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

D5.2 Big Data Mining and Analytics Tools II

Date: 2019-01-31

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:	D5.2 Big data mining and analytics tools II
Document version:	1.0
Document owner:	FIT
Work package:	WP5 – Integration of Internal and External Elements
Task:	T5.1 – Multi-Level and Cross-Domain Big Data Analysis and Management
Deliverable type:	[OTHER]
Document status:	☒ Approved by the document owner for internal review ☒ Approved for submission to the EC

Document history:

Version	Author(s)	Date	Summary of changes made
0.1	José Ángel Carvajal Soto (FIT)	2019-01-17	Initial TOC
0.2	José Ángel Carvajal Soto (FIT)	2019-01-22	Add content
0.3	Thanasis Vafeiadis, Nikolaos Alexopoulos, Christos Ntinas (CERTH)	2019-01-22	Big Data Visual Analytics' chapter added
0.4	José Ángel Carvajal Soto (FIT)	2019-01-28	Adopt changes from the review 1.
0.5	José Ángel Carvajal Soto (FIT)	2019-01-30	Adopt changes from the review 2
0.6	Alex Graß (FIT)	2019-01-30	Add descriptive data analysis
1.0	José Ángel Carvajal Soto (FIT)	2019-11-31	Final version

Internal review history:

Reviewed by	Date	Summary of comments
Vasiliki Charisi (ATL)	2019-01-28	Minor changes of the formatting and language. Missing conclusion.
Matteo Pardi (NXW)	2019-01-30	Minor changes of the formatting and language. Missing conclusion and Acronyms

Contents

List of Figures

List of Tables

1. Executive Summary

The Task 5.1 Multi-Level and Cross-Domain Big Data Analysis and Management can be split into two parts. On one side, the collection processing and management of data, which is the Big Data Analysis task. On the other side, the visualization of this data in order for humans to be able to interpret it easier and faster, which is the Visual Analytics task.

The Big Data Analytics task was mainly implemented by the LinkSmart® IoT Learning Agent (LA). Most of the new feature development had happed on the period between M5 to M15 and described in D5.1. This deliverable describes the period from M15 to M29, in which most of the work had been on the implementation of the pilots using the feature of the LA developed previously.

The main task of the LA is in the BSL-2 use case, in which the COMPOSITION IIMS should collect sufficient data to predict a future failure and notify the adequate employee. To do so, the BMS connects to BSL subsystems and serializes and transmits the information to the COMPOSITION cloud. Then the LA collects and processes the data for executing the learning process. The learning process is the management of the data and machine learning model in order to be able to create live reliable predictions and continuous learning. This process is orchestrated following the CEML methodology described in D5.1. In this process, the data is processed, cached and delivered to the DLT for creating predictions and train the model. With the predictions sent by the DLT, the LA propagates it to the Visual Analytics and other COMPOSITION subsystems such as DSS and SFT.

Finally, the LA had been tested beyond the normal data loads faced in COMPOSITION pilots. This had been done to test the Big Data capacity of the LA and its scalability capacity. The LA has shown that is able to process effectively a big amount of data and scale in case the data exceeds the capacity of a single LA capacity.

A Visual Analytics tool has been implemented in order to enhance the decision support offered by COMPOSITION IIMS. The tool provides to the end-user the capability to visualize, analyze and explore industrial data derived from multiple sources. Furthermore, coordinated views of the data are supported in order to enable multifaceted perception and discovery of hidden subtleties in it. The interactive Visual Analytics tool imports data from Data Analytics tools of the project and based on this data it applies visualization techniques and presents the output to the users as graphical representations. The tool completes the UIs of the project alongside with DSS and Marketplace interfaces and supports advanced visualizations that were not able to support by the aforementioned tools.

