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

D3.5 Computational Modelling, Simulation and Prediction of Production II

Date: 2018-12-21

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: D3.5 Computational Modelling, Simulation and Prediction of Production II
Document version: 1.0
Document owner: CERTH

Work package: WP3 – Manufacturing Modelling and Simulation
Task: T3.3 – Simulation and Forecasting in Production and Logistics
Deliverable type: R

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
0.1	Thanasis Vafeiadis, Alexandros Nizamis (CERTH)	2018-10-30	ToC and Initial text
0.2	Thanasis Vafeiadis (CERTH)	2018-11-20	Input and updates for Sections 5 and 6
0.3	Alexandros Nizamis, Nikolaos Alexopoulos, Nikolaos Vakakis, Christos Ntinas (CERTH)	2018-11-21	Input and updates for Sections 4, 6 and 7 related to other WPs (architecture, sensors and deployment) connected with SFT
0.4	Vasiliki Charisi (ATLANTIS)	2018-11-23	Input and updates for Sections 4
0.5	Vagia Rousopoulou (CERTH)	2018-12-10	Input and updates for Sections 6
0.6	Thanasis Vafeiadis, Alexandros Nizamis (CERTH)	2018-12-10	Input in Sections 3 and 8
1.0	Thanasis Vafeiadis, Vagia Rousopoulou (CERTH)	2018-12-21	Final version

Internal review history:

Reviewed by	Date	Summary of comments
Patrick McCallion (BSL)	2018-12-17	Minor corrections and suggestions
Theofilos Mastos (KLE)	2018-12-17	Minor corrections and suggestions

INDEX

1 Executive Summary

The present document is a deliverable of the "***Ecosystem for COllaborative Manufacturing PrOceSses – Intra- and Interfactory Integration and AutomaTION***" - (COMPOSITION) project, funded by the European Commission's Directorate - General for Research and Innovation (D-G RTD) under Horizon 2020 Research and Innovation programme (H2020). The deliverable presents the final version of Computational Modelling, Simulation and Prediction of Production developed until M28 of the project.

The COMPOSITION project needs simulation and prediction in both intra- and inter-factory scenarios. This report is focused on intra-factory scenarios which are related to production. In this first stage of the project and Task 3.3 - Simulation and Forecasting in Production and Logistics the research has been conducted to use cases which were selected as the cases with the highest priority from both pilot and technical partners of the project. Besides that, the first steps of the work been done to the rest of the use cases are also presented in this document.

2 Abbreviations and Acronyms

Table 1: Abbreviations and acronyms used in this deliverable

Acronym	Meaning
API	Application Programming Interface
CMMS	Computerised Maintenance Management System
DFM	Digital Factory Model
DSS	Decision Support System
ERP	Enterprise Resource Planning
LOF	Local Outlier Factor
MQTT	Message Queuing Telemetry Transport
MVDP	Machine Vibration Diagnosis Profile
NFA	Nondeterministic Finite – state Automata
SSP	Slope Statistic Profile
VA	Visual Analytics
OGC	Open Geospatial Consortium
JSON	JavaScript Object Notation
SFT	Simulation Forecasting Tool
REST	Representational State Transfer
M2M	Machine-to-Machine
CSV	Comma Separated Value
KPI	Key Performance Indicator
BMS	Building Management System

3 Introduction

3.1 Purpose, context and scope of this deliverable

This document presents the computational modelling, simulation and prediction functions on production developed until M28 of the COMPOSITION project. This document is part of the "Task 3.3 – Simulation and Forecasting in Production and Logistics" and aims to design and implement trend analysis techniques for linear trend analysis, an anomaly detection methodology on 3D time series from vibration sensor and the application of Dijkstra's algorithm in order to find the shortest path tree from a single source. This deliverable defines the final approaches for the core set of algorithms, techniques and methodologies dedicated on predictive maintenance. With the implementation of such techniques we aim to provide detection of possible deviations from normal conditions.

3.2 Content and structure of this deliverable

The content of this deliverable is organized as follows:
In Section 4 the final project architecture of the Simulation and Forecasting tool and a description of internal components (digital factory model, visual analytics, decision support system etc.) is provided, while in Section 5 a brief description of the data used for each use case that belongs to the area of computational modelling, simulation and prediction of production is provided. In Section 6 a description of the functions and methodologies developed (new) and utilized or modified (existing ones from scientific literature) until M28 of the project, along with their application on the use cases is provided. In Section 7 the deployment plan of Simulation and Forecasting Tool is briefly described and in Section 8 we draw our conclusions.

4 Simulation and Forecasting tool in Overall COMPOSITION Architecture

This section describes the position of the Simulation and Forecasting Tool in the COMPOSITION project's overall architecture. The main interactions of this component with the rest of the project's components are described too.

4.1 Overview

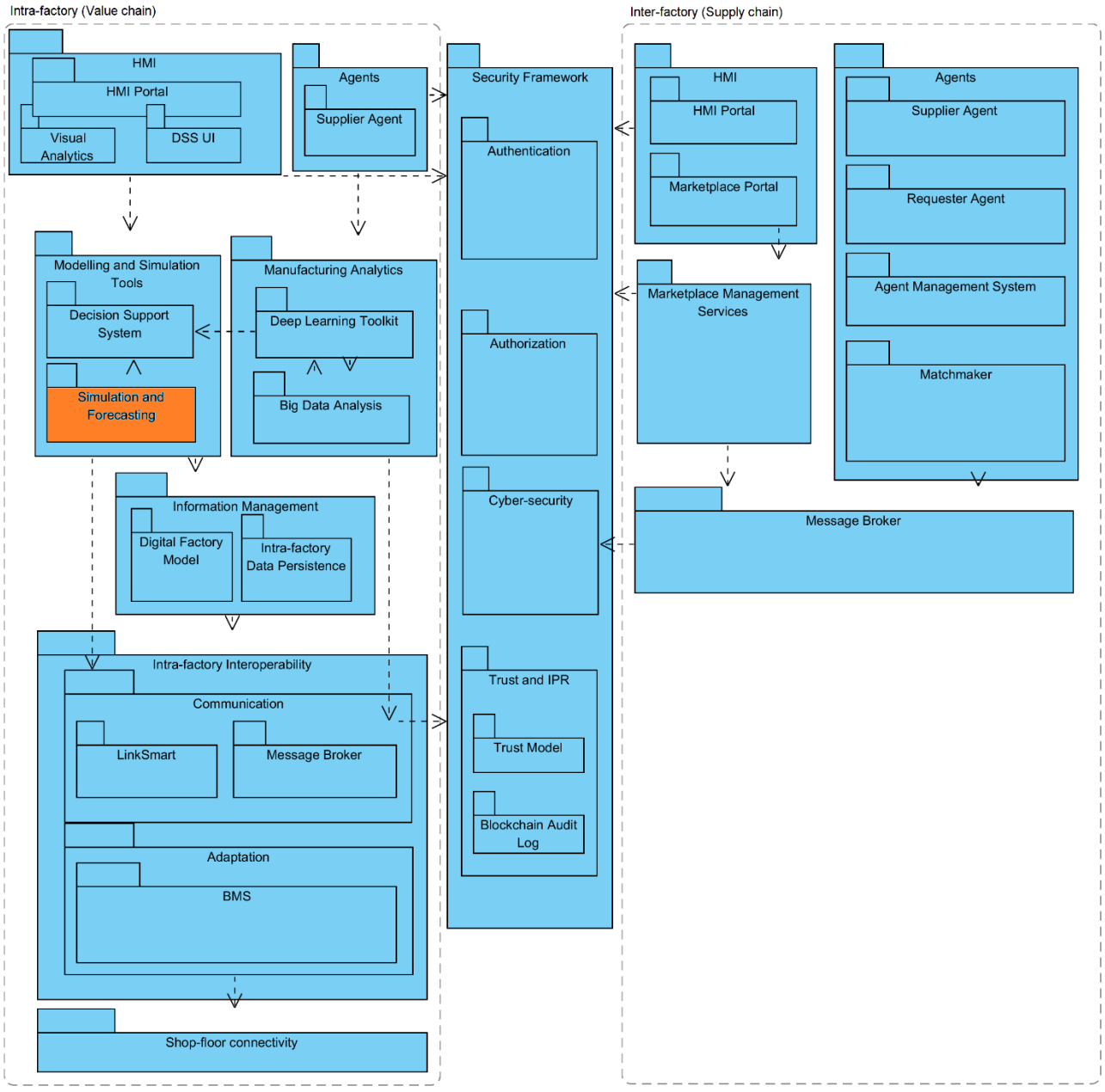


Figure 1: COMPOSITION architecture functional view

As depicted in Figure 1, the Simulation and Forecasting tool belongs to the Intra-factory package. More precisely, it is part of the Modelling and Simulation Tools module as described in *D2.4 - The COMPOSITION architecture specification II*. The Simulation and Forecasting tool is strongly correlated with DSS, Visual Analytics and DFM components. Furthermore, it is also correlated with the Event Broker. The Broker will be responsible to transfer a Simulation tool's prediction to the Agent Marketplace for the project's Interfactory

scenarios. As the correlation with the Event Broker is only matter of connectivity with the Marketplace, we decided to present in more details only the interaction with DFM, Visual Analytics and DSS components.

4.2 Digital Factory Model

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.

The Simulation and Forecasting tool is able to load data from DFM using the DFM API's services. Besides that the Forecasting tool's predictions become available to other components using DFM API and the corresponding factory instances. Every prediction is described as an Event using the defined format from the DFM schema. More precisely, the DFM Event format is covered by OGC Observation and Measurements JSON format. Every prediction will be displayed as an Observation. In the next figure, an example of Forecasting tool output for UC-KLE 1 Maintenance decision support in terms of DFM schema is presented. The output presents the probability of future faults (electrical fault and mechanical fault) as a collection of OGC observations.


```
{
    "member": [
        {
            "href": "http://www.composition-project.eu/KLE_1_Fault_Probability"
        },
        {
            "resultTime": "09:02:36.656514",
            "result": {
                "uom": "http://www.composition-project.eu/uom#percentage",
                "value": 0.06592865636147 44
            },
            "observedProperty": {
                "href": "http://www.composition-project.eu/bossi_fault_probability"
            },
            "id": "electrical_fault_id",
            "procedure": {
                "href": "http://www.composition-project.eu/predictive_maintenanceKLE"
            },
            "type": "CategoryObservation"
        },
        {
            "resultTime": "09:02:36.656514",
            "result": {
                "uom": "http://www.composition-project.eu/uom#percentage",
                "value": 0.06956004756242569
            },
            "observedProperty": {
                "href": "http://www.composition-project.eu/bossi_fault_probability"
            },
            "id": "mechanical_fault_id",
            "procedure": {
                "href": "http://www.composition-project.eu/predictive_maintenanceKLE"
            },
            "type": "CategoryObservation"
        }
    ],
    "phenomenonTime": {
        "instant": "09:02:36.656514"
    },
    "featureOfInterest": {
        "href": "http://www.composition-project.eu/faultProbability"
    },
    "_id": {
        "$oid": "5a7b0431c4e4cd27a4082c41"
    },
    "id": "KLE_1_SFT_Event"
}
```


Figure 2: SFT output as DFM event

As soon as the SFT predictions posted to a DFM instance, the DSS component is able to get the posted probabilities from the aforementioned DFM instance by using the *getObservation* service from DFM API.

Therefore, the usage of DFM instance and DFM APIs service enables the communication between DSS and Simulation and Forecasting tool.

More details about DFM are available at *D3.3 Digital Factory Model II*, which is published at M26.

4.3 Visual Analytics Tool

The COMPOSITION Visual Analytics (VA) tool imports and visualize data from Simulation and forecasting tool and Big Data Analytics tools. The VA offers an interactive user interface for the SFT algorithms and apply visual analytics techniques in order to present the output to the users as graphical representations. The Visual Analytics tool will provide the ability to manufacturers/end-users to evaluate the SFT results and identify possible problems. Based on the COMPOSITION architecture, the VA was designed as a completely web-based component. Itis developed in AngularJS[1] and a template similar to FUSE[2] template that follows Google's material design specifications. Many different widgets and directives are offered from the VA tool. A wide variety of charts, pies, line charts, tables and time series representation is available in the Visual Analytics tool as the open source Chart.js [3]library is adopted. The Visual Analytics Tool communicates with SFT using MQTT and REST protocol as both of them are supported by the aforementioned tools.

4.4 Decision Support System

Decision Support System should be designed in order to accommodate the needs in a manufacturing environment. DSS integrates Digital Factory Models with events data, and other information and knowledge about the products, manufacturing, planning, simulation, communication and controls at all levels of planning and manufacturing. Raw data from sensors on factories are acquired and transformed according to the Digital Factory Model Schema. Processed data is accessed by the DSS through the DFM API. Accessing and processing the transformed data is easier and DSS implementation can be applied without the complicated need of transforming data in a suitable format. As a deliverable related to DSS is not yet available a brief description of the component is presented below in order to clarify the component's functionality.

Designing a DSS data and algorithm specification should be considered. Data specifications derive form DFM API. The algorithms suitable to be applied on a DSS in a manufacturing environment are both data mining algorithms in order to retrieve suitable data from a repository and decision-making algorithms for the decision-making process. The most used data mining algorithms are classifications trees, generic algorithms, support vector machine, Naïve Bayes. Various combinations and modifications of the above algorithms are considered to designing the data mining part of the DSS. Additionally, Nondeterministic Finite – state Automata (NFA) could be used in the decision-making process. The automata provide the possibility to be expanded during the process. Originally, DSS can implement a rule engine based on Finite State Machines, where the rules applied are defined based on the use cases. The initial rules can be used during a time period to train data and then NFAs and non – deterministic algorithms can be implemented for the decision – making process with the collaboration of the Deep Learning Toolkit.

