end component so that it could talk to the new external data source component. Thanks to the modularity design inside the back end component (Figure 19), only the Real-time data connector (or optionally also the Model data connector) needs to be customized, minimising the efforts required for adapting the monitoring framework for new use cases.

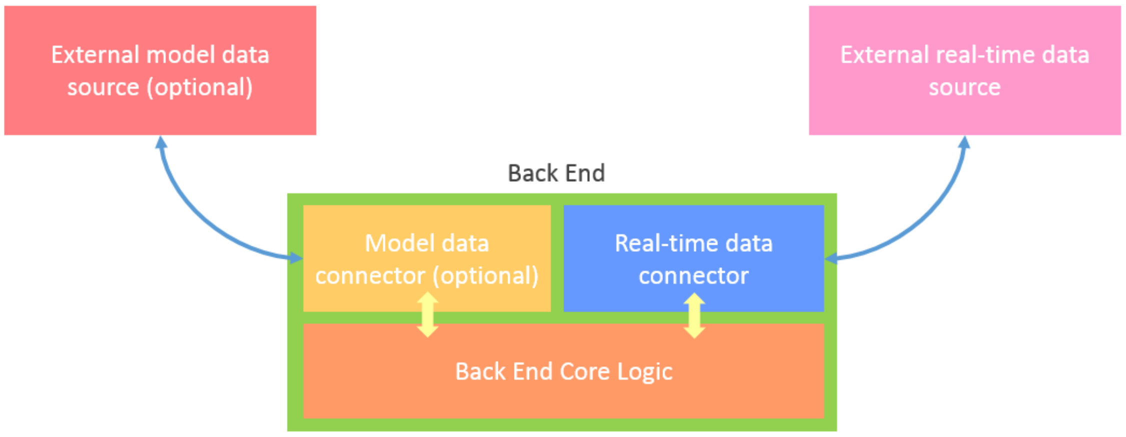

External model data source (optional)
External real-time data source
Back End
Model data connector (optional)
Real-time data connector
Back End Core Logic

Figure 19 Modularity design of the Back End component

9 Conclusion

In conclusion, this deliverable reports the activities of T3.1 - Process Modelling and Monitoring Framework. First, as background knowledge, the motivation for using BPMN in industrial scenario is explained. Different software tools for working with BPMN are introduced. The interaction of the BPMN Monitoring Framework with other components in COMPOSITION is also described.

Regarding the task in T3.1, a PCB manufacture process from BSL has been modelled in BPMN. The modelled process is presented and explained in detail. This model serves as the starting point of the later development of monitoring framework.

A BPMN Monitoring Framework has been implemented for the use case BSL-5, Equipment Monitoring and Line Visualization, to help user monitor the real-time equipment status of the production line. The requirements, design principles for the monitoring framework are presented to the readers to justify the design decision. Afterwards, further details regarding the architecture as well as interface of the monitoring framework are reported.

As T3.1 starts at M1 and finishes at M18, this document serves as the final documentation of the task activities. However, this would not limit the future activities of adjusting the implemented software to accommodate the new changes and requirements of other COMPOSITION components.

10 List of Figures and Tables

10.1 Figures

10.2 Tables

11 References

(BPMN, 2018)	Object Management Group Business Process Model and Notation: http://www.bpmn.org/
(BPMN Specification, 2011)	Object Management Group Business Process Model and Notation: http://www.omg.org/spec/BPMN/2.0/
(COMPOSITION D3.2, 2017)	http://www.composition-project.eu/about-composition/deliverables/