DIGITAL MANUFACTURING PLATFORMS FOR CONNECTED SMART FACTORIES

D5.4 Multi-Domain Simulation Framework for Multi-Stage ZDM Processes

Deliverable Id :	**D5.4**
Deliverable Name :	**Multi-Domain Simulation Framework for Multi-Stage ZDM Processes**
Status :	**final**
Dissemination Level :	**PU**
Due date of deliverable :	**30/09/2021**
Actual submission date :	**23/11/2021**
Work Package :	**WP5**
Organization name of lead contractor for this deliverable :	**Siemens**
Author(s):	**Birgit Obst (Siemens)** **Lukas Schulte (TUD)** **Marc Filz (TUBS)**
Partner(s) contributing :	

Abstract: This deliverable explaines the functionality of the simulation framework for QU4LITY. It gives an insight in the architecture and implementation and shows the application by example for validation.

Co-funded by the Horizon 2020 Programme of the European Union

Contents

HISTORY

Version	Date	Modification reason	Modified by
0.1	08/06/2021	Creation of TOC	Birgit Obst
0.2	08/20/2021	Chap. 2	Birgit Obst
0.3	09/29/2021	all	Lukas Schulte, Mark Filz, Birgit Obst
0.4	10/05/2021	revision	Birgit Obst
1.0	10/11/2021	Review improvement	Birgit Obst

Executive Summary

This report document explaines the functionality of simulation framework for QU4LITY. It gives an insight in the architecture and implementation and shows the application by example for validation.

In the first section, Introduction, the objectives from first deliverable D5.3 are reconsidered and the requirements are compared with the realized solution.
In section 2, Simulation Framework Architecture, the simulation framework architecture with its modules, interfaces and functionalities is explained.
In the third section, Validation, explains the usage of the framework with concrete examples.
As conclusion the Qu4lity Multi-Domain Simulation Framework for Multi-Stage ZDM Processes is a prototypically web service currently implemented for Plant Simulation as simulation tool. It is generic to be used for different models, and free in definition of the experiments and KPIs that should be calculated and analyzed.

The use case will be available at the QU4LITY marketplace.

1. Introduction

Objective of the deliverable

In this deliverable, the results up to project month 33 within Task 5.2, started in month 6, are presented.

Within the previous deliverable, D5.3, requirements and design decisions regarding the simulation framework architecture were described. The core of this deliverable is the presentation of the achieved implementations and validations. For this purpose, the work results achieved in the project pilots so far with respect to the developed architecture are taken into account. These results provide essential knowledge about the generally valid development of a simulation framework architecture.

Task Objectives

Taking the in Deliverable D5.3 elaborated objectives and requirements into account, the multi-domain simulation framework for multi-stage zero defect manufacturing needs the following functionalities (see also Figure 1):

1. Model: Model generation and configuration
    - A simulation model needs to be defined, loaded or adapted regarding the e.g. geometry (like CAD), the selection of components from libraries for systems or the topology of how elements are connected and are interacting.
    - In addition, the components (dimension, function,…) need to be parameterized, as well as the connections. And also to define the physical behavior (material stiffness, flow, reaction,…), coefficients for algebraic or differential equations are necessary.
    - To solve the model corresponding equation system the configuration (e.g. step size, convergence, …) of appropriate (to be selected) solver (linear, non-linear) is necessary.
2. Tool: Tool selection and interoperability
    - To execute a simulation model, the simulation engine must be selected.
    - For most of the simulation tools, a licence is necessary, so for execution the access to the licenses must be defined
    - Additionally, and especially for automated execution, all the manipulations regarding the simulation model and its evaluation need an input and output, which need to be in the correct format for import and export.
3. Scenario: Definition of simulation experiment for application and evaluation
    - Depending on the application of the simulation the evaluation delivers a specific system state (state estimation),
    - Gives a forecast, based on a specific state and based on the future boundary conditions (prognosis), or
    - Optimizes parameters for a given problem regarding defined target in several iterations.

4. State: System state definition for execution
   - To run a simulation experiment the state variables and boundary conditions need initial values. Usually a data pre-processing is necessary to define a model consistent system state and set the state values.
   - For online appliciations, a continuous model synchronization with the real system is necessary. For every synchronization step, the state values need to be set, as well as specific model parameters to also calibrate the model continuously, if necessary.
5. Visualization: Visualization as user interface
   - For some cases, it is necessary to visualize the system during model generation or adaption, to support the user in defining geometry, topology or parameter setting etc. Otherwise, the import of model defining files is an option.
   - In almost the same manner, the simulation results need to be visualized, to support the user in result interpretation in 1D, 2D or 3D plots. Otherwise, the results are exported in files.
6. Deployment: Infrastructure of simulation engine to run a application
   - Usually, simulation tools are running on a PC, but also applications using HPC infrastructures are common.
   - For distributed evaluation on local devices, simulation models are more and more able to run on edge devices, e.g. to support model predective control or advanced analytics and diagnosis.
   - For applications with high computational efforts, also distributed cloud computing is an option to run simulation.
7. Workflow: Engineering and execution of control sequence for simulation application
   - Select the appropriate simulation tool
   - Model the system behavior or adapt specific model variables
   - Define the evaluation sequence of scenarios whether the execution is a state estimation, a prognosis or an optimization
   - Interpret the simulation results and prepare them for further treatment
8. Indicator: Simulation output as contribution to quality or ZDM key performance indicator
   - The evaluation of simulation model has the aim to support in decisions. Therefore, the decision supporting indicators, as results from simulation, need to be interpreted towards the set target. This process can be done manually by the user, but also automated.
   - Regarding our vision, the simulation results should contribute to the targeted quality and ZDM key performance indicators (KPI). Therefore, the simulation outputs are semantically linked within the autonomous quality control service for performance management systems.