[1] https://angularjs.org/
[2] http://fusetheme.com/admin-templates/angular
[3] https://www.chartjs.org/

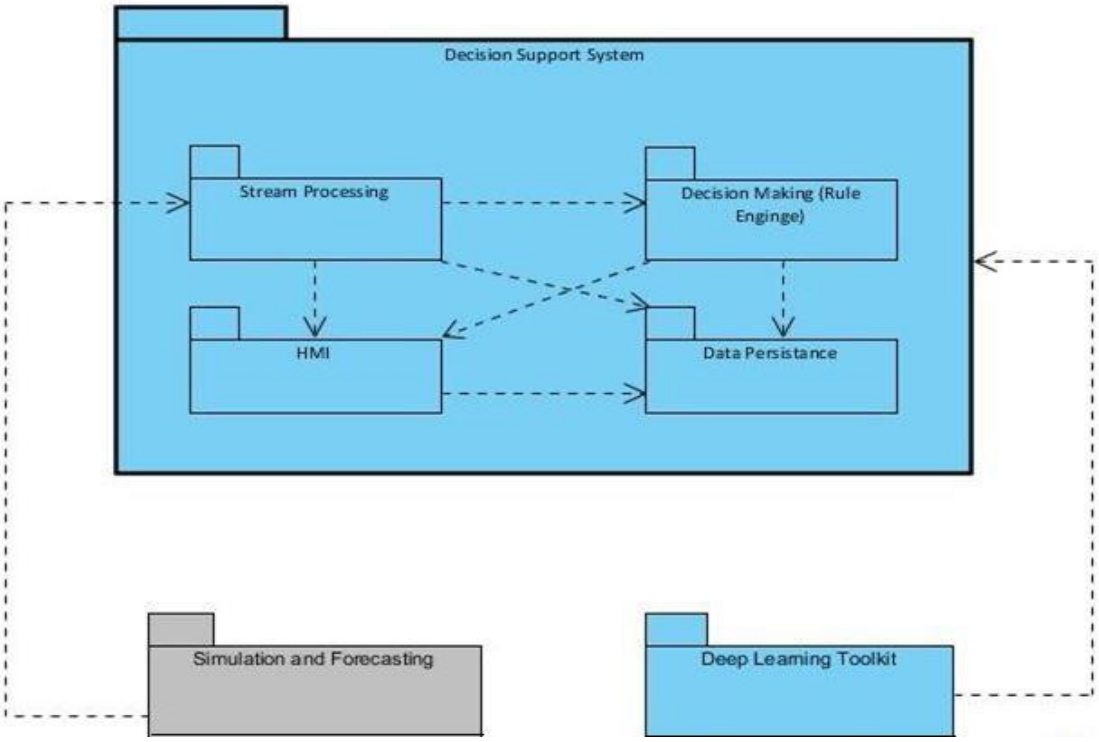


Figure 3: Decision Support System architecture

DSS architecture relates to the use of common data for all systems. Most manufacturing shop floors use the same sensors and they provide the same data. The differences are spotted to the levels of integration at the shop floors with the virtual factory models. Digital Factory Models create a virtual model of the factory, the DSS can work with. The process of making the virtual models of the factories follows at the section below.

Briefly, the sub-components of decision support system are:

- HMI - This component is responsible for the interaction with the user.
- Decision Making - The rule engine of decision support system
- Stream processing - This component processes the data of all external systems and extracts the information necessary for the decision support system
- Data persistence - This sub-component handles the storage of information for decision support internal use.

DSS communicates with the Simulation and Forecasting Toolkit and receives the prediction for the mechanical, electrical and hydraulic faults, as described in UC – KLE-1. SFT is a dockerised application and authentication is provided. The DSS is allowed to access the information from the SFT because it is subscribed as an authenticated application to the SFT docker container.

Furthermore, DSS receives live data from DFM, through an MQTT message broker. The broker is part of the BMS. SFT publishes the live data to topics created for that purpose. Their format contains the sensor id and timestamps. There are actually two different timestamps, one for the actual recorded data and one for the transmission type. There is a small difference in the two times, but it is within acceptable limits. The publishing topics have the following format:

```
Composition/KLE-1/v1/vibrometer
  {
          "id" : "6987bf2e4eb47d825a71",
          "type": "Vibrometer",
          "request":{
          "metadata":{
```

```
"z": "0.153,0.153 ..............,0.112"
 },
" updateState": true,
"eventDate": "2018-07-31T08:06:34Z"
 }
 }
```


DSS receives this live data from MQTT topics which it is subscribed. In these topics, data is transmitted by the BSM from the publishing topics. The topics send the data continuously when it is available, and when there is not available data the answer of the broker is the last available value along with its timestamp. The topics, in which the DSS subscribes have the following format:

`Composition/BMS/NXW_51/OGC/1_0/Datastreams/ds_{ID}/Observations`

Where the IDs are mapped to the different sensors existing in the COMPOSITION ecosystem.

DSS is authenticated in the message broker with Username/Password authentication. Critical live data from shop floors cannot be exposed to unauthorised users. For the COMPOSITION project there are two kinds of users: physical users and applications. DSS is considered and application in the COMPOSITION ecosystem and using the authentication mechanism for both the docker container and the message broker is considered an authenticated user with access to COMPOSITION sources.

4.4.1 Planning and Preparation

Planning virtual factory models should consider the existing architecture of the system described in Figure 3. Data streams transformed from DSS into meaningful indicators and actionable data to provide knowledge. Based on B2MML models and definitions, we created the underlying module and data structures. Also, the incoming data streams, provided by the SFT and the DFM to the DSS, use the same models for data persistence. The same format is applied to both streams based on models. Data uniformity is essential for a project where distributed data is used. The data received from the MQTT topics and the data coming out from the DSS should have the same format and follow the same rules, in order to be easy to use and understandable to both users and system developers. Also, data uniformity allows easier integration with other components, through APIs.

4.4.2 Analytic steps and modelling

In the COMPOSITION project there are several types to be modelled to aid the decision support process for maintenance prediction and process optimization. These are:

- Assets
- Asset Schematics
- Workers
- Workload
- Tasks
- Maintenance procedures
- Stream data from sensors
- Incident models

Based on the data above the models are categorized as follows:

- Context data
- Event data
- Performance indices
- Rules

- Actionable data
- Ingest the data into the data platform

The next step is to bring the relevant data from various sources, either from within or from outside the enterprise, into an analytic environment where it can be processed. The format of the data at source may differ from the format required at the destination. Data transformation may also be needed by the ingestion tool.

In addition to the initial ingestion of data, many intelligent applications are required to refresh the data regularly as part of an ongoing learning process. The learning process readapts data to provide KPIs for the decision – process. The readapted data can be inserted in the rule engine and define the states, transitions and parameters needed to implement a rule. This can be done by setting up a data pipeline or workflow. It is a part of an iterative process that includes rebuilding and re-evaluating the analytical models used by the intelligent application deploying the solution.

4.4.3 System Functionalities and API for the DSS

The next step is to obtain a deeper understanding of the data by investigating its summary statistics, relationships, and by using techniques such as visualization. This is also where issues of data quality and integrity, such as missing values, data type mismatches, and inconsistent data relationships, are handled. Pre-processing transformations are used to clean up the raw data before further analytics and modelling can take place.
Statistics provide the initial relationships between different kinds of types defined above. Also, statistics can be used to know the nominal operational procedures as well as the failure modes and exceptions. Data visualisation provides an easily readable format for the acquired data. Extraction of knowledge is visualised in the decision – making process and graphs are created with valuable information.
Identification of incosistences in data relationships and corrupted data are able to be handled with statistical analysis and visualisation. Outliners are spotted and whether or not they are included in the data set is decided by decision makers, according to pre – defined process in the factories. Outliners provide significant knowledge in the maintenace processes of a factory, because the indicative faults, fault frequencies and causes and they should be processed in order to extract maintenance KPIS.

Table 2: DSS services

Service	***Description***	***Methods***
SyncData	*Synchronizes data*	- *SyncUser* - *SyncTaskWorker* - *SyncAvailableWorker* - *SyncAssets* - *SyncModels* - *AddKPI* - *RemoveKPI* - *GetKPIS* - *AddRule* - *RemoveRule* - *GetRules* - *AddNotification* - *RemoveNotification*

		- *GetNotifications* - *AddNotificationChannel* - *RemoveNotificationChannel* - *GetNotificationChannels*
GetData	*Acquire data*	- *getAssetsOffline* - *getDescriptionFromAssetId* - *getTaskKindOffline* - *getWorkersOffline* - *getAvailableWorkersOffline* - *getTaskDataOffline* - *getCorrectTask* - *getTaskOffline*
NotifyOnAction	*Updates rules related data*	- *NotifyUser* - *NotifyUserOnAction* - *HandleSuccess* - *HandleFailure*

4.5 Result Visualisation and HMI

The Simulation and Forecasting Toolkit uses the same HMI with the DSS for data visualisation. The simulation process is visualised for the UC – KLE-1 in the Predictive Maintenance dashboard. There, the analytics tool of SFT provides the real time representation of the live data received from the vibrometers installed on the KLEEMANN shop floor. Also, the visualisation toolkit provides graphs with the analysis of the live data.

Data analysis consists of eigenvalues of the vibration level. Eigenvalues and eigenvectors were chosen because the vibrometers provide the measurements of the vibration level in the form of acceleration over the three axes: x, y and z.

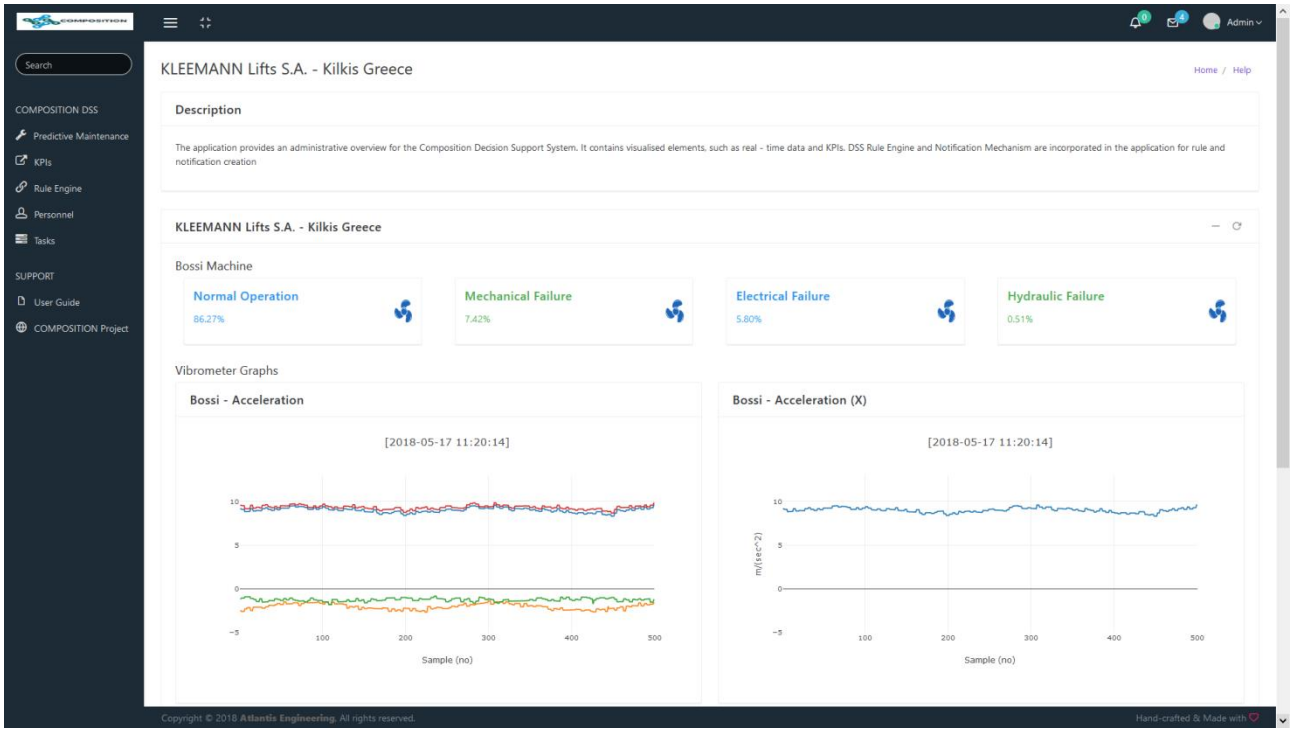


Figure 4: Fault Diagnosis for SFT and DSS HMI

Figure 4 shows the analytics based on the eigenvalues sum in the first panel, the variance of the eigenvalues with a sliding window, which can be defined as shown. The third panel shows the raw data. When a value is above the usual limits of operations it appears in all panels, and with the analysis provided can notify the DSS user for possible problems.

5 Industrial Data Description

In this section a brief description of the data used for each use case that belongs to the area of computational modelling, simulation and prediction of production, according to D2.1 – Industrial Use cases for an Integrated Information Management System is provided.

5.1 UC – KLE 1 Maintenance decision support

This use case focuses on the early detection of machine failure in the BOSSI polishing machine at KLEEMANN's shop-floor. A dataset generated by the Computerised Maintenance Management System (CMMS) is analysed. CMMS maintains a database including information about the company's maintenance operations, such as failure/problem description (mechanical, electrical, hydraulic), duration of breakdown repair, cost of machine breakdown repair, cost of person hours, cost of parts required for repairing etc. This set of data is extracted from CMMS in excel as a report file. 590 breakdowns have been recorded in a period of 10 years (2007-2017). CERTH has already provided a descriptive analysis on the types of faults, actions and changed parts. The probability of the type of the next breakdown to happen has also been calculated by utilizing time series.

A second set of data, will be generated from the sensors that will be installed inside and outside of the BOSSI machine by CERTH. The sensors that will be used in order to capture vibration data are accelerometers. More information about the sensor types can be found in the following chapter.

The data of both datasets generated from CMMS and the installed sensors, will be analysed together to give early indication that a motor inside or outside of the BOSSI machine will face a near future breakdown. This will then be communicated to the maintenance planner and maintenance manager via email or via the COMPOSITION platform.

5.2 UC – KLE 3 Scrap Metal and Recyclable Waste Transportation

This use case focuses on the detection of bin and container fill levels and the calculation of the optimal route for collecting bins inside KLEEMANN's shop-floors. A set of quantity data is generated by the company's ERP system. The ERP maintains a database including information about the produced scrap metal, the paper, wood and plastic wastes. This set of data is extracted from ERP in excel as a report file. In 2016 around 1.000 tons of scrap metal were produced. Also, 7 tons of plastic, 4 tons of wood and 79.5 tons of paper have been recycled. In this use case, a time of flight distance sensor will be used. More information about the sensor types can be found in next chapter.

5.3 UC – BSL 2 Predictive Maintenance

This use case focuses on the early detection of motor failure in the reflow ovens at BSL. In order to detect this, there are three data sets which will be analysed. The first set of data is the machine log files which are generated from the machines themselves. For each blower the machine logs SP (Set point) which is the temperature set by the user, PV (Present value) which is the real temperature measured by the thermometer and OP (Output power) which is the power measured as a percentage where 100% means full voltage is applied and 0% means no voltage is applied. These log files are daily files which are updated every five minutes.
The second set of data is generated from the sensors which have been placed in the reflow oven by Tyndall. These sensors can 'listen' and monitor performance (temperature, vibrations, power consumption) on and near fans (blowers) in reflow ovens. It is intended that the 'signature data' generated from these sensors and the log files from the oven will be analysed together to give early indication that a fan will fail in the near future. This will then be communicated to relevant personnel via email and displayed on large visualization screens in the factory floor.
The third data set are the audio data. Five audio sensors on and near fans in reflow ovens record twenty seconds of audio data and repeat every five minutes. The audio data provide amplitude samples stored as WAV files. From this data the amplitude is calculated in dB using all 20 seconds of the recorded data and store this value. This is stored for each of the five sensors in a single time stamped CSV file.