2. Abbreviations and Acronyms

Acronym	Meaning	Description
ANN	Artificial Neural Network	Computing systems vaguely inspired by the biological neural networks that constitute animal brains.
BSL	Boston Scientific	Partner Company in COMPOSITION
CEML	Complex-Event Machine Learning	A methodology to enable runtime autonomous and re-deployable machine learning. The method is partially developed in COMPOSITION project and described in D5.2
CEP	Complex-Event Processing	A method that combines data from multiple sources to infer events or patterns that suggest more complicated circumstances.
DLT	Deep Learning Toolkit	Component which implements the Machine Learning models used in COMPOSITION
DSS	Decision Support System	COMPOSITION subsystem and interface that provides information to operatives to take decisions.
EPL	Esper Procedural Language	An SQL-like language used by Esper CEP engine develop by Esper Tech Inc.
IIMS	Integrated Information Management System	Intra-factory subsystem of COMPOSITION platform
JSON	JavaScript Object Notation	An open-standard file format that uses human-readable text to transmit data objects consisting of attribute–value pairs and array data types (or any other serializable value)
JWS	JSON Web Signature	An open standard for JSON keeping the integrity of JSON documents; and verifying the document producer.
KLE	Kleemann	Partner Company in COMPOSITION
LA	LinkSmart® IoT Learning Agent	Component that implements the CEML. The component has been developed partially in COMPOSITION and in other projects as well.

MQTT	Message Queuing Telemetry Transport	An ISO standard (ISO/IEC PRF 20922) publish-subscribe-based messaging protocol
Msg	Message	Use as unit in this deliverable. Also msg/s (message per second)
REST	Representational State Transfer	A software architectural style that defines a set of constraints to be used for creating web services
SDK	Software Development Kit	Set of libraries, interface, and documentation for developers to develop other software or solution based on them.
SFT	Simulation and Forecasting tool	Component which implements the simulation and forecasting methods used in COMPOSITION
TCP	Transmission Control Protocol	One of the main protocols of the Internet protocol suite that governs how the payload should be transmitted.
VA	Visual Analytics	An outgrowth of the fields of information visualization and scientific visualization that focuses on analytical reasoning facilitated by interactive visual interfaces
VM	Virtual Machine	An emulation of a computer system

3. Introduction

In this deliverable, we present the developments between the M15 to M29 regarding the Big Data Analytics tools. Most of the new features were developed in the first period of the project, therefore; most of the work described on the current deliverable document and developed in the mentioned period, are the development of the pilot use case. This includes bug fixing, integration, and, testing and benchmarking.

This is the second issue of this deliverable, and describes the developments of the software developed in task 5.1 Multi-Level and Cross-Domain Big Data Analysis and Management. This task was responsible for processing and managing the data on runtime. The task can be divided into three. (1) Collect, process and manage the data in run time for on-the-fly processes. (2) Manage the data and orchestrate the continuous learning process. (3) Visualize the data for operative be able to analyze it. Part 1 and 2 are made by the LA developed by FIT. However, the learning itself happens in the DLT, developed by ISMB in task T5.2 but integrated in T5.1. Finally, the visualization is done by the Visualization Tool developed by CERTH in this task.

The deliverable is structured in the following manner. In chapter 4, the development in the big data analytics processing is described. In this chapter, we explore the usage of data analytics in the pilot use case. Afterward, we present some testing and benchmarking performed on to the data analytics service. In chapter 5, we present the implementation of the big data visualization as well as its functionality. Finally in chapter 6, we describe the contribution of COMPOSITION of this task and technology used. We close with a short conclusion.