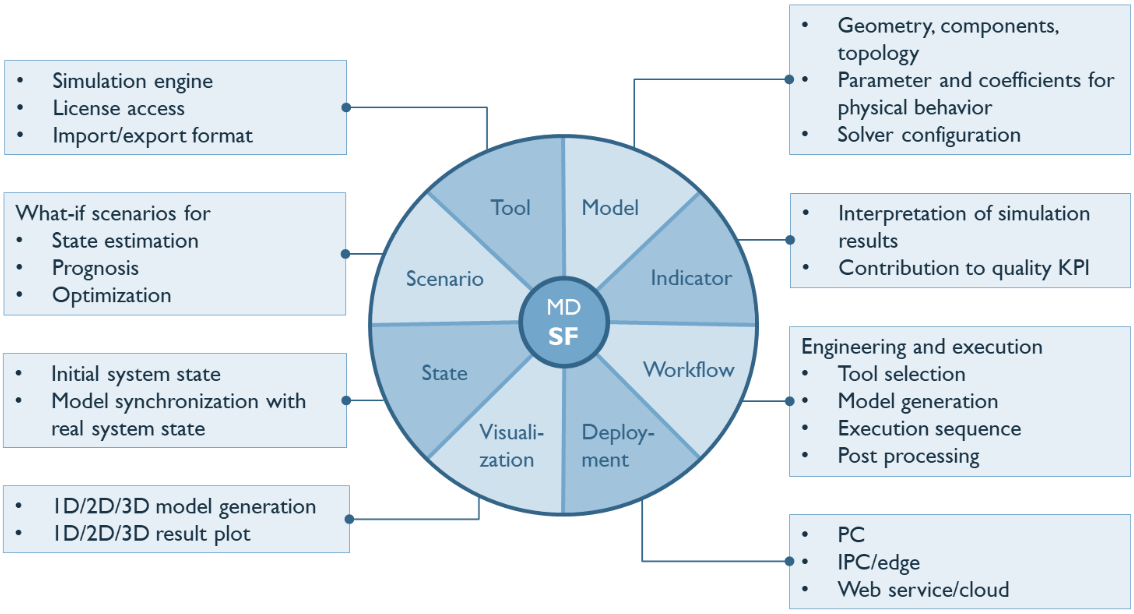


Figure 1 Multi-domain simulation framework building blocks

These functionalities define the simulation framework building blocks for multi-domain and multi-stage simulation. Quality supporting services will be able to engineer and process simulation models to evaluate the current or future status of quality and zero defect manufacturing KPIs. The multi-domain simulation framework will deliver the necessary interfaces based on these functional blocks, to get quality services within the framework architecture, as services operating and exchanging data with the data space layer connected to the industrial data space (IDS), via Open APIs (see Figure 2).

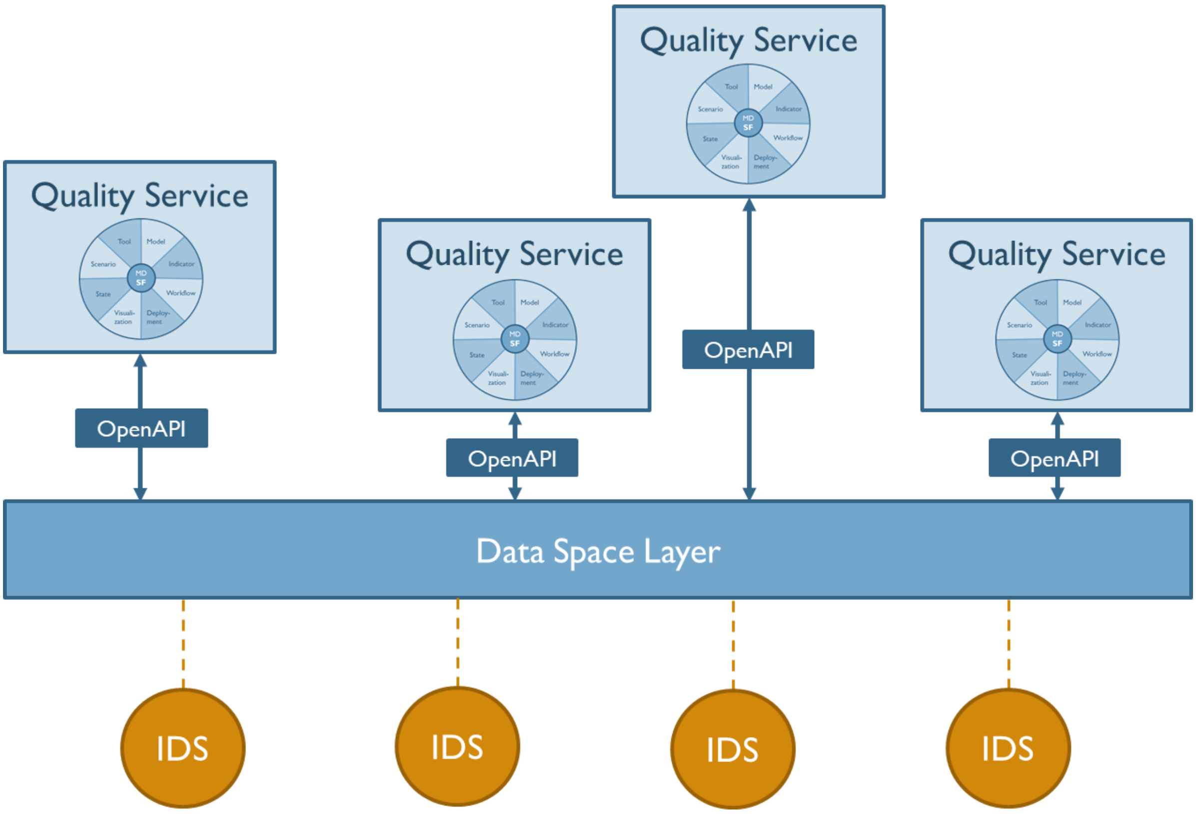


Figure 2 Multi-domain simulation framework architecture

To connect simulation tools and their simulation models for execution, it will be necessary to create a framework counterpart, as so called, wrapper for the different simulation tools. The wrapper has to support the simulation framework interfaces and has to operate with the simulation tools regarding their interoperability.

Task Solution

The multi-domain simulation framework for multi-stage zero defect manufacturing developed for QU4LITY fulfills the obove mentioned functionalities in the following approach:

1. Model: Model generation and configuration
   - Currently, the framework supports the simulation tool Plant Simulation, which is the one used by the pilots for quality issues, identified in D5.3.
   - Within Plant Simulation, the simulation model can be defined, including the topology, components and parameterization.
   - The main important element is the so called "ExperimentManager". With this Plant Simulation component, several experiments can be defined by setting model parameters and plotting the relevant results, like, e.g., KPIs.
2. Tool: Tool selection and interoperability

- For Plant Simulation, a licence is necessary.
- A Python wrapper was developed to get access to the tool engine on server side to run the experiments in an automated manner.

3. Scenario: Definition of simulation experiment for application and evaluation
    - To feed the Experiment Manager of Plant Simulation, a configuration file (e.g., Excel) can be used to define the experiments with their parameter variants.
    - For all the defined experiments, a simulation run will be executed and the results will be again stored in Excel file.
4. State: System state definition for execution
    - The initial values for the simulation run are defined in the model itself.
    - The additional pramaters from experiment configuration will be imported automatically.
5. Visualization: Visualization as user interface
    - The results stored in the Excel file can be selected and plotted on the web service visualization.
6. Deployment: Infrastructure of simulation engine to run a application
    - Due to licence issues the Plant Simulation model has to run locally.
    - The application itself is a web service realized with Mendix.
7. Workflow: Engineering and execution of control sequence for simulation application
    - The workflow is shown along the GUI.
    - Simulation model selection, configuration file selection, experiment selection, run simulation or load a existing result file and select KPIs to be plotted.
8. Indicator: Simulation output as contribution to quality or ZDM key performance indicator
    - Comparing the experiments on the plots, the analysis towards the target can be evaluated and support in decisions.