6 Data Preparation and Processing

This section provides a description of the functions and methodologies developed (new) and utilized or modified (existing ones from scientific literature) until M28 of the project, along with their application on the use cases.

6.1 UC – KLE 1 Maintenance decision support

6.1.1 Methodologies

6.1.1.1 Heatmaps

A **heat map** (or **heatmap**) is a graphical representation of data where the individual values contained in a matrix are represented as colours. The term 'heat map' was originally coined and trademarked by software designer Cormac Kinney in 1991, to describe a 2D display depicting financial market information (US Patent, 1993).Heat maps originated in 2D displays of the values in a data matrix. Larger values were represented by small dark gray or black squares (pixels) and smaller values by lighter squares.

For this use case, we have applied heatmaps to describe the metric of Pearson's correlations among the variables of the dataset.

6.1.1.2 Regression

The earliest form of regression was the method of least squares, which was published by Legendre in 1805 (Legendre, 1805) and by Gauss in 1809 (Gauss, 1809) Legendre and Gauss both applied the method to the problem of determining, from astronomical observations, the orbits of bodies about the Sun (mostly comets, but also later the then newly discovered minor planets). Gauss published a further development of the theory of least squares in 1821 (Gauss, 1821)] including a version of the Gauss–Markov theorem.
Linear and Logistic regressions are usually the first algorithms people learn in predictive modelling. Due to their popularity, a lot of analysts even end up thinking that they are the only form of regressions. The ones who are slightly more involved think that they are the most important amongst all forms of regression analysis. Regression analysis is a form of predictive modelling technique which investigates the relationship between a dependent (target) and independent variable (s) (predictor). This technique is used for forecasting, time series modelling and finding the causal effect relationship between the variables. As mentioned above, regression analysis estimates the relationship between two or more variables. There are multiple benefits of using regression analysis. They are as follows:

1. It indicates the significant relationships between dependent variable and independent variable.
2. It indicates the strength of impact of multiple independent variables on a dependent variable.

Regression analysis also allows us to compare the effects of variables measured on different scales, such as the effect of price changes and the number of promotional activities. These benefits help market researchers / data analysts / data scientists to eliminate and evaluate the best set of variables to be used for building predictive models.
There are various kinds of regression techniques available to make predictions. These techniques are mostly driven by three metrics (number of independent variables, type of dependent variables and shape of regression line). The types of regression most mentioned in scientific and statistics literature are: linear regression, logistic (or logit) regression, polynomial regression, stepwise regression, ridge regression, lasso regression and elastic net regression. Due to the nature of the data of this use case, we had applied only the linear regression.

Linear Regression
It is one of the most widely known modelling techniques. Linear regression is usually among the first few topics which people pick while learning predictive modelling. In this technique, the dependent variable is continuous, independent variable(s) can be continuous or discrete, and nature of regression line is linear. Linear Regression establishes a relationship between dependent variable, denoted here after as Y, and one or more independent variables, denoted hereafter as X, using a best fit straight line, also known as regression line. It is represented by an equation:

$$Y = a + b * X + e$$

where a is intercept, b is slope of the line and e is error term. This equation can be used to predict the value of target variable based on given predictor variable(s). The difference between simple linear regression and multiple linear regression is that, multiple linear regression has (>1) independent variables, whereas simple linear regression has only 1 independent variable.

Logistic Regression
Logistic regression is used to find the probability of event = Success and event = Failure. We should use logistic regression when the dependent variable is binary (0/ 1, True/ False, Yes/ No) in nature. Logistic regression is widely used for classification problem and doesn't require linear relationship between dependent and independent variables. It can handle various types of relationships because it applies a non-linear log transformation to the predicted odds ratio.

Polynomial Regression
A regression equation is a polynomial regression equation if the power of independent variable is more than 1. The equation below represents a polynomial equation:

$$Y = a + b * X\text{^}2$$

In this regression technique, the best fit line is not a straight line. It is rather a curve that fits into the data points.

Stepwise Regression
This form of regression is used when we deal with multiple independent variables. In this technique, the selection of independent variables is done with the help of an automatic process, which involves *no* human intervention. This feat is achieved by observing statistical values like R-square, t-stats and AIC metric to discern significant variables. Stepwise regression basically fits the regression model by adding/dropping co-variates one at a time based on a specified criterion. Some of the most commonly used Stepwise regression methods are listed below:

- Standard stepwise regression does two things. It adds and removes predictors as needed for each step.
- Forward selection starts with most significant predictor in the model and adds variable for each step.
- Backward elimination starts with all predictors in the model and removes the least significant variable for each step.

The aim of this modelling technique is to maximize the prediction power with a minimum number of predictor variables. It is one of the methods to handle higher dimensionality of data set.

Ridge Regression
Ridge Regression is a technique used when the data suffers from multicollinearity (independent variables are highly correlated). In multicollinearity, even though the least squares estimates (OLS) are unbiased their variances are large which deviates the observed value far from the true value. By adding a degree of bias to the regression estimates, ridge regression reduces the standard errors.

Lasso regression
Similar to Ridge Regression, Lasso (Least Absolute Shrinkage and Selection Operator) also penalizes the absolute size of the regression coefficients. In addition, it is capable of reducing the variability and improving the accuracy of linear regression models. Lasso regression differs from ridge regression in a way that it uses absolute values in the penalty function, instead of squares. This leads to penalizing (or equivalently constraining the sum of the absolute values of the estimates) values which causes some of the parameter estimates to turn out exactly zero. Larger the penalty applied, further the estimates get shrunk towards absolute zero.

ElasticNet Regression
ElasticNet is a hybrid of Lasso and Ridge Regression techniques. It is trained with L1 and L2 prior as regularizer. Elastic-net is useful when there are multiple features which are correlated.

Due to the nature of the data of this use case, the most suitable type of regression is the linear regression.

6.1.1.3 Probability Theory

Probability theory is the branch of mathematics concerned with probability. Although there are several different probability interpretations, probability theory treats the concept in a rigorous mathematical manner by expressing it through a set of axioms. Typically, these axioms formalise probability in terms of a probability space, which assigns a measure taking values between 0 and 1, termed the probability measure, to a set of outcomes called the sample space. Any specified subset of these outcomes is called an event. Central subjects in probability theory include discrete and continuous random variables, probability distributions, and stochastic processes, which provide mathematical abstractions of non-deterministic or uncertain processes or measured quantities that may either be single occurrences or evolve over time in a random fashion. Although it is not possible to perfectly predict random events, much can be said about their behaviour. Two major results in probability theory describing such behaviour are the law of large numbers and the central limit theorem.

For this use case, we have developed an approach where the probabilities of an upcoming event are calculated based on scenarios prior to that event.

6.1.2 Application

In this section the application of correlation heatmap, linear regression and probability theory methodologies are provided and briefly described below.

6.1.2.1 Heatmap - Linear Regression

The variables of the dataset's use case are: machine fault type (PROBLEM DESCRIPTION), duration of problem solution (in hrs) (DURATION), the actual person hours (PERSON HOURS), the actual cost of person hours (COST OF PERSON HOURS), the actual cost of parts to be replaced (COST OF PARTS), the type of parts to be replaced (PARTS) and the action taken by the personnel to solve the problem (ACTION). A correlation heatmap of all variables mentioned above is given in Figure 5.

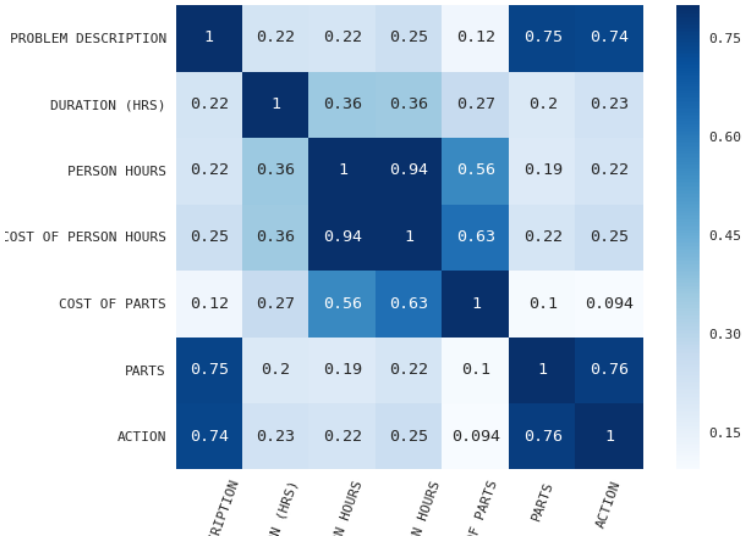


Figure 5: Correlation heatmap of all use case dataset's variables.

The heatmap on Figure 5, point to the fact that there is strong positive correlation between the machine fault type, the parts and the action taken by personnel so as to solve the problem. The actual problems of the industrial machine are three: electronic, hydraulic and mechanical. Figure 6, provides the distribution of machine fault types in a time range of 10 years. One can see that the most common fault types are electrical and mechanical ones.

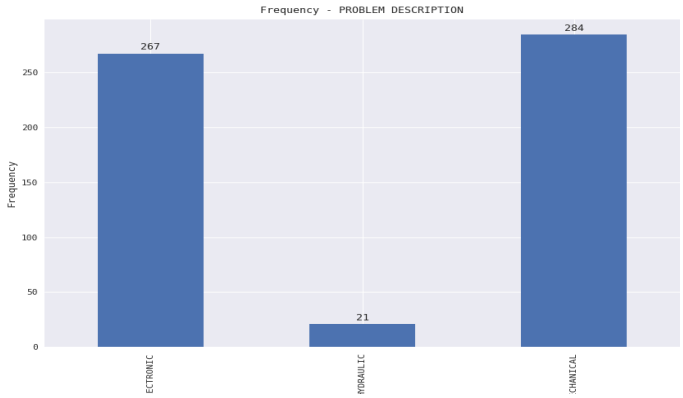
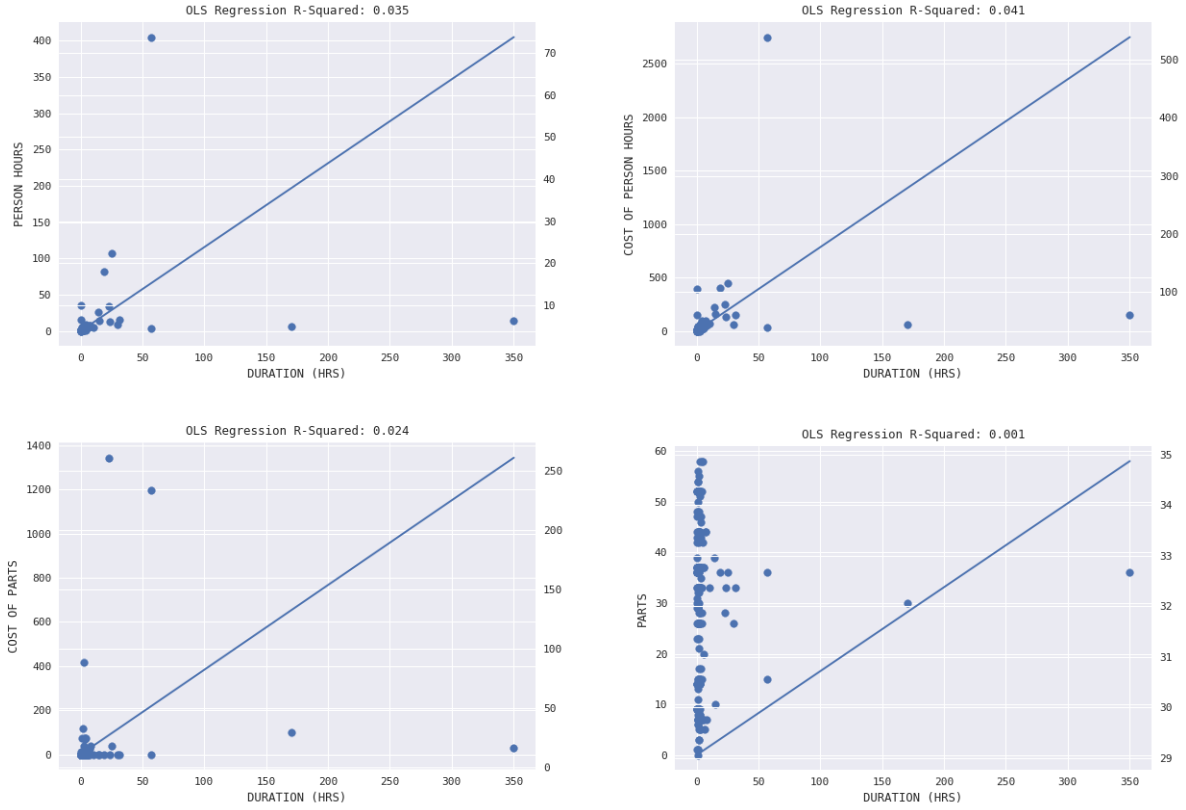
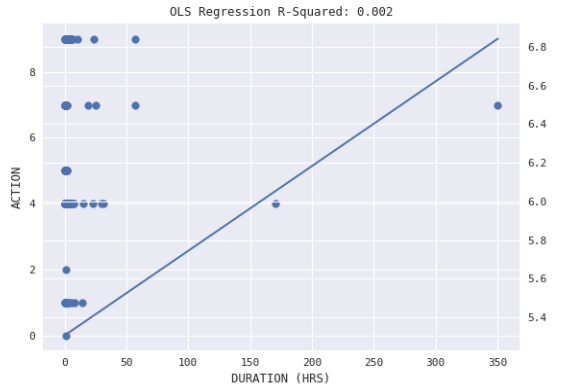
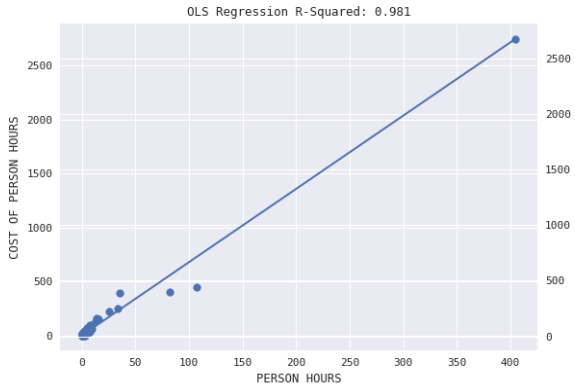


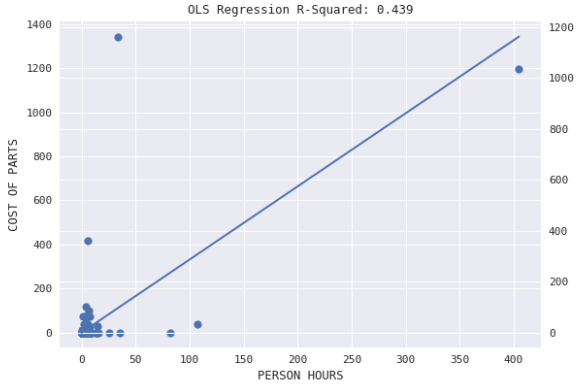
Figure 6: Frequency of machine fault types in a time range of ten years

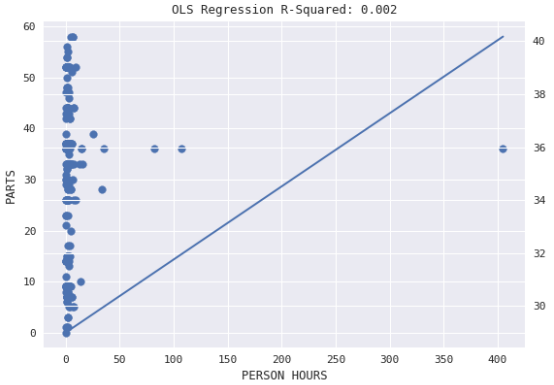
In Figure 7, the linear regression model between different variables of the dataset is provided. The variables used for this task are DURATION, PERSON HOURS, COST OF PERSON HOURS, COST OF PARTS, PARTS and ACTION. For the validation of the models, the metric of coefficient of determination, denoted R^2 or r^2 and pronounced "R squared", is used. The coefficient of determination is the proportion of the variance in the dependent variable that is predictable from the independent variable(s). The coefficient of determination ranges from 0 to 1, where the closer to 1 the better the fit of the linear model on the data.