4. Big Data Analytics Provided by the LinkSmart® IoT Learning Agent

4.1 Pilot BSL-2 Use Case Predictive Maintenance Scenario (To Be)

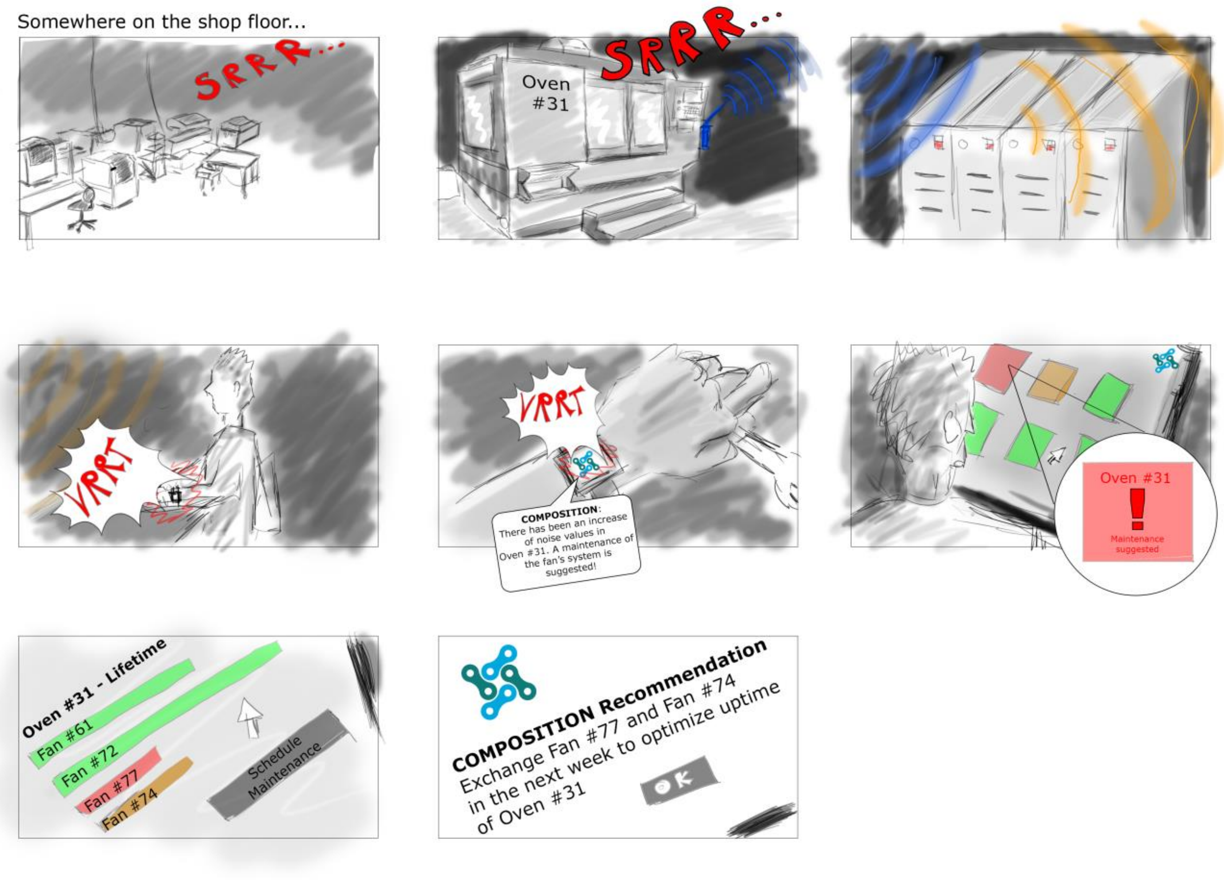

Somewhere on the shop floor...
SRRR...
Oven #31
SRRR...
VRRT
VRRT
COMPOSITION: There has been an increase of noise values in Oven #31. A maintenance of the fan's system is suggested!
Oven #31
Maintenance suggested
Oven #31 - Lifetime
Fan #61
Fan #72
Fan #77
Fan #74
Schedule Maintenance
COMPOSITION Recommendation
Exchange Fan #77 and Fan #74 in the next week to optimize uptime of Oven #31
OK

Figure 1: Predictive Maintenance Scenario for Use Case BSL-2: An employee of BSL is notified by the COMPOSITION system about the near future failure of a fan in the oven

The COMPOSITION IIMS collects information about actual performance (real-time) and history of performance. Data is captured at machine level (laser power, soldering paste, fans, mechanical conveyers, etc.). The capture data is power, temperature, noise from fans and the machine log files.

The actual status is compared to static data models (performance specs, history, costs) about the optimum process performance and algorithms can predict the likely point in time where critical components in the machine or process may fail thus causing the manufacturing process to be disrupted or products to be scrapped. Based on historic performance the prediction of failures can be further improved by using different machine learning technologies. The COMPOSITION Early Failure Prediction System is shown in Figure 2.