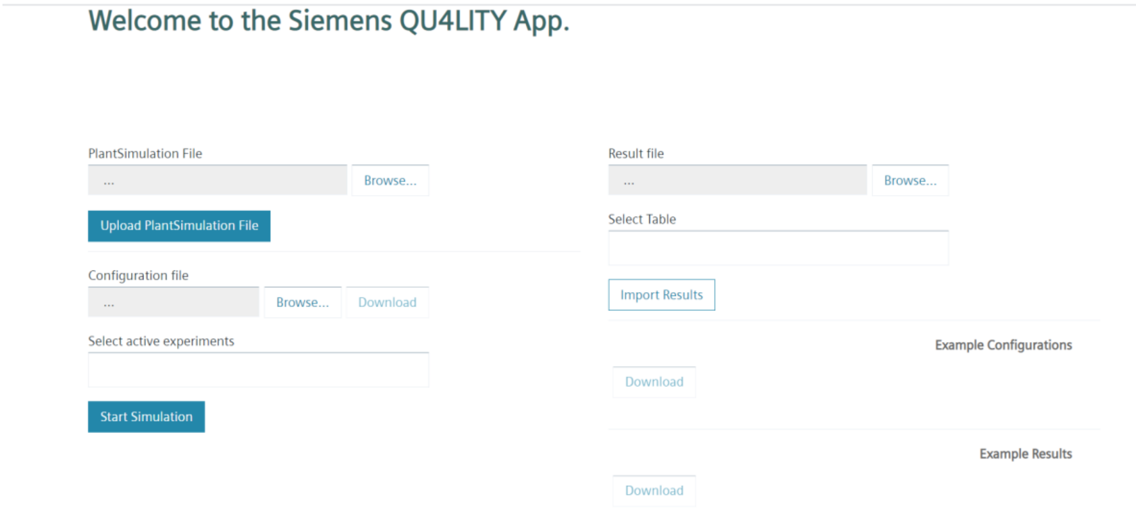


Figure 3 Graphical user interface or simulation framework

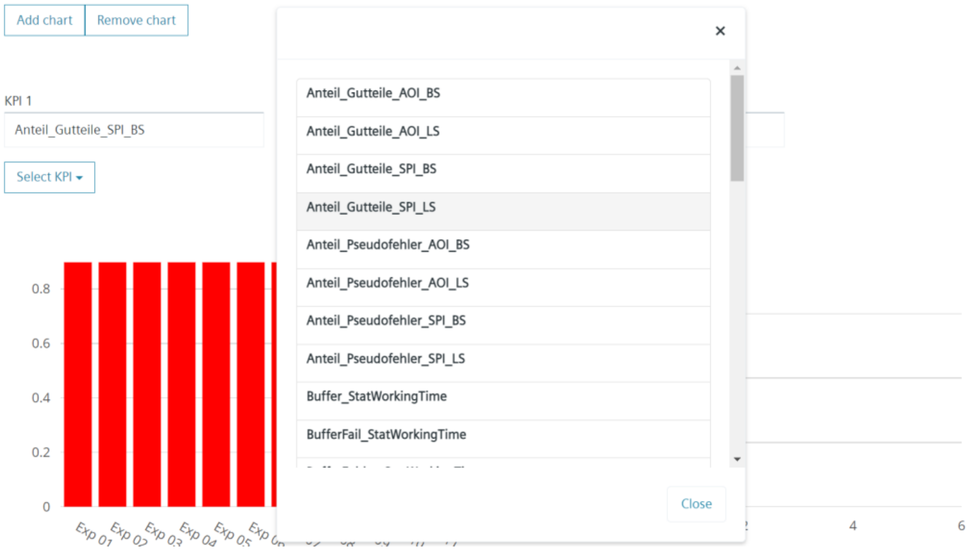
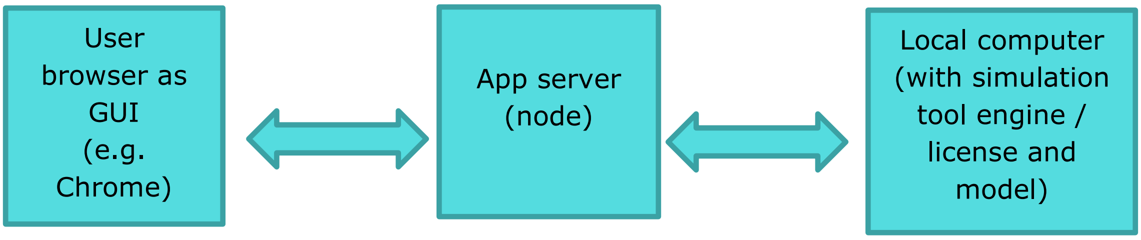


Figure 4 KPI selection

2. Simulation Framework Architecture

Modules

Modules overview:



For example:

PlantSimulation Model, Configurations, Results

MendixApp, DB

PlantSimulation, Python Server

The simulation framework has three modules: A serverside module, a clientside module and a computer where an external program is being executed. While Javascript and Nanoflows work in the user's browser, the microflows and JavaActions are proceeded by the server. PlantSimulation and Python runs on the computer.

There are several concepts of how to deploy an app, as a WebApp, a Native Mobile App, a Progressive Web App (PWA) and a hybrid app. In this case, the WebApp was chosen, so that the user is enabled to load the needed model up to the backend of the app, run it there and send the results back to the browser where they're shown. So the user needn't have access to PlantSimulation, the only circumstance he needs to know is, that his PlantSimulation model contains an ExperimentManager with root ".Models.Frame.ExperimentManager". Otherwise an error will occur in Python.

The simulation framework is developed with Mendix.

Mendix is a software to develop apps. It contains a Mendix part, a Java part and a Javascript part. We will focus on the first two.

First of all, Mendix contains a domain model, which includes all entities, in this case the entities COLUMN, KPI, FILEPATH, STRING and XLSFile.