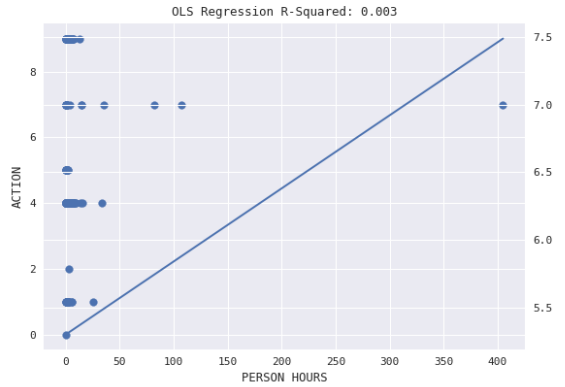


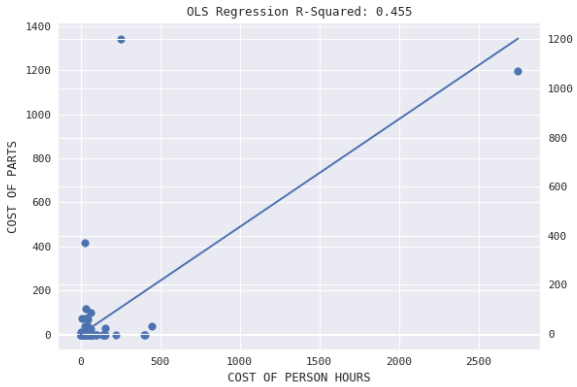

OLS Regression R-Squared: 0.002
ACTION
DURATION (HRS)

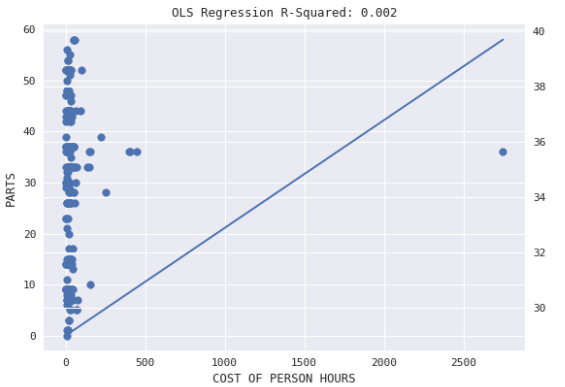

OLS Regression R-Squared: 0.981
COST OF PERSON HOURS
PERSON HOURS

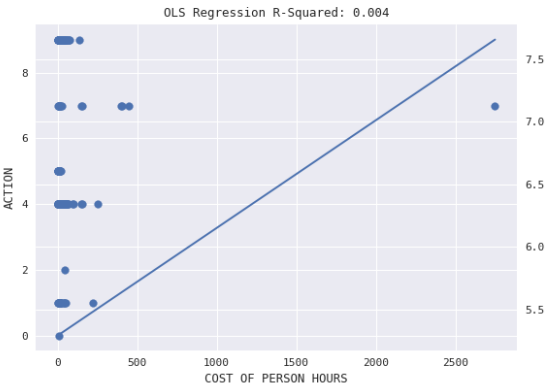

OLS Regression R-Squared: 0.439
COST OF PARTS
PERSON HOURS

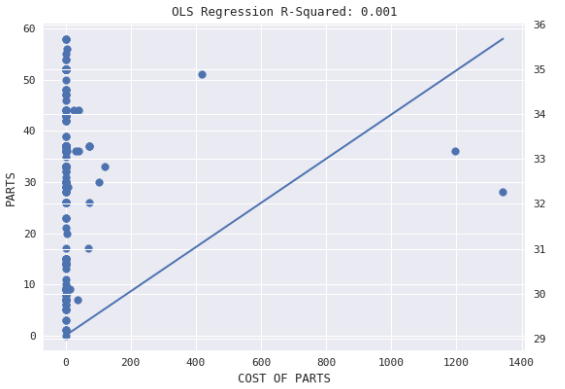

OLS Regression R-Squared: 0.002
PARTS
PERSON HOURS

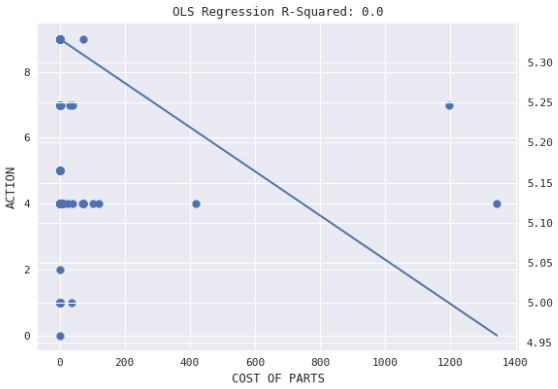

OLS Regression R-Squared: 0.003
ACTION
PERSON HOURS


OLS Regression R-Squared: 0.455
COST OF PARTS
COST OF PERSON HOURS


OLS Regression R-Squared: 0.002
PARTS
COST OF PERSON HOURS


OLS Regression R-Squared: 0.004
ACTION
COST OF PERSON HOURS





Model	Variable 1	Variable 2	R-squared
1	PERSON HOURS	DURATION	0.035
2	COST OF PEARSON HOURS	DURATION	0.041
3	COST OF PARTS	DURATION	0.024
4	PARTS	DURATION	0.001
5	ACTION	DURATION	0.002
6	COST OF PEARSON HOURS	PERSON HOURS	0.981
7	COST OF PARTS	PERSON HOURS	0.439
8	PARTS	PERSON HOURS	0.002
9	ACTION	PERSON HOURS	0.003
10	COST OF PARTS	COST OF PERSON HOURS	0.455
11	PARTS	COST OF PERSON HOURS	0.002
12	ACTION	COST OF PERSON HOURS	0.004
13	PARTS	COST OF PARTS	0.001
14	ACTION	COST OF PARTS	0.000
15	ACTION	PARTS	0.000

Figure 7: Linear regression models among different variables of use case dataset. R-squared metric for each linear regression model.

From the calculation of r-squared metric for the 15 linear regression models, the best model is found for the variables COST OF PERSON HOURS and PERSON HOURS (0.981) and the models between COST OF PARTS and PERSON HOURS (0.439) and COST OF PARTS and COST OF PERSON HOURS (0.439) are the second best. The overall outcome is that the approach of linear regression for prediction of future events does not provide reliable predictions and outcomes.

6.1.2.2 Probability model

For the calculation of the probabilities of an upcoming event or machine fault (no fault, electrical, hydraulic and mechanical) per day we provide a visual analytics approach where the probabilities are calculated based on scenarios prior to that fault. The concept behind this analytic is to provide reliable calculations based on four scenarios: scenario 1 – provide the probability of an event to happen (no fault, electrical, hydraulic and mechanical) after one (1) day of no-fault, scenario 2 – provide the probability of an event to happen (no fault, electrical, hydraulic and mechanical) after two (2) days of no-fault, scenario 3 – provide the probability of an event to happen (no fault, electrical, hydraulic and mechanical) after five (5) days of no-fault, scenario 4 – provide the probability of an event to happen (no fault, electrical, hydraulic and mechanical) after ten (10) days of no-fault. For example, in Figure 8(a), for scenario 3 (event after 5 days of no fault) the probability of no-fault is 0.86, the probability of an electrical fault is 0.12, the probability of a hydraulic fault is 0.0 and the probability of a mechanical fault is 0.02, while for scenario 4 (event after 10 days of no fault) the probability of no-fault is 0.85, the probability of an electrical fault is 0.12, the probability of a hydraulic fault is 0.0 and the probability of a mechanical fault is 0.04. After 220 time moments (days) the probabilities are: for scenario 3 (event after 5 days of no fault) the probability of no-fault is 0.79 the probability of an electrical fault is 0.09, the probability of a hydraulic fault is 0.03 and the probability of a mechanical fault is 0.1, while for scenario 4 (event after 10 days of no fault) the probability of no-fault is 0.82 the probability of an electrical fault is 0.11, the probability of a hydraulic fault is 0.03 and the probability of a mechanical fault is 0.05 (see Figure 8(b)). After 280 time moments (days) the probabilities are: for scenario 3 (event after 5 days of no fault) the probability of no-fault is 0.79 the probability of an electrical fault is 0.1, the probability of a hydraulic fault is 0.02 and the probability of a mechanical fault is 0.09, while for scenario 4 (event after 10 days of no fault) the probability of no-fault is 0.79 the probability of an electrical fault is 0.14, the probability of a hydraulic fault is 0.02 and the probability of

a mechanical fault is 0.05 (see Figure 8(c)). The initialization length (parameter) for this visual analytic is set at 100 days and the is functional for both historical data or in real time.

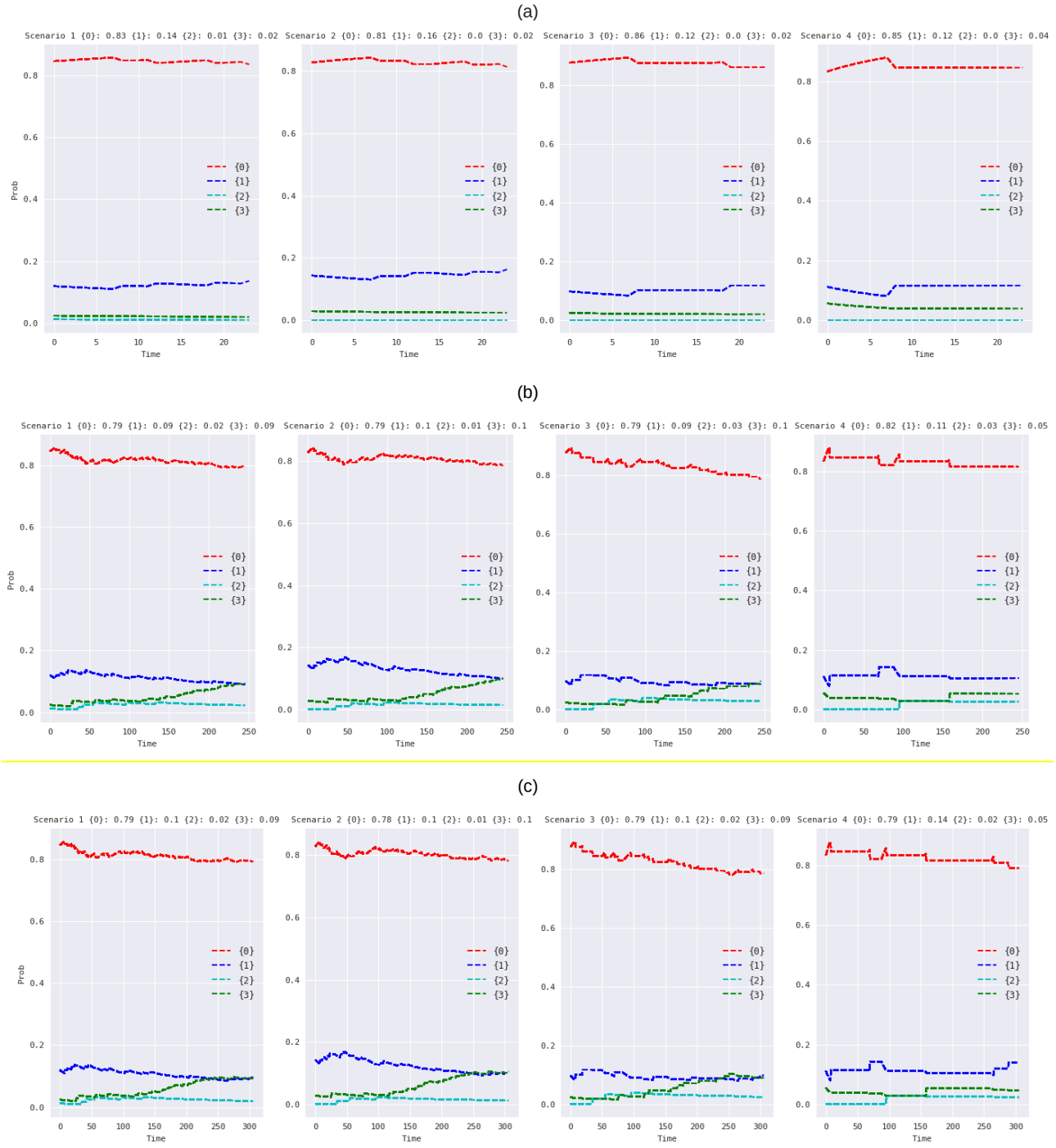


Figure 8: Visualizations of various moments of the probability visual analytic.