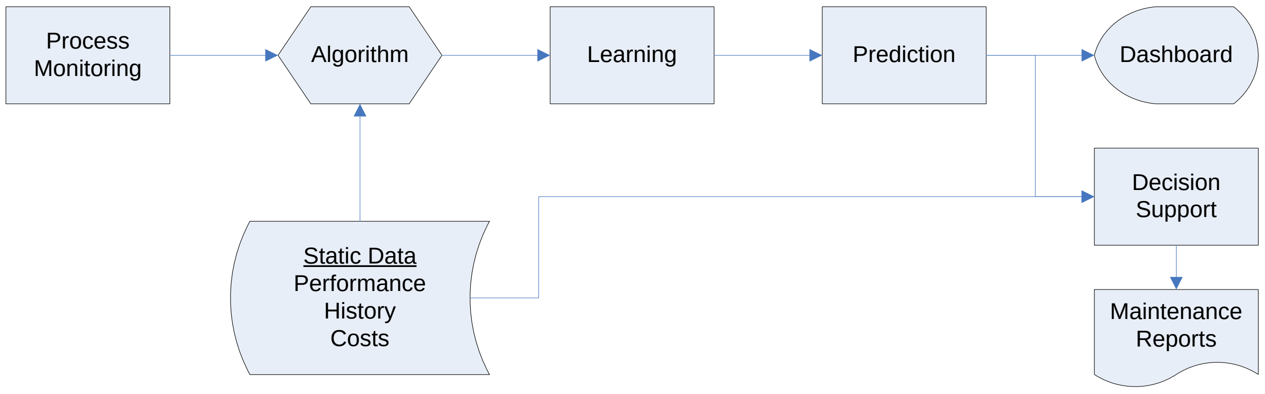


Figure 2: COMPOSITION Predictive Maintenance System

The predictions are presented to the operator to support their decision about when and what to replace before failure occurs. The operator will view a selection of critical components and their estimated time of failure as shown in Figure 3.

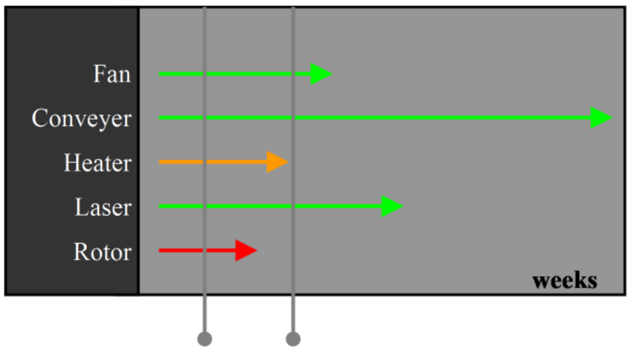


Figure 3: Replacement Decision Support

Replacing a machine part means unnecessary costs to the manufacturing process. Replacing it too late may lead to the part failing and stopping the production process. Replacing the part too early means its useful lifetime is reduced and the total cost of owner ship increases. The optimum time of replacement is also influenced by the time it will take to replace the part. A fast exchange will minimise downtime. Other parts will also influence the decision. Several parts may be easier to replace together (e.g. the parts are in all the same physical place but access is difficult) whereas other parts may be replaced easily.

The prediction will present the operator with decision support as exemplified in Figure 3 where each part has a predicted remaining lifetime measured in months. The Rotor needs to be replaced, but the Fan can last a long time.

Choosing the left-hand time-line indicates that it would make sense to replace the Rotor in the coming month. However, it would be cost effective to replace the fan, which has a remaining lifetime of 8 months. But the Heater is located next to the Rotor in the heart of the production unit and the Heater needs to be replaced in the coming two months. If the cost of the Heater is minor, it may be economically wiser to replace it at the same time as the Rotor.

The prediction provides certain margins of error. Assuming that downtime cost is significantly higher than the cost of the replaced components, it may be more economically effective to postpone the replacement of the Rotor until the time indicated by the right-hand time line, which may even be a scheduled down-time. At this point, it will be wise to replace the Rotor, the Heater, and the Fan all together. There is a calculated risk that the Rotor may fail before, but this must be balanced against the potential gains of delaying the downtime.

The Laser is the next parts in need of replacement, but since it is a minor operation and a high-valued part, this part will be monitored individually for replacement.

Finally, replacing the Conveyer is a major task, underdone in its own time and it is not of necessary consideration at present.

The COMPOSITION IIMS will help the pilots to efficiently and effectively manage machine downtimes and failures based on the prediction of failures of critical components. Information such as levels and temperature of solvents, vibration of machines, etc., will be provided in the IIMS. Prediction of the Blower motors within the Rhythmia Ovens are very important to BSL from a quality and cost perspective. Potential cost of a non-recoverable oven alarm (motor/blower failure) resulting in non-conforming product being scrapped is estimated as $60K

- Two To-Be use cases have been identified for Predictive Maintenance in Section 8.