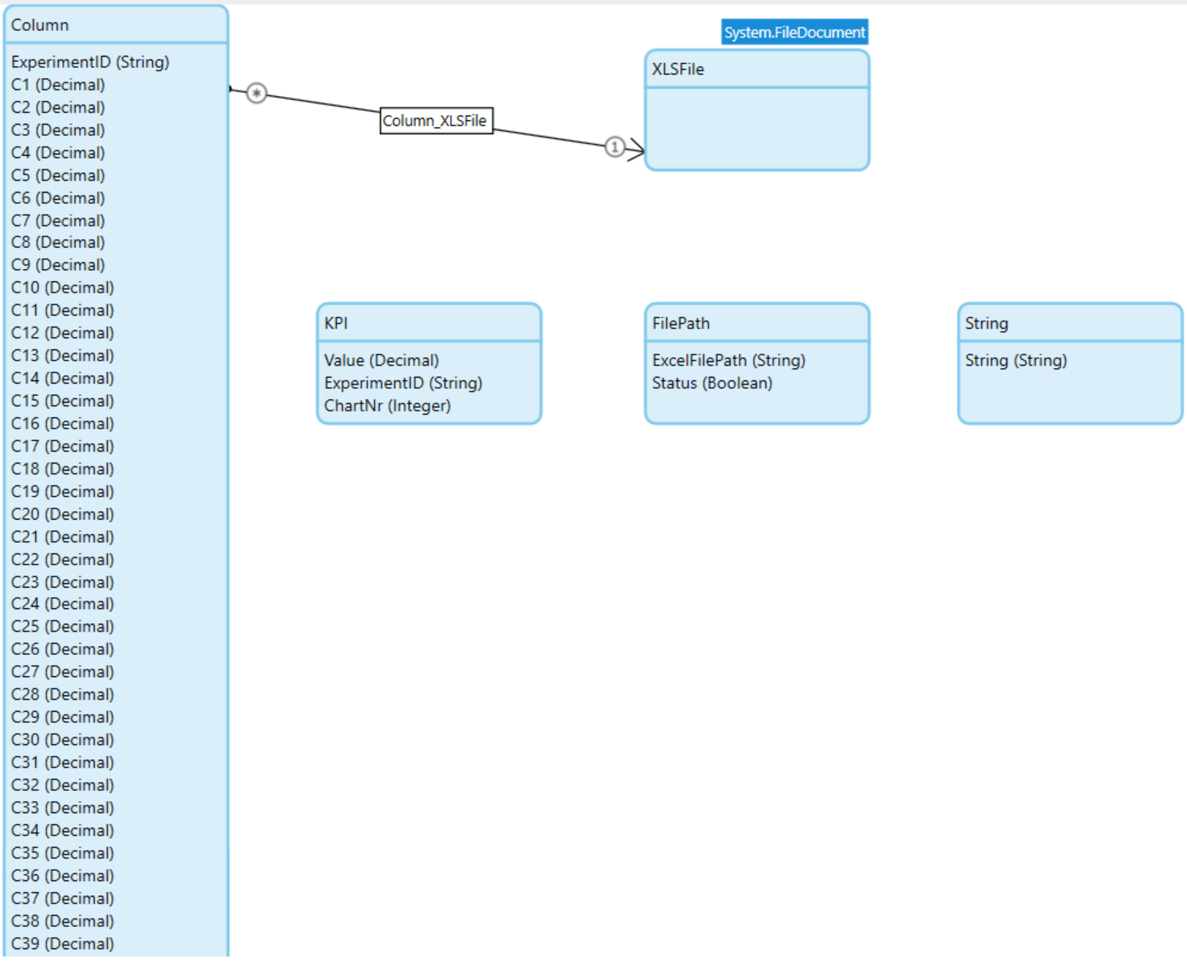


Figure 5 Domain model within Mendix

In an object of the entity **COLUMN**, the results of a single experiment are stored. Therefore, COLUMN contains many attributes, one for each column of an Excel file you want to store. By now there exists 51 columns. The name of the first column is "ExperimentID", some usual values would be "Exp 1", "Exp 01", "Exp 13", and so on.

The objects of entity **KPI** contain the values that will be displayed at the charts. It contains a "ChartNr", which determines which chart has access to this KPI. For example the third chart will display KPI's with ChartNr 3 only. The "ExperimentID" is set on the x-axis, the "value" on the y-axis.

STATUS stores some useful attributes when the user runs an experiment like the given filepath and the Boolean value if the experiments are running (true when they do, false when they don't). The user can configure several experiments in the Excel file and the select, which ones should be executed. The number of executed experiments affects the simulation runtime.

The **XLSFile** is necessary to apply Mendix "ExcelImporter", more about this will come later.

STRING is just a helper-entity which is needed if you want to set up textfields on the homepage like the "Select active Experiments" field. The reason is that textfields have to be embedded in so-called "dataviews" which need an object to store the string from the textfield. To do this, an entity that contains a string value is necessary. In fact, all STRING objects contain just one string value.

If the user uploads a configuration file, it is an object of entity XLS-File. If the mentioned user wants to run the simulation, a label will show up "The simulation is running. Please expect to wait at least one minute per experiment.". At least if he selects some experiments, he writes a string which is stored in an object of entity STRING as we have seen before.

Data Model

The ExcelImporter app:

ExcelImporter is an app offered by mendix that allows to import Excel files and store its data by a template in objects of a connected entity. It is possible to change the settings by opening the overview page of the ExcelImporter:

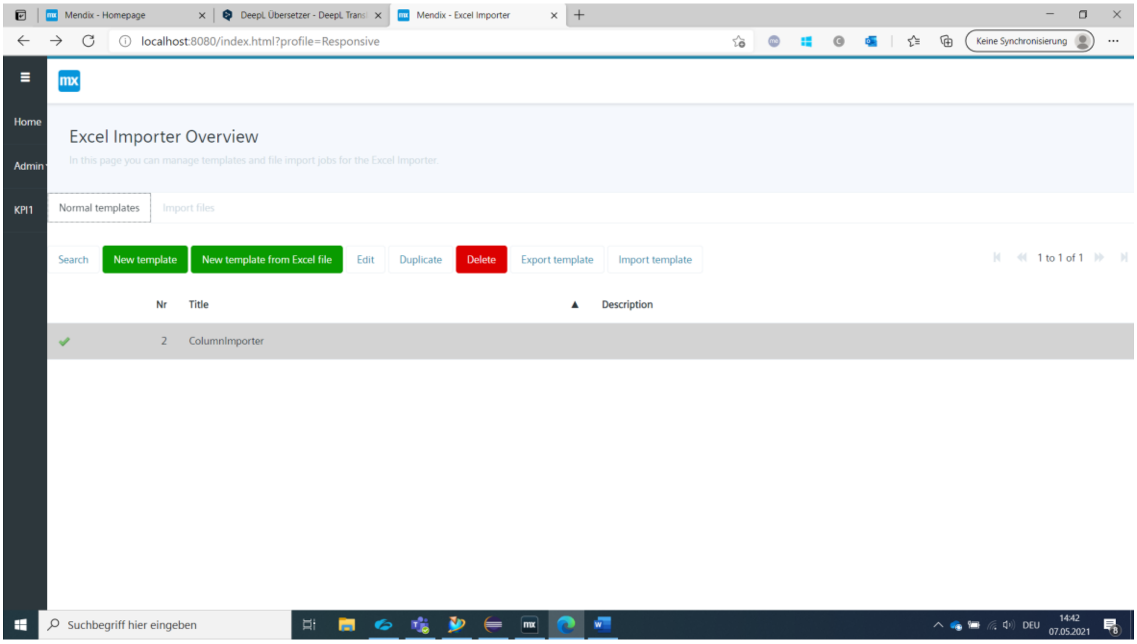


Figure 6 Excel importer

Click on the template you want to change:

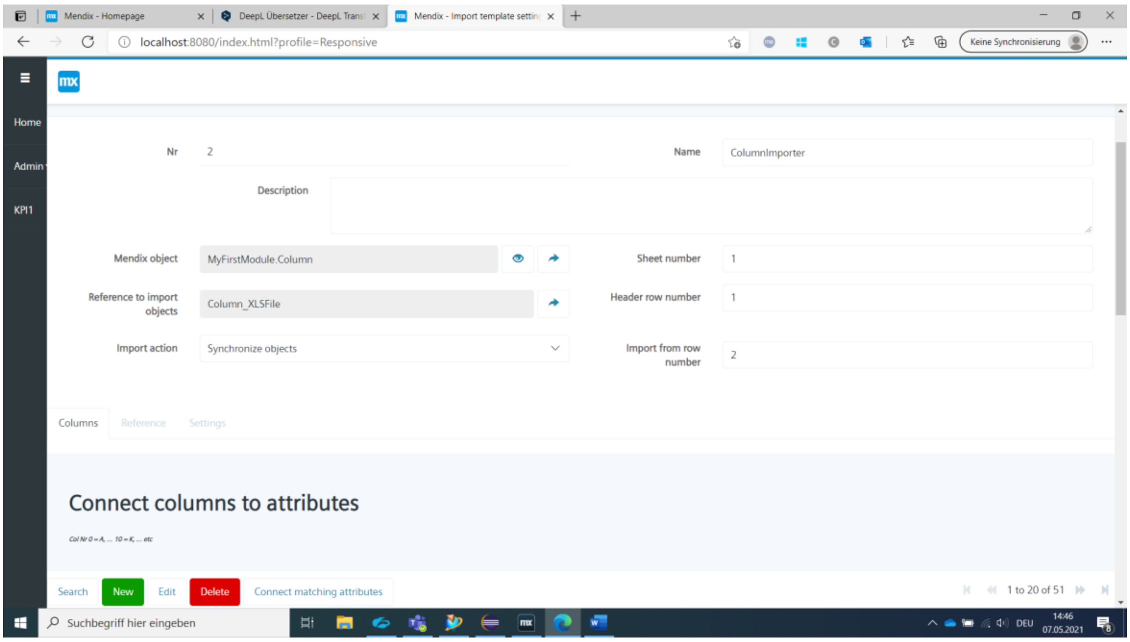


Figure 7 Settings change for Excel import

The mendix object is the object that stores the data of the Excel file. The "reference to import objects" is the XLSFile-entity related to the mendix object. If such a reference is missing, you need to create it in the data model. And, after that, to refresh the app by the MxModelReflection app. We will not discuss this case further.

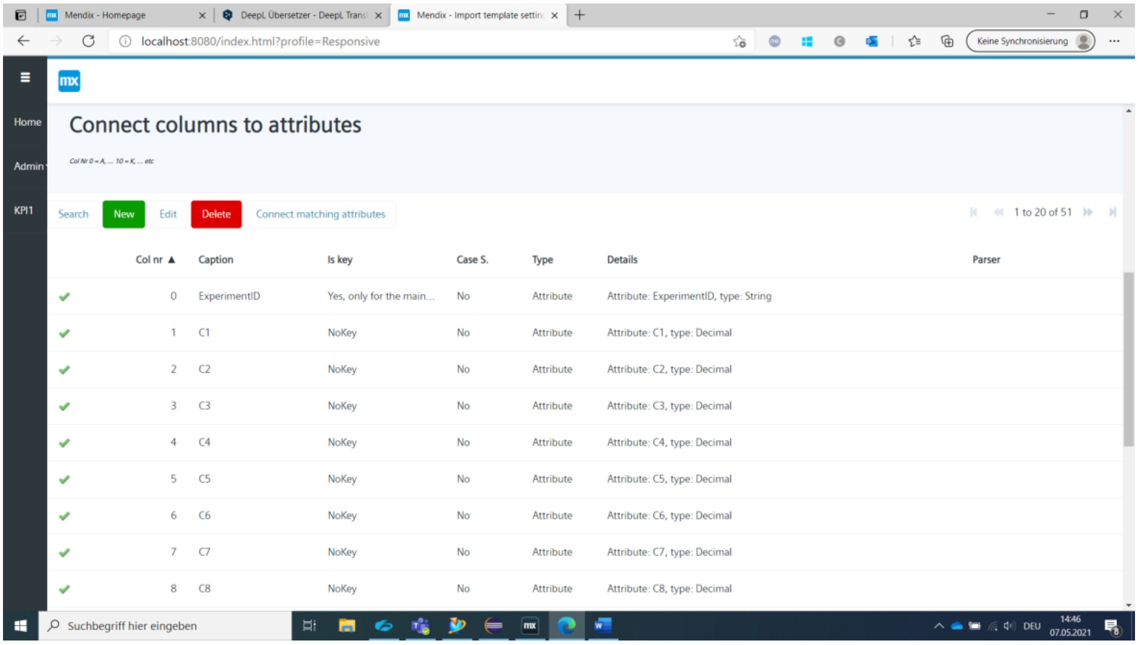


Figure 8 Data base for data model

If you want to add a new Column, click on "New":

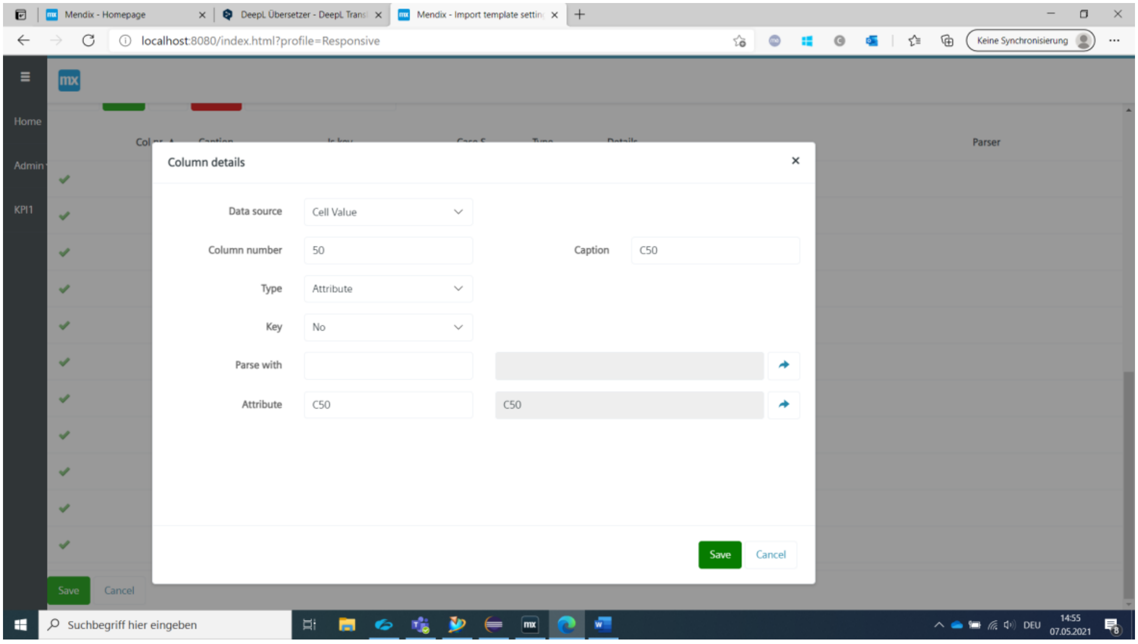


Figure 9 Add new column in data model

If your new column shares the same name as the related attribute of your entity (in this case, C50), Mendix offers a very easy way: if you created the name of your column and saved it just click on "connect matching attributes" – it will automatically connect it.

It is certainly not wise to use the overview pages to import Excel files automatically – better use the JavaAction "StartImportByTemplate". As input, it needs a template that you can get by its auto-generated number and a FileDocument (a Mendix-internal entity), either created by FileDocumentFromFile (see the COmmunityCommons App) or by the homepage from a file Manager.

The user must be aware that, by now, Mendix stores data in type decimal – so if you want to store a file with time format 'hh:mm:ss' or any other with colons, the system will run into an error. For example if PlantSimulation stores it Tables in .xls-format, it shows this behaviour. A prospective developer could change this to create a method, that set the time format automatically in the right form.