6.1.3 Vibration Sensor Deployment

The previous described approach is based on historical data which have been provided by KLEEMANN and they are related to previous faults of the Bossi machine. However, as the Bossi machine has no built-in sensors it was unable to provide live data for further analysis in order to enhance the efficiency of predictions. After a visit at KLEEMANN's shop-floor and discussions with its technicians about the Bossi and its performance before it breaks, the installation of a vibration sensor was considered as the most promising approach in order to get useful data for the prediction of future failures.

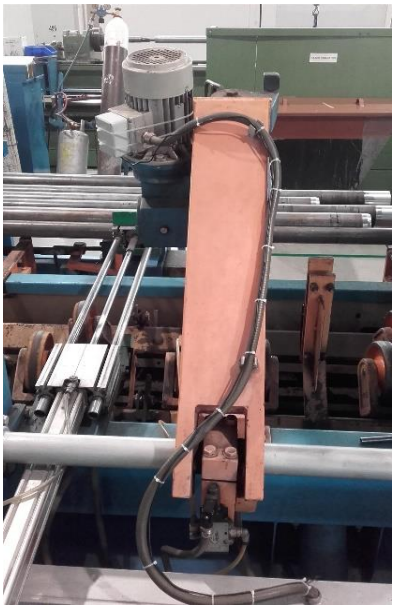


Figure 9: Vibration sensor position at Bossi machine

Primary data for Vibration are coming from LIS3DH 3-axis MEMs accelerometer controlled by ESP-32 SoC with integrated WiFi, mounted on a Bossi Motor. The data are samplings of 1344 samples of accelerations from three axis(x, y, z) at a sampling rate of 1.344 kHz. Restrictions on the ESP32 memory confine us to the number of samples. The sensor is powered from the Bossi machine's power supply so it is ON only when the Bossi apparatus is ON. Furthermore, the sensor is only sending data when there are vibrations present above a programmed threshold to ensure that data are sent only when the motor is operating. Samplings are sent in average approximately every 1.5 seconds, in proper motor operation, depending highly on network conditions. Due to WiFi buffer restrictions, every sampling is split into three packets, each corresponding to an axis. The packets are sent utilizing the popular M2M protocol MQTT. Those packets contain raw sensor values and are translated in the back-end to accelerations measured at m/s2 before being further analysed.

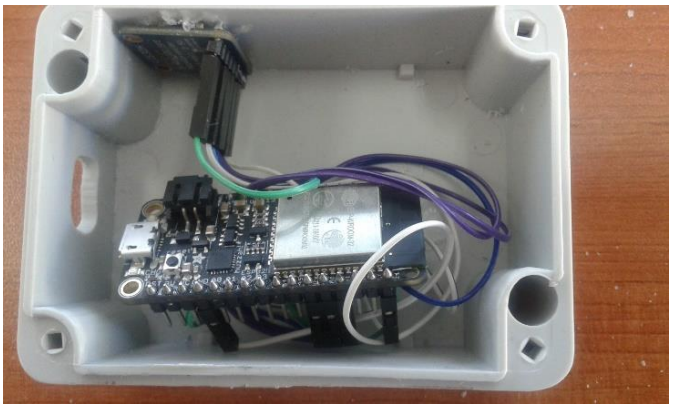


Figure 10: Vibration Sensor

6.1.4 Vibration Sensor Data Analysis

The Machine Vibration Diagnosis Profile (MVDP) method aims to estimate the time point(s) *T* when abnormal vibrations occur from the profile of the eigenvalues summation , where i is the time of the recording, and the calculated variance in a sliding window w of fixed size and step one, calculated from the vibration sensor recordings, described in Section A. The basic assumption of MVDP is that significant eigenvalue sums with simultaneous significant variations could point out to abnormal vibrations. It should be noted that the analysis of data from vibration sensor take place per recording. The vibration sensor provides recordings in a 3-dimensional coordinate system at $1{,}344KHz$ in a time interval of 1.5 seconds (on average), meaning that the initial data matrix (*D*) of a single recording is $=$, where $i = 1, 2, 3$ and $j = 1, \dots, 1344$. The Euclidean distance (*ED*) between each one of the 1344 points and the 3-dimensional coordinate system origin is calculated and

the = , where $i = 1, \ldots, 1344$, vector norm is created. After, the pairwise distance (*PD*) matrix in 1344-dimensional space is calculated = where i and *j* ranges from 1 to 1344. The formation kernel of *PD* matrix is again Euclidean distance, but other kernels/options, such as Chebyshev, Cosine, Dice, Jaccard, Kulsinski, Mahalanobis, Minkowski and Seuclidean distances, could be tested in the future. Consequently, principal component analysis (PCA) is applied on *PD*. Thus, from the eigenvalue (E) vector = , where $i = 1, \ldots, 1344$, the sum of top-3 greatest values denote . Again, the number selected eigenvalues for summation could be also a point for research. Moreover, when) = w the also starts to calculated.

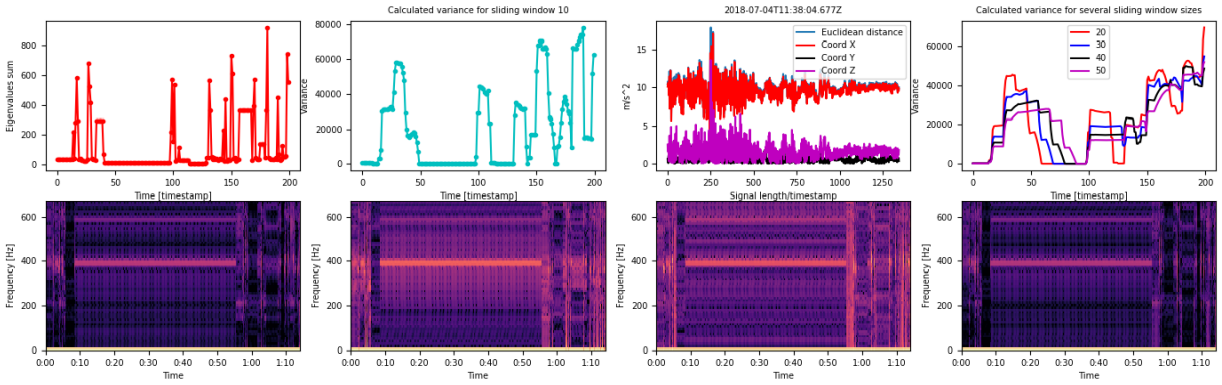


Figure 11: Eigenvalue sum, variance$_{w=10}$, raw signal, variance$_{w=20,30,40,50}$ and spectrograms of coordinates x, y, z and their ED for typical machine vibrations.

In Figure 11 and Figure 12 on the first line, the plots of eigenvalue sums, variance for $w = 10$, raw signals (coordinates x, y, z and their Euclidean distance from origin) and variances for $w = 20,30,40$ and 50 (for comparison) are given. On the second line, the spectrograms of coordinates x, y, z and their Euclidean distance from origin are provided, as a visualization measure that "connects" the existence of abnormal vibrations with specific frequencies. Figure 11 visualizes eigenvalue sums and calculated variance when typical vibrations exist, while Figure 12 visualizes a time point when a major eigenvalue sum exist. The difference between Figure 11 and Figure 12 are obvious.

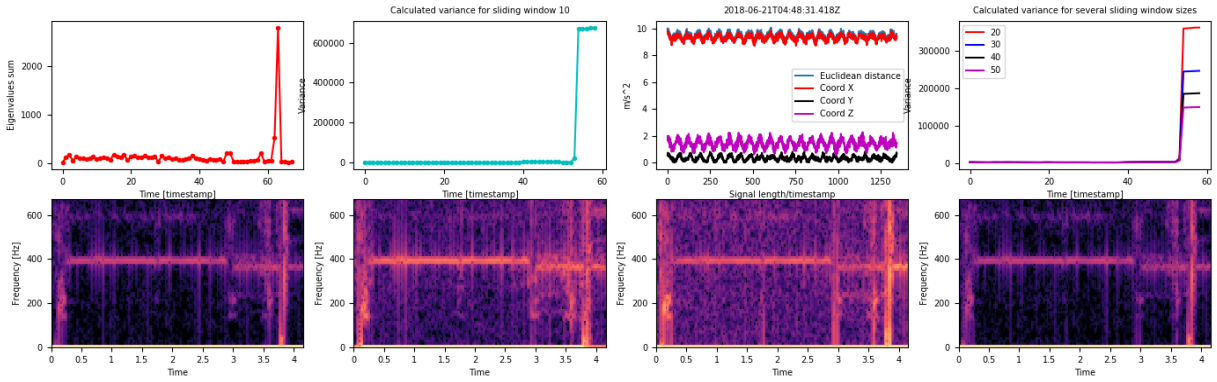


Figure 12: Eigenvalue sum, variance$_{w=10}$, raw signal, variance$_{w=20,30,40,50}$ and spectrograms of coordinates x, y, z and their ED for abnormal machine vibrations.

The thresholds for and time series that point out to an abnormal situation is key-point for discussion and the calibration of MVDP methodology for a specific machine is based on its vibrations/behaviour during operation. By that, repeated patterns that point to normal vibrations/behaviour must be identified firstly. For the machine in KLEEMANN pilot plant, where the vibration sensor is installed, the abnormal vibration threshold for is set at $EST = 1500$ and for $VT \geq \approx 172000$, where $E(\mu)$ is the average of window w means μ during a hole shift. This work is ongoing and we aim to introduce and design a real-time dynamic threshold so as to identify different situations clearly and avoid confusions regarding the output of the sensor used.

6.2 UC – KLE 3 Scrap Metal and Recyclable Waste Transportation

6.2.1 Fill Level Sensor Deployment

The UC – KLE 3 Scrap Metal and Recyclable Waste Transportation is triggered by a full bin in KLEEMANN's production. In order for the Prediction engine to be able to estimate the fill level in the future and propose to a worker the optimal path in order to transport the wastes to a central bin outside of the production line, a fill level sensor for the internal bins has been designed and developed.

For the fill level sensor data, the VL53L0X micro-Lidar Time of flight Sensor was used to capture raw measurements of distance of the waste heap from the deployment point. The STM32L053c8t6 low power microcontroller controls the sensor and the communication. The microcontroller-sensor communication is carried out through an I2C bus that can serve multiple sensors. The microcontroller also checks for faulty measurements, repeats the measurement process again, and if the measurement is faulty sends an error flag. The wireless communication is carried out via the sx1272mbas LoRa Module for STM.

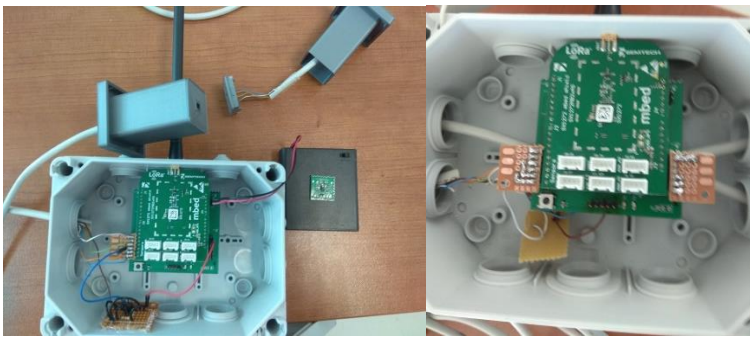


Figure 13: Fill level sensor for (a) single (b) multiple bins

Data is transferred via the LoRa low power protocol to the LoRa Gateway. The gateway to be used is the LoRank 8. It has to be noted that LoRa protocol allows transmission of only very small packets of data, so this means that only raw measurements of distance are transferred along with an error value and the measurement of battery level. The gateway that is connected to the internet via Ethernet or Wi-Fi publishes those data on an MQTT topic on the cloud. A listener, connected to the same broker as the gateway, reprocesses the data to convert the raw distance measurement into a fill percentage, assigns the ids for each bin and derives the json object format to be sent on the destination platform.

Five prototypes were installed at KLEEMANN. Three prototypes are monitoring twelve recycle bins (each prototype monitors four bins) and two prototypes are monitoring two scrap metal bins. The prototypes are mounted on custom-made metal bases as seen in the figures below.

Figure 14: Fill level sensor for (a) scrap metal bin (b) recycle bins

6.2.2 Fill Level Sensor Data Analysis

The Slope Statistic Profile (SSP) method aims to estimate the change point *T* from the profile of a linear trend test statistic, computed on consecutive overlapping time windows along the time series. The test statistic for linear trend estimation is selected based on its high power compared to other test statistics for both correlated and white noise residuals. In the SSP approach, a first candidate change point T is the time point at which the profile crosses the threshold line of rejection of the null hypothesis of no trend at $\pm$, where *a* is the significance level, *w* is the size of the sliding window and *t* follows the Student distribution with *w-2* degrees of freedom. The search of change point T is confined in a time interval corresponding to the profile segment bounded by and for positive trends and by and for negative trends, where significance levels and for two side test are 0.20 (or 20%) and 0.05 (or 5%), respectively. The t-statistic for the parametric linear trend test that is used in SSP approach is $t = \sim$, where is least square estimator for the trend parameter and *s* is the estimated standard error of . The null hypothesis of no trend is rejected at a significance level a if $\geq$. The selection of two significant levels is based on the assumption that there are not sudden changes in natural variations, which means that some time is needed in order a time series, which has a not obvious structural break to pass from no linear trend to linear trend condition. Thus, the existence of two significant levels describes the transition between these conditions. Thus, , will be denoted as *Upper₁* and *Upper₂* thresholds, respectively and – , will be denoted as *Lower₁* and *Lower₂* thresholds, respectively. The SSP method is initially created in order to detect significant changes in linear trend of known time series. The version of the SSP presented here detects all possible changes in linear trend of a time series in real time.

In the following, the parametric linear trend test for a sliding window of size w on the time series $\mathrm{Y_t}$, $\mathrm{t} = 1, \ldots, \mathrm{n}$, is presented. Thus, for the first window $[\mathrm{Y_1}, \ldots, \mathrm{Y_w}]^{\mathrm{T}}$ the least square estimator for the trend parameter β is obtained as $\hat{\beta} = \sum_{t-1}^{w}(t-\bar{t})Y_t / \sum_{t-1}^{w}(t-\bar{t})^2$ (1), where $\bar{t}$ is the average time. The standard error of $\hat{\beta}$ is estimated from the power spectrum $s_1(\hat{\beta}) = \left[2\int_0^{0.5} W(f)S(f)\right]^{1/2}$ (2). In (2), $W(f) = \left|\sum_{t=1}^{w} b_t e^{-2\pi i f}\right|^2$ with $b_t = \frac{t-\bar{t}}{\sum_{t=1}^{w}(t-\bar{t})^2}$ and $S(f)$ denotes the sample power spectrum of ε_t given as $S(f) = \frac{1}{2\pi}(\hat{\gamma}_0 + 2\sum_{\kappa=1}^{w-1}\hat{\gamma}_\kappa \cos(2\pi f k)$. Parameter $\hat{\gamma}_\kappa$ denotes the estimate of the kth order autocovariance of ε_t, given as $\hat{\gamma}_\kappa = \frac{1}{w}\sum_{t=1}^{w-k}\hat{\varepsilon}_{t+k}\hat{\varepsilon}_t$ for $k > 0$, where $\hat{\varepsilon}_t = Y_t - \hat{a} - \hat{\beta}t$ are the estimated residuals ($\hat{a} = \overline{Y}_t - \hat{\beta}t$ and $\overline{Y}_t$ is the mean of the time series), and $\hat{\gamma}_0 = \frac{1}{w-2}\sum_{t=1}^{w}\hat{\varepsilon}_t^{\,2}$ for $k = 0$.