4.2 Realization

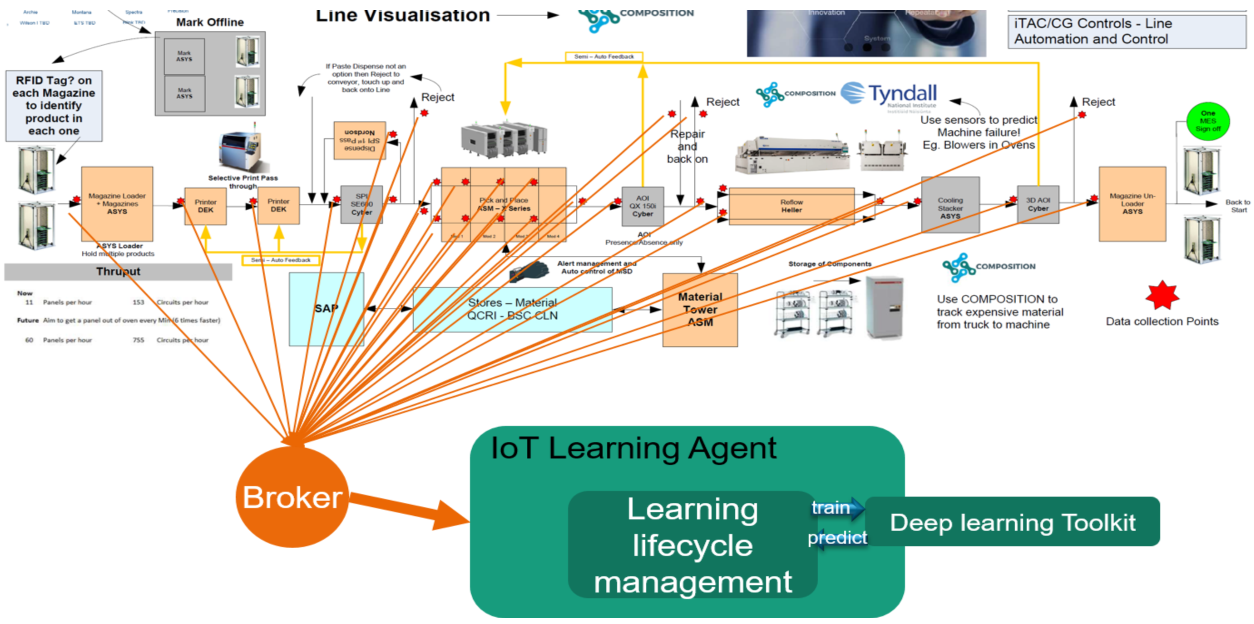

Mark Offline
Mark ASYS
Mark ASYS
Line Visualisation
COMPOSITION
iTAC/CG Controls - Line Automation and Control
RFID Tag? on each Magazine to identify product in each one
Magazine Loader + Magazines ASYS
ASYS Loader Hold multiple products
Selective Print Pass through
Printer DEK
Printer DEK
Semi – Auto Feedback
If Paste Dispense not an option then Reject to conveyor, touch up and back onto Line
Dispense SPI 1st Pass Nordson
Reject
SPI SE600 Cyber
Pick and Place ASM – X Series
Mod 1
Mod 2
Mod 3
Mod 4
Semi – Auto Feedback
AOI QX 150i Cyber
AOI Presence/Absence only
Repair and back on
Reject
COMPOSITION
Tyndall National Institute
Use sensors to predict Machine failure! Eg. Blowers in Ovens
Reflow Heller
Cooling Stacker ASYS
3D AOI Cyber
Reject
Magazine Un-Loader ASYS
One MES Sign off
Back to Start
Thruput
Now
11 Panels per hour 153 Circuits per hour
Future Aim to get a panel out of oven every Min (6 times faster)
60 Panels per hour 755 Circuits per hour
SAP
Stores – Material QCRI - BSC CLN
Alert management and Auto control of MSD
Material Tower ASM
Storage of Components
COMPOSITION
Use COMPOSITION to track expensive material from truck to machine
Data collection Points
IoT Learning Agent
Broker
Learning lifecycle management
train
predict
Deep learning Toolkit

Figure 4: Description of the Data Collection and Processing Mechanism for BSL-2 Use Case

In order to build an accurate model, we set a continuous learning process, which involves the IoT Learning Agent (LA) and the Deep-Learning Toolkit (DLT). The process analyses, processes and arranges data for the DLT and deliver it in real-time. Additionally, The LA monitors the development of the model in real-time too.