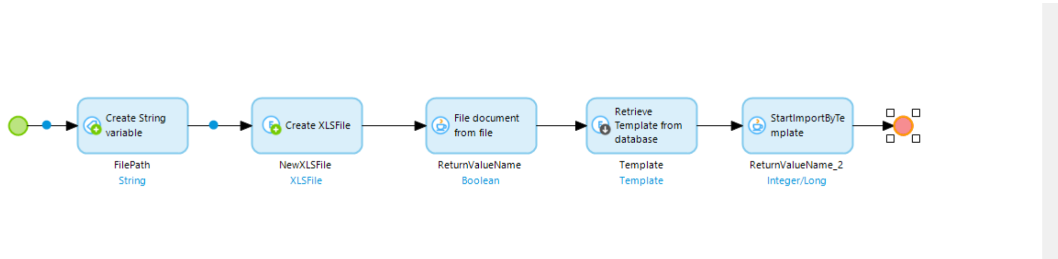


Figure 10 The ImportAutomatically microflow

At first, the filepath is created, it is always: `'C:\Temp\ExcelConnection.xlsx'`. Then, an empty XLSFile is created and –by the FileDocumentFromFile JavaAction–

filled with the data of the ExcelConnection file. After that, the JavaAction "StartImportByTemplate" of the ExcelImporter app is storing the data in the database.

Interfaces

An overlook over some APIs:

PlantSimulation offers several APIs:

- COM: With this interface, it is possible to start PlantSimulation, to load a model and nearly everything you can do in PlantSimulation. It is unfortunately difficult to connect by Java (but there are some projects on GitHub that worked on that issue), but Python does not. Hence, it is necessary that Java calls Python to do this task.
- Socket: it will not be discussed here.
- HTML: An API which is callable from the browser, the setup is as following: Click with the right key on PlantSimulation, open "properties" and write in "aim" " -WebServer:30001". Now, PlantSimulation can be controlled by the opening of certain windows (look in the PlantSimulation guide). Unfortunately, this API is extremely slow, since loading a website takes too much time (and on the other hand does not look nice and clean).
- OPCUA (and OPC CLassic): There exists a OPCU API in Mendix in the AppStore, but the EventController of PlantSimulation can only run in real-time which would extremely slow down the simulation.
- C: To connect a C API by Java seems not to be trivial.
- PLCSIM Advanced: special tool.
- SIMIT: special tool.
- Teamcenter: special tool.

How Mendix sends to PlantSimulation:

The connection between both is set up with a Python Flask Server. Mendix calls it via a REST API; and the Python Server calls itself PlantSimulation by a COM API. As stated by other programmers, the COM API is a little bit outdated, but it fulfills its purpose.

To have access to the COM API (and to many other packages of python), you have to know, how to import them; this can be done via pip.

The REST API is a standard service which include calls like "GET", "POST", "PUT" …

Importing an experiment

If an experiment is going to run, Mendix sends the configuration of the user per REST API (call 'POST') to the Flask Server, which stores it and loads it in the ExperimentManager of the PlantSimulation Model. That is all.

Execution

Plant Simulation file

The user can browse for the Plant Simulation model and then upload it to the server.

Configuration file

The user can browse for configuration Excel file.

Select Active experiments

If the user does not want to run all experiments of the configuration file, the user is able to select specific experiments, for example by "1,2,3" or "1;2;3" or "1-3" or "1,2;3" in the "Select active experiments" GUI field.

The app will recognize the string and then select the right experiments.

Run simulation

At first, all experiments are set internally to "False". After reading the string, the selected experiments will be set to "True". After that the ExperimentManager is started and an end is awaited. The results are send back to the Mendixserver, and there they are stored and imported, so that the user can read and get them.

Adding or removing a chart

Mendix does not really add or remove charts, it just changes their visibility status. (It is possible to extend for more charts).

Changing a KPI

If the user change a KPI in a chart with ChartNr n, it is possible to delete and change all of the KPIs with the same ChartNr. Mendix select by the zeroth row of the Excel file in which column the searched KPI is, and, after that, stores the connected attribute of the COLUMN-objects in a KPI object.

Import Result file

If the user wishes to import an existing results file in xlsx-Format, it is possible to just browse for it.

At first, a String with the FilePath "C:/Temp/ExcelConnection.xlsx" will be created to later tell the ImportAutomatically-Microflow to import the file at the given filepath. Before that, Mendix stores the given FileDocument (which is just Mendix-internal) as a proper File at "C:/Temp/ExcelConnection.xlsx". This is necessary, since the "HeaderSearcher" needs to read the first row. The method FileFromFileDocument is part of the CommunityCommonsApp maintained by Mendix. At the third step, Mendix deletes the whole Database to create space for the new results.

At last, ImportAutomatically stores the file up to a maximum of 50 columns. If you want to know how this works, and, why, by now, the xlsx-Format is necessary, please have a look into "How Mendix stores Data".

3. Validation

Functionality

The process flows and process parameters of the manufacturing process chain of the use case are transferred to a simulation model using the software Plant Simulation, in order to evaluate the impact of in-line quality monitoring measures and to be able to analyze the entire process chain in the process. The aim of this section is to explain the created simulation model by using different screenshots, control methods, and tables that were used in the software as part of the simulation setup. Figure 11 provides an overview of the simulation model. The creation of the simulation model and the Execution of the simulation experiments are done in a structured manner considering all the points mentioned until now.

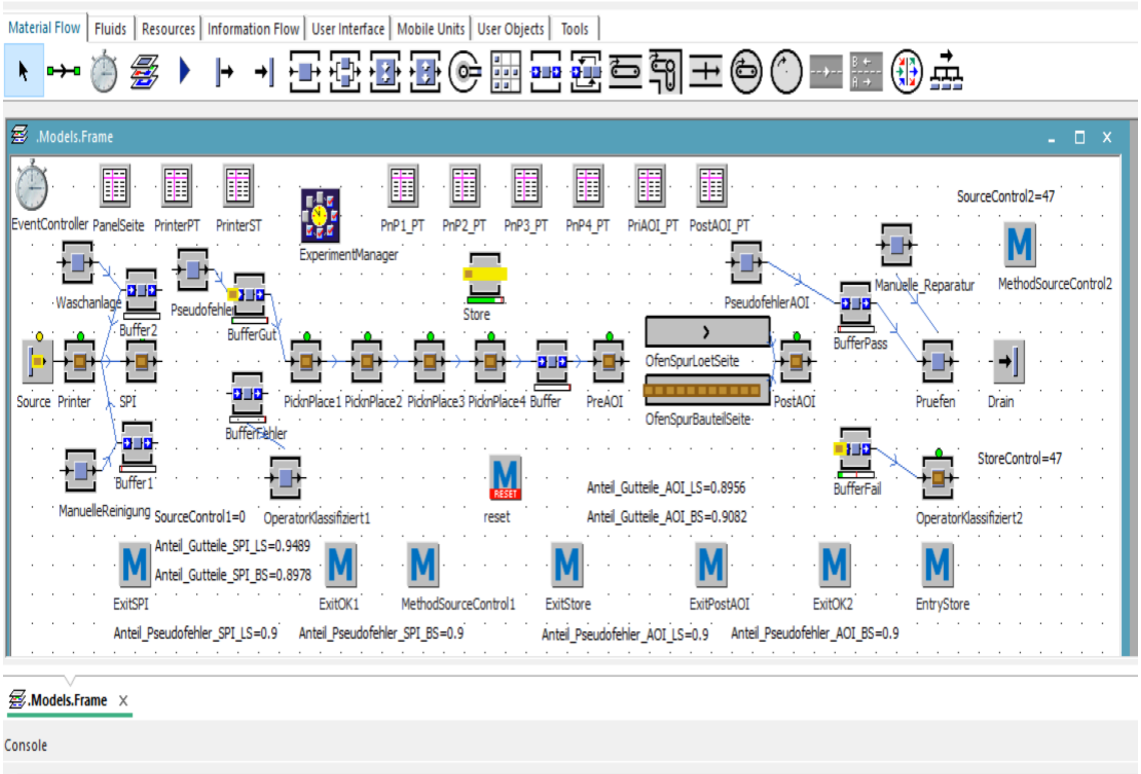


Figure 11: Overview of Plat Simulation Model

The simulation model shows a production line for PCB (printed circuit board) with production steps and test stations. The main production steps are printer, pick and place 1-4, buffer and reflow oven. The main quality gates are SPI and AOI, with option PreAOI and PostAOI.