(a) (b)

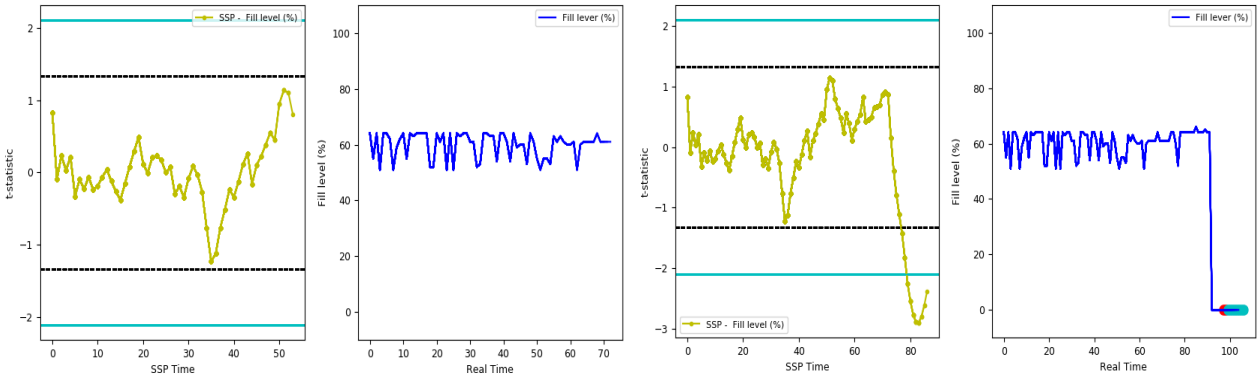


Figure 15: Linear trend profile (left subplot) and fill level time series (right subplot) (a) before and (b) during the detection of changes in linear trend. The cyan and dotted lines in negative denote the Lower₁ and Lower₂ thresholds, respectively, and the cyan and dotted lines in negative denote the Upper₁ and Upper₂ thresholds, respectively.

SSP method is applied on the time series of recordings (percentages) of fill level sensor described in Section 6.2.1. The interval between two recordings is 5 seconds (on average). The selected size of sliding window for SSP is 20 elements, based on intensive simulations and experiments. Figure 15(a) shows the linear trend profile (left subplot) and the fill level time series (right subplot). By this time, when the linear trend of fill level time series is steady, the linear trend profile points out that there are not significant changes indeed. When the fill level drops from 60% to 0%, the profile of linear trend almost immediately responds to that change and provides a notification (see Figure 15(b), right subplot). In general, the real time approach of SSP method has the ability to detect multiple change points in linear trend of a time series and it is most effective in cases when the change(s) in a time series is(are) not abrupt, as happens in most natural variations. Moreover, has great performance in time series from industrial sector.

6.2.3 Recyclable Waste Transportation Application

A common way to mathematically model and represent road networks, in order to deal with problems such as the shortest path problem, is graphs G that are composed by sets of nodes N and sets of edges E. In graph theory, the shortest path problem is the problem of finding a path between two nodes in a graph such that the sum of the weights of its constituent edges is minimized. There are several works trying to solve this problem, the well-known Dijkstra's algorithm solves the single-source shortest path problem in $O(V^2)$ (worst case computational complexity), while there are various implementations of Dijkstra's algorithm that reduce the computational cost (Ahuja, 1993; Cherkassky,1996; Zhan, 1998).

An extension of the Dijkstra's algorithm is the A* search algorithm (Hart, 1968) which achieves better performance by using heuristics to guide its search. Moreover the Bellman–Ford algorithm (Kenneth, 2003) solves the single-source problem if edge weights may be negative. There are, also, the Floyd–Warshall (Bang-Jensen, 2000) algorithm which solves all pairs' shortest paths and the Johnson's algorithm which solves the same problem, and may be faster than Floyd–Warshall on sparse graphs. Additional algorithms and associated evaluations may be found in Cherkassky, Goldberg & Radzik (Cherkassky, 1996).

Dijkstra's algorithm is decided to be used in the specified use case. As mentioned above Dijkstra's algorithm find a shortest path tree from a single source node, by building a set of nodes that have the minimum distance from the source. The source node graph has the following:

- vertices, or nodes, denoted in the algorithm by u or v
- weighted edges that connect two nodes: (u, v) denotes an edge and $w(u, v)$ denotes its weight.

The pseudocode of Dijkstra's algorithm is given below:

$dist[s] \leftarrow O$
for all $v \in V - \{s\}$
    do $dist[v] \leftarrow \infty$
$S \leftarrow \emptyset$
$Q \leftarrow V$
while $Q \neq \emptyset$
do $u \leftarrow$ mindistance$(Q, dist)$
    $S \leftarrow S \cup \{u\}$
    for all $v \in neighbors[u]$
        do if $dist[v] > dist[u] + w(u, v)$
            then $dist[v] \leftarrow dist[u] + w(u, v)$
return $dist$

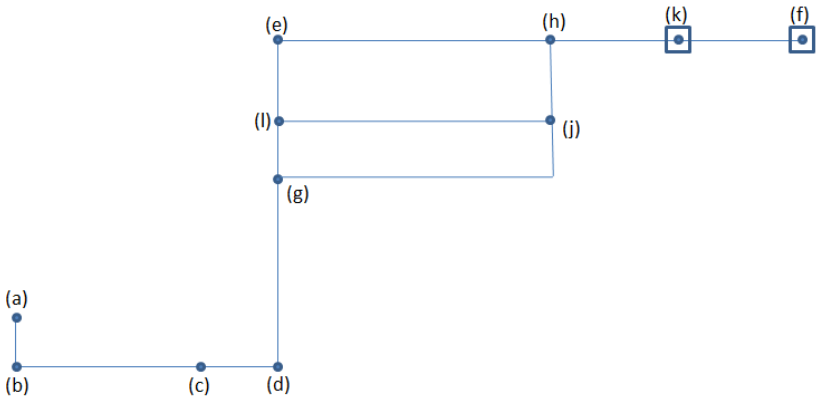


Figure 16: Source node of KLEEMANN plant site.

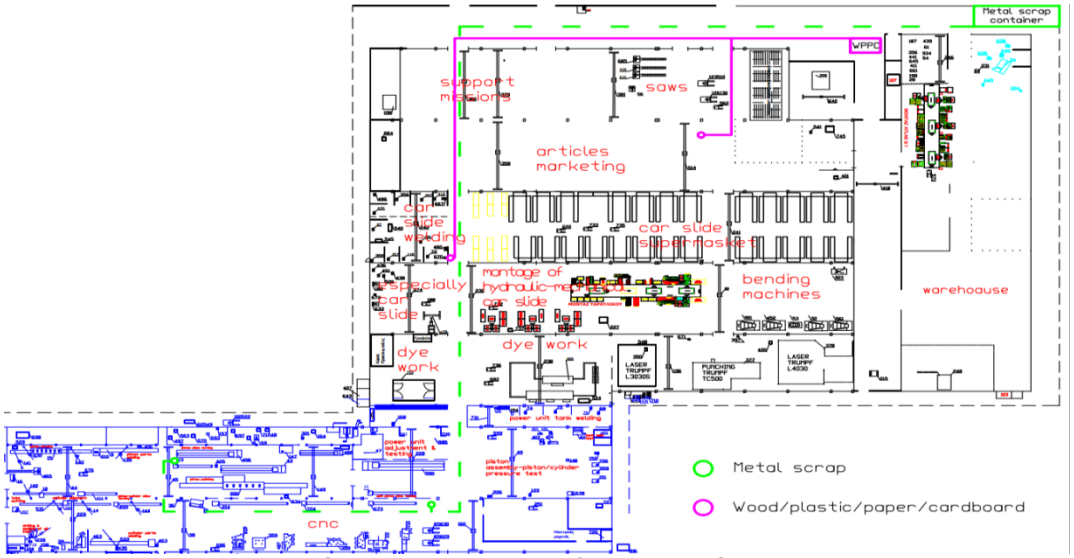


Figure 17: KLEEMANN factory top view.

The main concern of the Dijkstra's algorithm is to visit all nodes in the graph and to find the smallest distance to each node. Thus, $dist$ contains the shortest path tree from the source node. For the recyclable waste transportation problem in KLEEMANN plant site, the source node with waste bins and scrap containers based on the top view of the factory (see Figure 17 and Figure 16, respectively) are given below:

The waste bins are placed in nodes (a), (c), (g) and (j), while the scrap containers are placed in nodes (k) and (f). The actual distances (in meters) between the nodes are given in the table below:

Table 3: Distance matrix (in meters) between nodes.

	a	b	c	d	e	f	g	h	j	k	l
a	0	20	0	0	0	0	0	0	0	0	0
b	20	0	60	80	0	0	0	0	0	0	0
c	0	60	0	20	0	0	0	0	0	0	0
d	0	80	20	0	140	0	70	0	0	0	0
e	0	0	0	140	0	120	70	60	0	100	0
f	0	0	0	0	120	0	0	60	0	20	0
g	0	0	0	70	70	0	0	0	90	0	0
h	0	0	0	0	60	60	0	0	40	40	0
j	0	0	0	0	0	0	90	40	0	0	60
k	0	0	0	0	100	20	0	40	0	0	0
l	0	0	0	0	0	0	0	0	60	0	0

The distance matrix is the main input of the Dijkstra's algorithm, and thus it has to be very precise and accurate. The measurements in KLEEMANN plant site took place with the use of GPS. After the application of proposed algorithm on distance matrix the optimal paths from waste bins to scrap containers are given in Table 4 below:

Table 4: Optimal path from point A to point B

Point A	Point B	Route
a	f	[a, b, d, e, f]
c	f	[c, d, e, f]
g	k	[g, e, k]
j	k	[j, h, k]

These paths from point A to point B will be integrated with the fill level sensor (see section 6.2.2) so as when the fill level sensor algorithm raise an alarm, along with the notification to provide also the optimal path for the scrap container.

6.3 UC – BSL 2 Predictive Maintenance

6.3.1 Methodologies

6.3.1.1 Heatmaps

A **heat map** (or **heatmap**) is a graphical representation of data where the individual values contained in a matrix are represented as colours. The term 'heat map' was originally coined and trademarked by software designer Cormac Kinney in 1991, to describe a 2D display depicting financial market information (US Patent, 1993). Heat maps originated in 2D displays of the values in a data matrix. Larger values were represented by small dark grey or black squares (pixels) and smaller values by lighter squares.

For this use case, we have applied heatmaps to describe the metric of Pearson's correlations among the variables of the dataset.

6.3.1.2 Outlier Detection

According to Hawkins definition (Hawkins, 1980), "An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism". Considering that normal data objects are spawned from a particular mechanism, the abnormal ones deviate drastically from this mechanism. An outlier may cause simply by chance, but it may also indicate measurement error or that the given data set has a heavy-tailed distribution. The outlier detection is the process of identifying observations which raise suspicions by differing significantly from the majority of the data. Out of the of outlier detection techniques, next are described Median Absolute Deviation (MAD), Local Outlier Factor (LOF) and Density-based spatial clustering of applications with noise (DBSCAN).

Median Absolute Deviation (MAD)
The median absolute deviation (MAD) is a robust measure of how spread a dataset is. It is a distance-based technique that can be used in temporal data. MAD is a robust estimator of scale and works better with distributions without a mean or variance. The median absolute deviation is defined as

$$MAD = \left|x_i - M_j(x_j)\right|,$$

where x_i is one or a set of data points to be evaluated and $M_j(x_j)$ is the median of the whole set of points, for normally distributed data sets.

The evaluation criterion is a modified z-score, a standardized score that measures outlier strength by estimating the difference of the score from the median. The modified z-score is calculated by multiplying MAD values by a constant to approximate the standard deviation. The calculation is defined by the following equation:

$$\frac{x_{i-M}}{1.4826 * MAD} > |\pm threshold|$$

Local Outlier Factor (LOF)
Local Outlier Factor (LOF) is a density based outlier detection technique. It provides a factor (LOF) of how close is a data point to its neighbors in respect to its neighbor being also close to it. LOF can be calculated based on one or more signal distances or extracted features. The outliers are compared to their local neighbours, instead of the global data distribution.

The density around an outlier object substantially differs from the density around the outlier's neighbours. The degree of the object being an outlier is indicated by the relative density of an object against its neighbours.

The distance between an object o and its k-th nearest neighbour (k-distance) is represented as $dist_k(o)$. Given the k-distance of o, the k-distance neighbourhood of o contains every object whose distance from o is not greater than the k-distance, represented as $N_k(o)$. The reachability distance from o' to o is:

$$reachdist_k\left(o \leftarrow o'\right) = \max\{dist_k(o), dist(o, o')\}$$

The local reachability density of o is:

$$lrd_k(o) = \frac{\|N_k(o)\|}{\sum_{o' \in N_k(o)} reachdist_k\left(o \leftarrow o'\right)}$$

The LOF of an object o is the average of the ratio of local reachability of o and those of o's k-nearest neighbours:

$$LOF_k(o) = \frac{\sum_{o' \in N_k(o)} \frac{lrd_k(o')}{lrd_k(o)}}{\|N_k(o)\|} = \sum_{o' \in N_k(o)} lrd_k(o') \cdot \sum_{o' \in N_k(o)} reachdist_k\left(o \leftarrow o'\right)$$

The LOF value grows as the local reachability density of o gets lower and the local reachability density of the k- Nearest Neighbours of o gets higher.

Density-based spatial clustering of applications with noise (DBSCAN)

DBSCAN is a data clustering algorithm which relies on a density-based notion of cluster (Sander, 1998). It is adequate for discovering clusters of arbitrary shape in spatial dataspaces with noise. The algorithm basic concept is to group together points in high-density by greedily agglomerating those that are close to each other. Respectively, the points that lie alone in low-density regions are marked as outliers.