4.2.1 Learning Process

The learning process is the instance of a CEML process (explained in D5.1). The process instance has the following lifecycle:

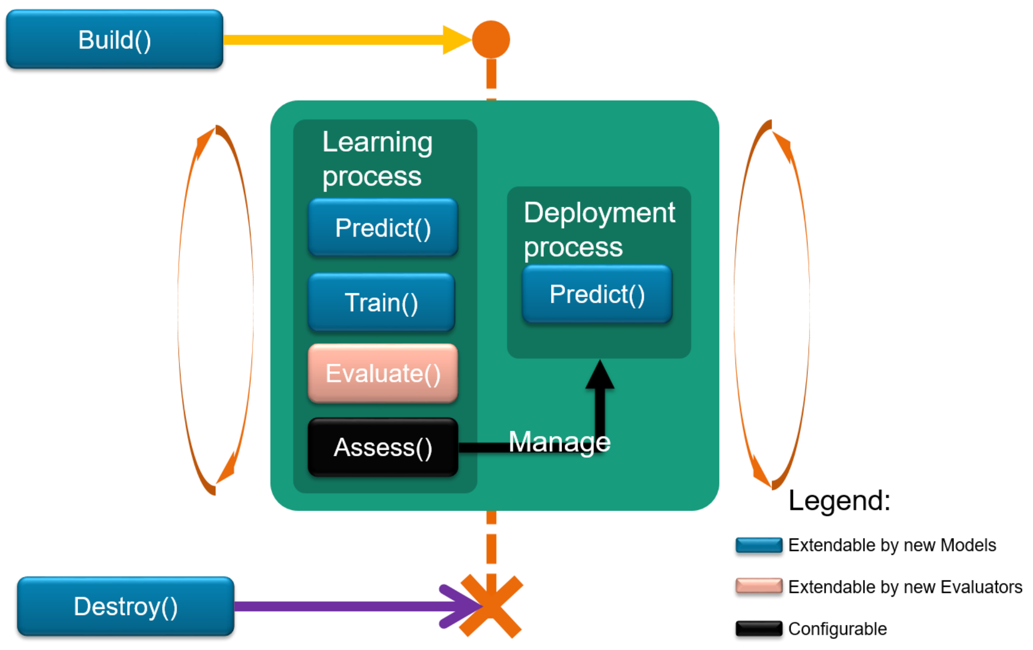


Figure 5: Machine Learning Process Lifecycle Scheme

The instance of the learning process is managed by the LearningHandler. The LearningHandler is the component of the LA who manage the overall learning process, and there is one learning handler per process. The most important part of the learning process is the continuous learning and evaluation. Below (Figure 6) we describe the continuous learning and evaluation step by step.