In order to be able to analyze and evaluate the effects of the quality gates SPI and AOI, various simulation experiments are carried out. The focus here is on the influence of the pseudo-defects on the entire manufacturing process chain. The following questions concerning the system behavior and possible improvements of the manufacturing process will be discussed during the simulation experiment:

- How does the production system behave if the proportion of the pseudo-error rate changes?
- Does the current pseudo-error rate of the system lead to high productivity loss?
- To what extent can the entire manufacturing system be improved to a zero defect rate?
- And To what extent can the overall throughput be improved if the current 100% inspection rate changes?

In terms of inspection strategies, these simulation experiments can be summarized into three different scenarios.

1. In the first scenario, the proportions of pseudo-defects are varied from 0% or no pseudo-defects to 20% of each side for each quality gate.
2. In scenario two, the Quality SPI is turned off and only the Quality Gate AOI is used.
3. In scenario three, a random sample test is tested instead of the current 100% test coverage.

Scenario 1: Effect of pseudo-errors on manufacturing system

The inspection of the printed circuit boards using the quality gates SPI and AOI is performed in two steps, according to the real manufacturing process and the created simulation model. In the first step, the PCBs are classified by each quality gate into good parts and defective parts based on the proportion of the total defect rate. The parts which are to be recognized as pseudo-defects are included in the defective parts. In the second step, a distinction is further made between the real defects and the pseudo-defects by the operator and on the basis of the proportion of the pseudo-defect rate.

In order to analyze the influence of the pseudo-error on the entire manufacturing process and to test different and to be able to test different pseudo-defect rates, it is assumed in the simulation model that the proportion of real errors does not change. This means that the simulation experiments are to be carried out while maintaining the proportion of real errors. This assumption aims at evaluating the effect of the pseudo-faults independently of the proportion of the real errors. To achieve this goal, here essentially between the proportion of the pseudo-faults or the genuine faults from the defective parts and the proportion of pseudo-defects or real defects of the total parts produced.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	root.Anteil_Gutteile	l_Pseudofeh	teil_Gutteile	l_Pseudofeh	teil_Gutteile	l_Pseudofeh	teil_Gutteile	l_Pseudofeh	oot.test_pro	.drain.statNu	rKlassifiziert	assifiziert1.s	dofehlerSPI.S	ehlerSPI.stat	rKlassifiziert	assifiziert2.St
2	Exp 01	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	1	11872,6	0	0	0	0	13398,6	2206616,41
3	Exp 02	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,9	0	0	0	0	0	29,7	4677,59163
4	Exp 03	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,8	0	0	0	0	0	29,4	4556,05257
5	Exp 04	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,7	0	0	0	0	0	25,8	4043,92731
6	Exp 05	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,6	0	0	0	0	0	22,5	3465,00638
7	Exp 06	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,5	0	0	0	0	0	18,5	2803,62217
8	Exp 07	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,4	0	0	0	0	0	14,4	2192,17321
9	Exp 08	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,3	0	0	0	0	0	7,6	1130,45969
10	Exp 09	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,2	0	0	0	0	0	5	744,810995
11	Exp 10	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,1	0	0	0	0	0	1,5	222,931072
12	Exp 11	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0	0	0	0	0	0	0	0

Figure 12 Configuration file for Scenario 1

Figure 13 Resulting KPIs for Scenario 1

Scenario 2: Disable Quality Gate SPI

In this scenario, it is assumed that only the Quality Gate AOI is used for inspecting. The goal here is to eliminate the pseudo-error rate of the quality gate SPI, by using only one quality gate as the final inspection. Two experiments are performed and are compared to the baseline. In the first experiment, the Quality Gate AOI is used to test the solder side and the component side. In the second experiment, the Quality Gate AOI is used only for testing the component side. In addition, the proportions of the real defects are not changed in each experiment and are converted to the converted to the available quality gate.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	root.Anteil_Gutteile	l_Pseudofeh	teil_Gutteile	l_Pseudofeh	teil_Gutteile	l_Pseudofeh	teil_Gutteile	l_Pseudofehl	.drain.statNu	rKlassifiziert	assifiziert1.s	dofehlerSPI.S	ehlerSPI.stat	rKlassifiziert	assifiziert2.S	lofehlerAOI.S
2	Exp 1	0,9489	0,9	0,8978	0,9	0,9082	0,9	0	0,9	37055,8	0	0	0	0	7781,6	1282100,33	7006,2
3	Exp 2	0	0	0	0	0,898	0,858	0	0,8099	36115,7	0	0	0	0	7957,4	1310824,67	6631,5
4	Exp 3	0	0	0	0	0,8824	0	0	0,7028	34633,8	0	0	0	0	8170,9	1345541,66	6376,8

Figure 14 Configuration file for Senario 2

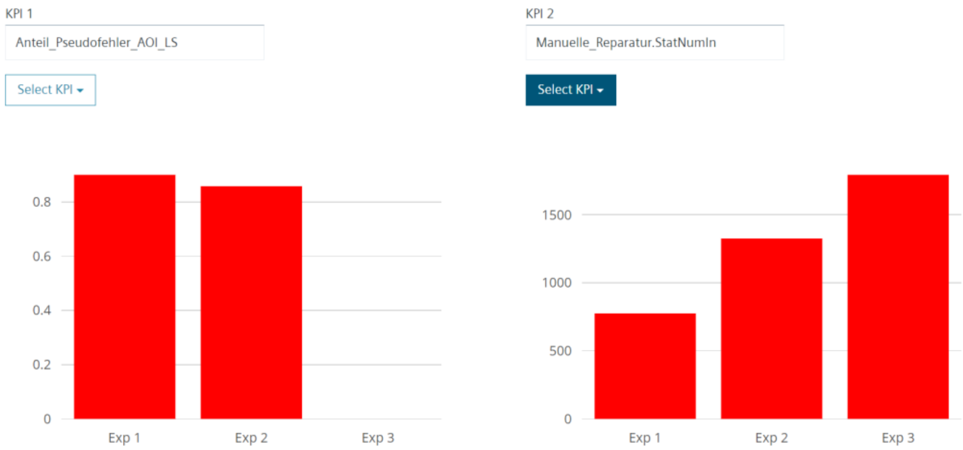


Figure 15 Resulting KPIs for Scenario 2

Scenario 3: Random Inspection

As part of the initial situation of the manufacturing process, all parts are inspected by the two quality gates: SPI and AOI. This inspection strategy is to be named as 100% inspection scope. To determine the impact of the 100% inspection scope on the performance of the manufacturing process, in this scenario a random sample inspection is simulated. Here it is assumes that not all parts pass through the quality gates. In 10% steps the parts are steered through the two quality gates from 100% (initial position) to 0% (no inspection). In this way, the inspection range is tested from 0% to 100%. This assumption represents a quality risk, since the proportions of true defects of the are only applied to the parts to be inspected.