The local point density at a point p is defined by two parameters:

(1) ε is the radius for the neighbourhood of point p. The ε-Neighbourhood is the representation of all points within a radius ε of from the point p and is given by :

$$N_\varepsilon(p) := \{q \; in \; data \; set \; D \; | dist(p, q) \leq \varepsilon \}$$

(2) MinPts is the minimum number of points in the given neighbourhood N(p).

The algorithm starts from a random point and calculates the point's ε-Neighbourhood. A cluster is created if the point's ε-Neighbourhood contains a sufficient number of points or the point is labelled as noise if the number of points is not enough. Though, a noise point can become a part of a cluster at the next steps of the algorithm. The ε-Neighbourhood of the points that are a dense part of a cluster are noted as part of the cluster too. This process is repeated until the density-connected cluster is finished and then, a new point is investigated to be classified as dense in a cluster or noise.

DBSCAN has the advantage of forming clusters with arbitrary shape and size. The number of clusters is automatically defined. The algorithm can easily separate clusters from surrounding noise and can be supported by spatial index structures.

6.3.1.3 SVM Classification

Support Vector Machines (SVMs) are very efficient supervised learning methods used for classification, regression and outliers detection. They were developed by Cortes and Vapnik (Cortes, 1995) for binary classification. SVMs are very efficient in high dimensional spaces using a subset of training points in the decision function ending up in memory saving. The algorithm is based on the concept of finding decision planes that define decision boundaries. A decision plane divides a set of objects that do not have the same class associations. The division aims at maximizing the margin between the classes closest points. The points lying on the boundaries are called support vectors, and the middle of the margin is the optimal dividing hyperplane. SVM handles the non-linear data spread by using a kernel function which transforms the data into a higher dimensional feature space in order to perform linear separation.

6.3.2 Application

In this section the application of heatmaps is provided and briefly described below.

6.3.2.1 Heatmaps

For the specified use case, we focus on the early detection of motor failure in the reflow ovens at BSL. A number of 20 fans per oven are measured. The workflow of the ovens is divided on three different conditions: the *wake-up*, the *work under load* and the *cooldown.* There were three available measurements the *set point*, the *present value* and the *output power*. The variable with the most useful information is the *present value* of the real temperature measured by the thermometer. As for the other variables *set point* is steady and *output power* is always zero. Thus, the heatmaps were created based on *present value* variable measurements.

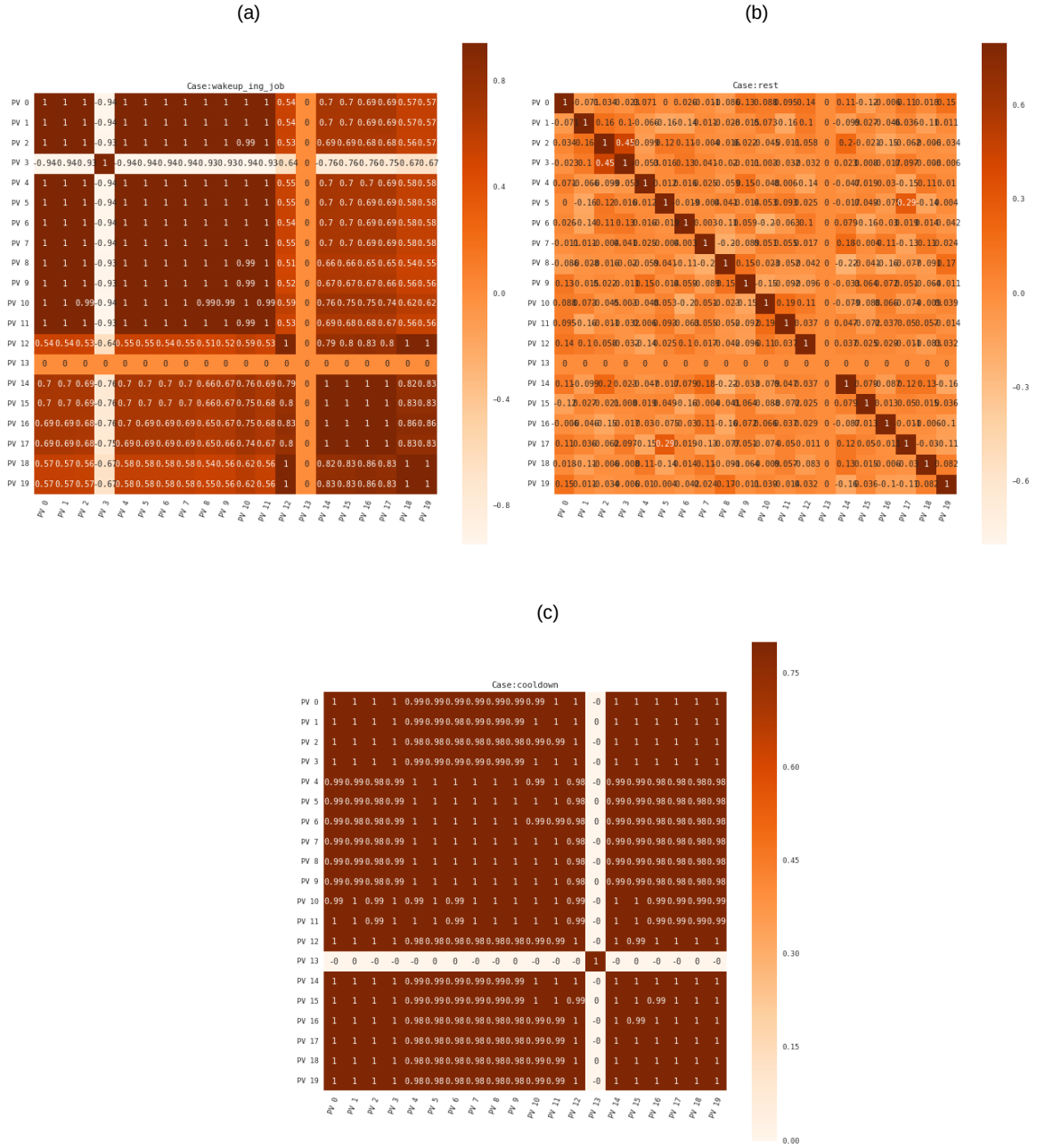


Figure 18: Correlation heatmaps overlay for the different condition of the oven: (a) wake up, (b) work under load and (c) cooldown.

For the first set of the available data the *present value* measurements per condition were: *wake up* (136), *work under load* (131) and *cooldown* (19). Thus, based on the current measurements, the pearson's correlations were calculated among ovens' fans for the different conditions and the correlation heatmap of each condition are given on figure 12. On the correlation heatmaps, dark red areas show a direct positive correlation, whiter areas show an indirect negative correlation (or anti-correlation), while orange areas indicate no apparent correlation between variables. A significant output of the correlation heatmaps described on figure 12 is that during *wake up* (see Figure 11(a)) and *cooldown* (see Figure 12(c)) conditions variables are corrlated on an

obvious manner as they begin stop functioning concurrently. On the other hand, on the *work under load* condition (see Figure 12(b)), fans work independently.

6.3.2.2 Outlier Detection

At this point, predictive maintenance will be based on failure prediction through outlier detection. The scope is to find observations in the audio data which deviate so much from the other samples as to indicate that there might exist a possible failure in the reflow ovens. An operational period can be considered as an outlier and indicate an upcoming abnormality. For the outlier detection step, three algorithms were implemented in the third dataset described in section 5.3 above: MAD, LOF and DBSCAN. The calculated outliers will constitute the false class of the upcoming classification and prediction process.

MAD

The first algorithm for outlier detection is MAD. The MAD value is calculated over a rolling window with a fixed number of data points of the sample. This defines the number of raw observations used to calculate the MAD value. This technique is sensitive in local outlier detection and is easily computed. A rejection criterion of the MAD value must be defined. Miller (Miller, 1991) proposes the values of 3 (very conservative), 2.5 (moderately conservative) or even 2 (poorly conservative). The threshold 3 is chosen at this point and the outlier criterion is:

$$-3 * MAD > x_i - M_j > 3 * MAD$$

The values that lie between these limits are considered outliers. Figure 19 below shows the detected outliers in the acoustic data of one of the fans.

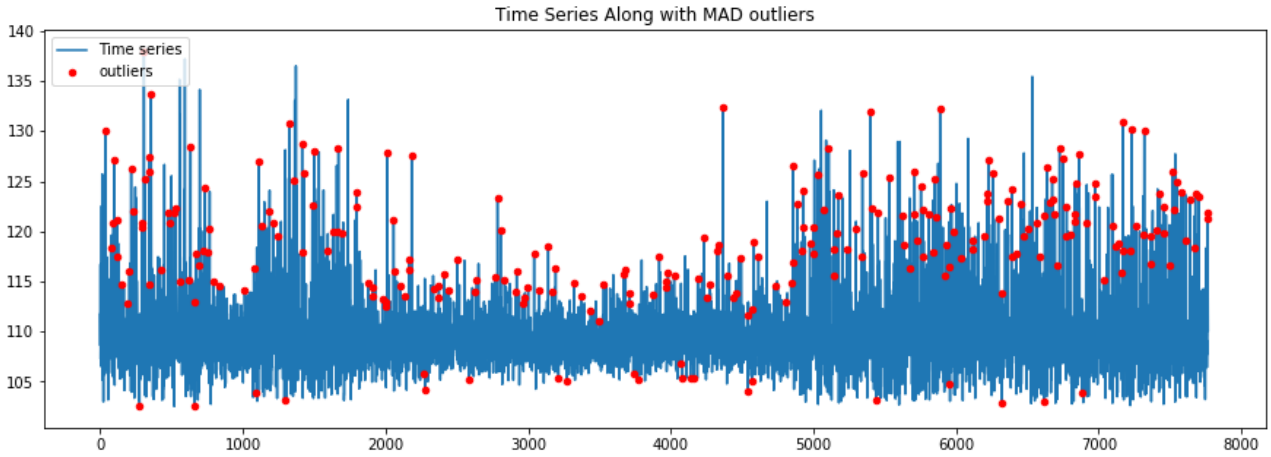


Figure 19: The MAD outliers pointed in the time series of the acoustic data of an indicative fan.

It is worth to mention that the number of outliers pointed is strictly related to the size of the rolling window. As the size of the window increases, the number of detected outlier points decreases. Figure 20 illustrates the relative rate of the number of outliers to the selected windows size.

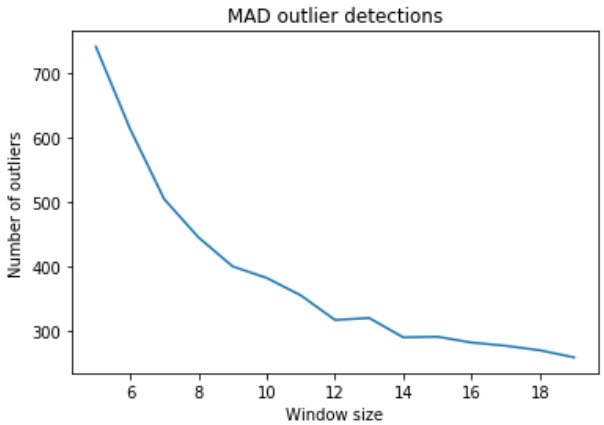


Figure 20: The ratio of number of detected outliers to the rolling window size.

LOF

Next, the Local Outlier Factor (LOF) algorithm is implemented to compute the local density deviation of an acoustic data sample with respect to its neighbors. It considers as outliers the samples that have a substantially lower density than their neighbors. The number of neighbors is set to 20 and the indicative outlier detection for one of the fan's acoustic data is shown in figure 21. The algorithm identified 387 outliers out of 7767 data points.

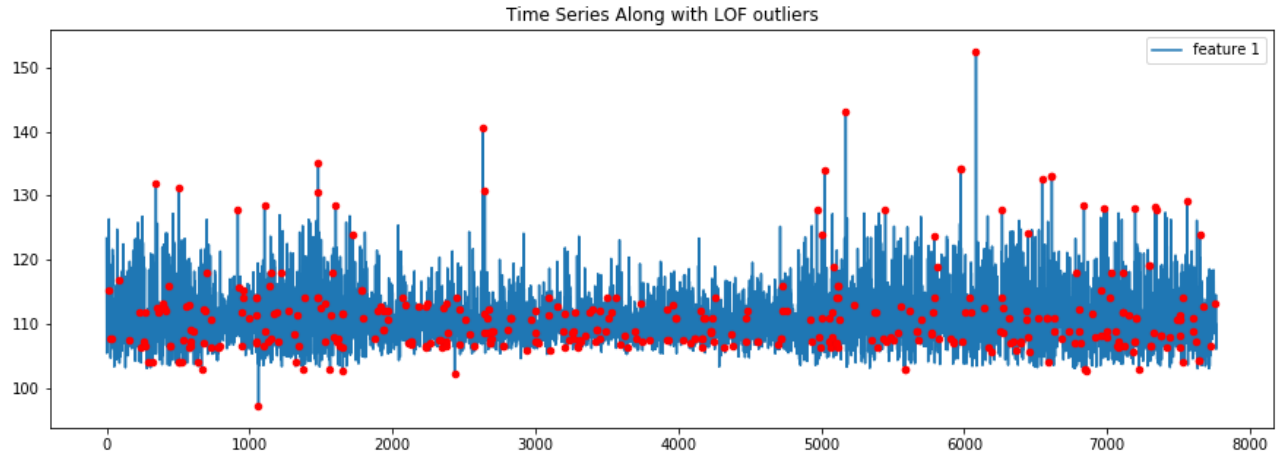


Figure 21: The LOF outliers pointed in the time series of the acoustic data of an indicative fan.

DBSCAN

The last outlier detection algorithm used is DBSCAN. The data are fitted to DBCAN, clusters are created and each acoustic sample is assigned to a cluster. The number of clusters is estimated automatically and outliers (noise) are assigned to the -1 cluster. The parameters of DBSCAN are selected after a few trials and are eps=0.1 and minPts=100.