The results of different simulation scenarios can then be analyzed based on proper graphical representation.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	root.Anteil_Gutteile	l_Pseudofeh	teil_Gutteile	l_Pseudofeh	teil_Gutteile	l_Pseudofeh	teil_Gutteile	l_Pseudofehl	ifwahrschein	.drain.statNu	rKlassifiziert	assifiziert1.s	dofehlerSPI.S	ehlerSPI.stat	rKlassifiziert	assifiziert2.St
2	Exp 01	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	1	11868,7	0	0	0	0	13225,2	2178304,78
3	Exp 02	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,9	12934,6	0	0	0	0	13004	2141735,21
4	Exp 03	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,8	14214,7	0	0	0	0	12675,7	2087837,7
5	Exp 04	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,7	15857,2	0	0	0	0	12326,2	2029898,87
6	Exp 05	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,6	17758	0	0	0	0	11843,8	1950145,95
7	Exp 06	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,5	20277,1	0	0	0	0	11243,3	1851238,51
8	Exp 07	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,4	23577,7	0	0	0	0	10444,9	1719428,98
9	Exp 08	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,3	28081,5	0	0	0	0	9341,6	1537789,26
10	Exp 09	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,2	34170	0	0	0	0	7582,6	1249298,92
11	Exp 10	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0,1	37577,7	0	0	0	0	4181,2	687820,09
12	Exp 11	0,9489	0,9	0,8978	0,9	0	0,9	0	0,9	0	38100	0	0	0	0	0	0

Figure 16 Configuration file for Scenario 3

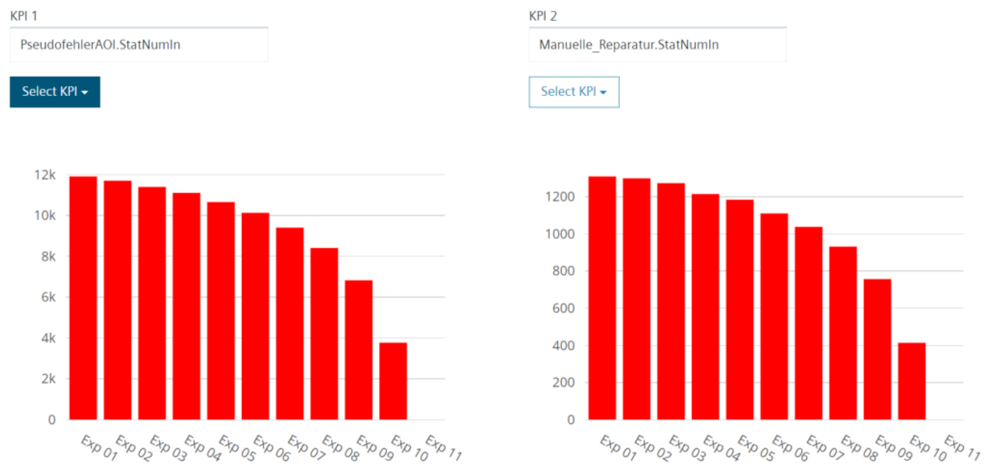


Figure 17 Resulting KPIs for Scenario 3

Deployment

The Mendix Aplication is deployed on a Node. It is available by:

https://siemensappqu4lity-sandbox.mxapps.io/

To run the simulation, it is necessary to have access to the app server. As the QU4LITY project has no server available for all the project partners, the server is hosted at Siemens, unfortunately with no external access.

The simulation framework is also a solution for the QU4LITY marketplace. There an application video will be available.

4. Conclusion

The QU4LITY Simulation Framework is a prototype to show the functionalities and how to use simulation for multi-domain, multi-stage zero defect manufacturing services.

The solution fulfills the requirements defined for simulation framework in D5.3.

Currently, it is implemented for Plant Simulation as simulation tool. It is generic to be used for different Plant Simulation models, and free in definition of the experiments and KPIs that should be calculated and analyzed.

With the web service it is very easy to get access to the evaluation process and to be used from several users at several locations.

The use case will be available at the QU4LITY marketplace.

List of figures

List of Abreviations

Abbreviation	Explanation
AOI	Automatic optical inspection
AQ	Autonomous Quality
AR	Augmented Reality
CAD	Computer-aided drafting
CAE	Computer-aided Engineering
CAM	Computer-aided Manufacturing
DES	Discrete Event Simulation
DSS	Decision Support System
ERP	Enterprise Resource Planning
FMI	Functional Mock-up INterface
HPC	High Performance Computing
IDS	Industrial Data Space
KPI	Key Performance Indicator
MES	Manufacturing execution system
NC	numerical control
PCB	Printed Circuit Board
PDM	Product data management
PLM	Product lifecycle Management
SPI	Solder paste inspection
WP	Work Package
ZDM	Zero defect manufacturing

References

Jahangirian, Mohsen, Tillal Eldabi, Aisha Naseer, Lampros K. Stergioulas, and Terry Young. "Simulation in manufacturing and business: A review." *European Journal of Operational Research* 203, no. 1 (2010): 1-13.

Mourtzis, D., M. Doukas, and D. Bernidaki. "Simulation in Manufacturing: Review and Challenges." *Procedia CIRP*, 2014.

Obst, B., J. Fischer, and B. Lee. "Harmonization of generic simulation and specific parameterized models into one Simulation Environment." D4.1, 2017.

Psarommatis, Foivos, Gökan May, Paul-Arthur Dreyfus, and Dimitris Kiritsis. "Zero defect manufacturing: state-of-the-art review, shortcomings and future directions in research." *International Journal of Production Research*, 2019.

Partners:

INTELLIMECH Consorzio per la Meccatronica | SIEMENS | AtoS

AIRBUS | MONDRAGON | NXT CONTROL

ATB Institut für angewandte Systemtechnik Bremen GmbH | Continental | RIA STONE

UNIMETRIK Metrology and Calibration | Technology Transfer System | EPFL

SQS | FAGOR FAGOR ARRASATE | TTTech Ensuring Reliable Networks

technische universität dortmund | thyssenkrupp | Jožef Stefan Institute Ljubljana, Slovenia

PRIMA INDUSTRIE | eit Digital | ENGINEERING

PACE a TXT company | innovalia association | Telefonica

Synesis sustainable automation | ghi | UNPARALLEL

DANOBAT | POLITECNICO MILANO 1863 | Whirlpool

VTT | VISUAL COMPONENTS | ASTI

Fraunhofer | cea | INTERNATIONAL DATA SPACES ASSOCIATION

PHILIPS | ATLANTIS | aic AUTOMOTIVE INTELLIGENCE CENTER

KOLEKTOR | INTRASOFT INTERNATIONAL | SINTEF

+GF+ | Technische Universität Braunschweig | TNO innovation for life

TTTech industrial