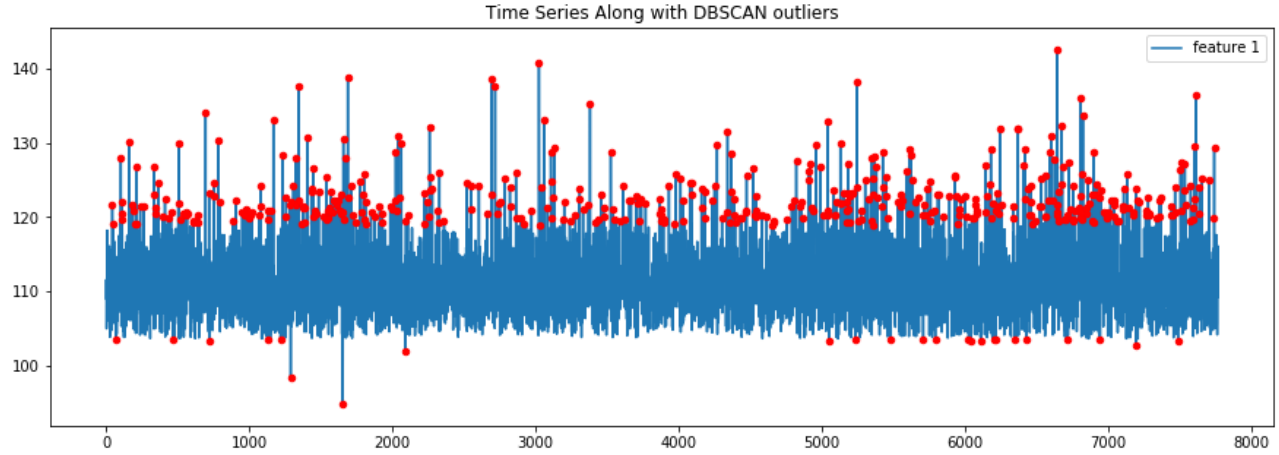


Figure 22: The DBSCAN outliers pointed in the time series of the acoustic data of an indicative fan.

The three algorithms search for samples that deviate from the majority. The lack of faulty data lead to a quite small number of detected outliers, though DBSCAN identified the most outlier points, 1149 out of 7767 samples, which are noted as the faulty class for the classification process.

6.3.2.3 SVM Classification

The previously described algorithms are used for indicating noisy points in the acoustic samples. The outliers are characterized as faulty data in order to create a binary class problem. However, the faulty data are only 14% of the overall data points, forming an imbalanced dataset. A classifier fed with this dataset will be more sensitive to identify the majority class of non-faulted fans and the classification will be biased always predicting the positive class.

Data pre-processing

In order to address the issue of imbalanced dataset, an oversample of the minority class is performed. Synthetic Minority Oversampling Technique (SMOTE) is used to resample and synthesize new elements for the minority class, based on those that already exist (Chawla, 2002). A minority class point is chosen randomly and the k-neighbours of it are calculated. The synthetic data are placed between the picked point and the calculated neighbours. SMOTE technique diminishes the issue of overfitting as the new synthetic data are created rather than reproducing samples and corrupting information.

Figure 23 illustrates the oversampling of the detected outliers (red points). The original dataset in shown in the left sub-figure of figure 23, where the positive class consists of the 86% of the dataset and the negative class is the 14% respectively. After implementing SMOTE for oversampling the two classes contain 50% each (right sub-figure of figure 23).

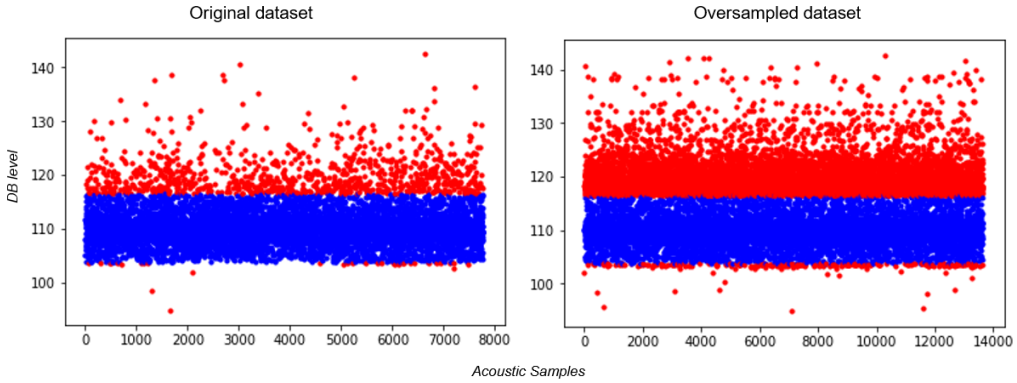


Figure 23: The oversampling with SMOTE

An SVM classifier is used for training the new dataset. The two possible label classes are faulted acoustic sample (=1) and non-faulted acoustic sample (=0). A training dataset is created including the 70% of the overall dataset and the rest 30% is used for the testing dataset. The SVM model is trained with the following parameters: cost=0.5 and kernel = RBF.

In order to evaluate the trained model the testing dataset is used to predict the class of the points based on their DB values. Figure 24 is the illustration of the confusion matrix, to evaluate the quality of the output of the SVM classifier on the acoustic data set. The diagonal elements represent the number of points for which the predicted label is equal to the true label, while off-diagonal elements are those that are mislabelled by the classifier. The diagonal values of the confusion matrix are quite high, indicating many correct predictions of the faulted and not-faulted acoustic data.

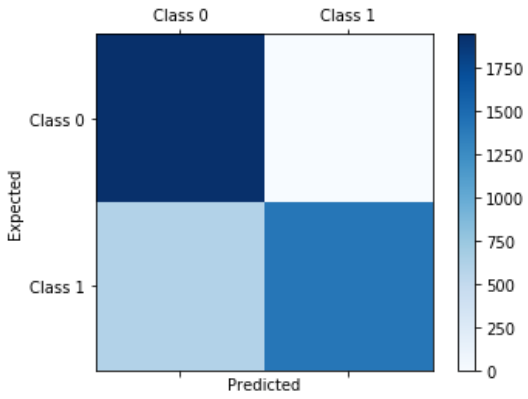


Figure 24: The confusion matrix of the SVM classifier on the acoustic dataset.

The evaluation metrics are accuracy, precision, recall and F1 score. Accuracy is a ratio of correctly predicted observation to the total observations. Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. Recall is the ratio of correctly predicted positive observations to the all observations in actual positive class. The F1 score is a weighted average of the precision and recall. Table 5 presents the results of the trained model.

Table 5: Evaluation metrics of SVM model

Metrics	Result
Accuracy	0.85
Precision	0.76
Recall	1.0
F1 score	0.86

The evaluation metric results indicate that the model is capable of detecting faulted measurements. All of the metrics are over 75%. Note that the recall metric reaches 1.0 because of the fact that the non-faulted data are generated and do not exist in the initial dataset. New measurements can be given to the SVM prediction model, optimally live data that can predict if there exist a possible failure in the reflow ovens with 85% accuracy.

7 Deployment of Simulation and Forecasting Tool

The SFT algorithms deployed using Docker[4] deployment tool as the rest components of the COMPOSITION project do.

Docker is an open-source project aiming at automating the deployment of applications as portable, self-sufficient containers that can run virtually anywhere, on any kind of server. It can be considered as a lightweight alternative to full machine virtualization provided by hypervisors. While in the traditional hypervisor approaches each virtual machine (VM) needs its own operating system, in Docker applications operate inside a container that resides on a single host operating system that can serve many different containers at the same time. Docker containers are designed to run on a wide range of platforms ranging from physical computers to bare-metal servers and up to cloud clusters.

A dedicated web management tool, named Portainer[5] is used. It offers to partners and in general all technical stakeholders the ability to publish, run and test the single COMPOSITION components, which are under their respective responsibility. Continuous monitoring and logging infrastructure allow deep analysis of the performances of deployed software that can both be carried before the final deployment inside factories and during real-world operation.

The SFT component designed as a completely web-based component. It is implemented in Python and it is deployed in Docker containers. Different Docker images are built for the different supported algorithms. The Docker images that have been created:

- use pre-built official Python image
- import dependencies/libraries
- add the python files that contain the SFT algorithms

The Docker containers of the SFT are deployed at COMPOSITION intra-factory production server.

An example of an SFT container for the KLE-1 use case, which is deployed on the COMPOSITION intra-factory production server, is presented below:

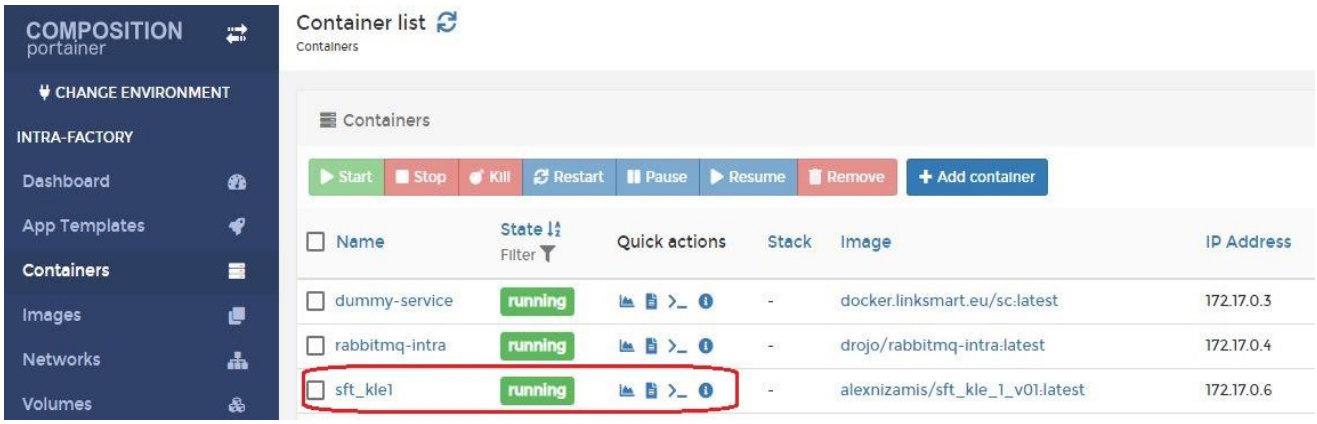


Figure 25: Example of SFT Docker Deployment

From a security perspective, the SFT Docker containers do not publish ports, so there are no ports or endpoints that should be secured. The SFT containers post data to other containers (e.g. DFM container) were contained in the same protected network of the COMPOSITION intra-factory production server.

[4] https://www.docker.com/
[5] http://portainer.io/

8 Conclusions

In conclusion, the deliverable D3.5 Computational Modelling, Simulation and Prediction in Production II describes the whole effort spent in the period M4 to M28 and represents the final status of T3.3 – Simulation and Forecasting in Production and Logistics of WP3. More precisely, this report documents the results in the part of production as the logistics results are documented in D3.7 – Computational Modelling, Simulation and Prediction of Logistics I which will be submitted in parallel with the current report.
In the first steps of Task 3.3 the research and development were focused on the architecture design, the analysis of existing pilot partners' data, the selection of the proper methodologies for this data, and the development of the aforementioned methodologies for the use cases with the highest priority. Based on the analysis of the applied methods, the Probability Model for KLEEMANN's maintenance decision support enhance the efficiency of the DSS component by providing the possibility of future faults (mechanical, hydraulic and electrical). The probabilities of fault are available to DSS via the DFM component. The Slope Statistics Profile (SSP) methodology provides an estimation of fullness from indoor bins in the KLEEMANN's shop-floor. Based the estimation of fullness, the simulation and prediction engine provides the optimal route from the bins inside the factory to the scrap containers based on Dijkstras' algorithm. Furthermore, the application of Correlation Heatmaps, Median Absolute Deviation, Local Outlier Factor and density-based spatial clustering of applications with noise methodologies provides anomalies' detection for the BSL predictive maintenance scenarios in order to support the decision making process. At its current state, the components of the Simulation and Forecasting Tool are deployed utilizing Docker deployment tool.

List of Figures and Tables

8.1 Figures

8.2 Tables

9 References

(Ahuja, 1993) R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. Network Flows: Theory, Algorithms, and Applications. Prentice Hall, 1993

(Bang-Jensen, 2000) Bang-Jensen, Jørgen; Gutin, Gregory (2000). "Section 2.3.4: The Bellman-Ford-Moore algorithm". Digraphs: Theory, Algorithms and Applications (First ed.). ISBN 978-1-84800-997-4.

(Breunig, 2000) Breunig, M. M., Kriegel, H.-P., Ng, R. T., Sander, J., (2000). LOF: Identifying Density-based Local Outliers (PDF). Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data. SIGMOD. pp. 93–104.

(Chawla, 2002) Chawla, Nitesh V., et al. "SMOTE: synthetic minority over-sampling technique." *Journal of artificial intelligence research* 16 (2002): 321-357.

(Cherkassky, 1996) Cherkassky, Boris V.; Goldberg, Andrew V.; Radzik, Tomasz (1996). "Shortest paths algorithms: theory and experimental evaluation". Mathematical Programming. Ser. A. 73 (2): 129–174

(Cortes, 1995) Cortes, C. & Vapnik, V. (1995). Support-vector network. Machine Learning, 20, 1–25.

(Gauss 1809) Gauss, C.F., Theoria Motus Corporum Coelestium in Sectionibus Conicis Solem Ambientum. (1809)

(Gauss, 1821) Gauss, C.F., Theoria combinationis observationum erroribus minimis obnoxiae. (1821/1823)

(Hart, 1968) Hart, P. E.; Nilsson, N. J.; Raphael, B. (1968). "A Formal Basis for the Heuristic Determination of Minimum Cost Paths". IEEE Transactions on Systems Science and Cybernetics SSC4. 4 (2): 100–107

(Hawkins, 1980) Hawkins, Douglas M. *Identification of outliers*. Vol. 11. London: Chapman and Hall, 1980.

(Kenneth, 2003) Kenneth H. Rosen (2003). Discrete Mathematics and Its Applications, 5th Edition. Addison Wesley. ISBN 0-07-119881-4.

(Legendre, 1805) Legendre, A.M,. Nouvelles méthodes pour la détermination des orbites des comètes, Firmin Didot, Paris, 1805. “Sur la Méthode des moindres quarrés” appears as an appendix.

(Miller, 1991) Miller, J. (1991). Reaction time analysis with outlier exclusion: Bias varies with sample size. The Quarterly Journal of Experimental Psychology A: Human Experimental Psychology, 43A(4), 907-912.

(Sander, 1998) Sander, J., Ester, M., Kriegel, HP. et al. Data Mining and Knowledge Discovery (1998) 2: 169. https://doi.org/10.1023/A:1009745219419

(US Patent, 1993) "United States Patent and Trademark Office", 1993-09-01.

(Vafeiadis, 2011) Vafeiadis, T., Bora-Senta, E., Kugiumtzis, D., (2011). ‘Estimation of linear trend onset in time series’. Simulation modelling – Practice and Theory, Vol. 19, Issue 5, p 1384 - 1398.

(Zhan, 1998) F.B. Zhan and C.E. Noon. Shortest Paths Algorithms: An Evaluation Using Real Road Networks. Transportation Science, 32:65–73, 1998.