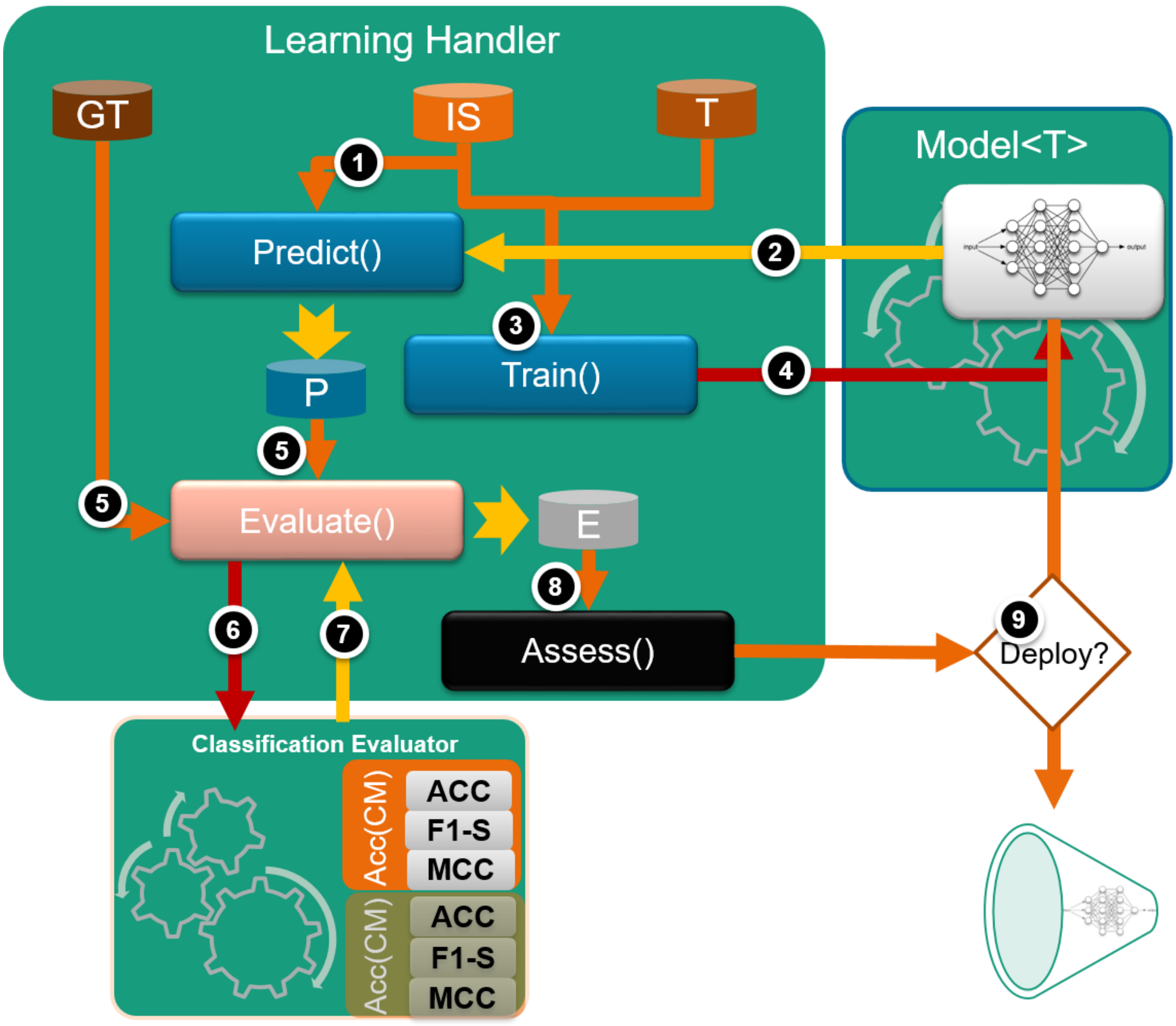


Figure 6: Continuous Learning Algorithm

Input Set: Set of inputs that in case of prediction, it is use by the Model to create a prediction; in case of learning, it is use together with the **Target** to train the Model. An **Input Set** it is the equivalent of a single row in a Data Set in Batch Learning.

Target: It is the value that the Model should learn/train together with the **Input Set**. In classification cases, it is the label, in regression it is a double or a vector of them.

Ground Truth: It is the truth against which the prediction of the Model will be compared to evaluate the performance of the Model. In most of the cases, the **Target** and **Ground Truth** are the same, just for predictions of time series using sliding data they must be treated different.

Model: It is a class that implements Build, Predict, Train and Destroy.

Evaluator: A class that implements Evaluate. Currently, there is the ClassificationEvaluator and the RegressionEvaluator, for classification and regression problems, respectively.

1. **IS** is extracted and given to the Model
2. The Model provides a Prediction **P**
3. The **IS** and **T** are given to the Model for train it (the Model)
4. The Model is trained
5. The **GT** and the **P** (step 2) are given the Evaluator. In case of Regression, RegressionEvaluator; in case if Classification, then the ClassificatorEvaluator
6. The Evaluator calculates current state of the Model

7. The Evaluator returns the state of the Model **E**
8. The using **E** the Handlers together with the given threshold (by configuration) the LearningHandler decide if the Model is ready for deployment
9. If it is ready and is not deployed yet, then is deployed. If it is not read for deployment and the Model is deployed then it is removed, otherwise do nothing.

4.2.2 Processing Scheme in BSL-2

4.2.2.1 DLT Feature Space

In order to understand the pre-processing work of the LA, we need to understand the exacted input of the DLT. The expected input of a Machine Learning model is called feature space. In this chapter, we will explain the DLT feature space.

There are three kinds of data concerning the UC-BSL-2 in the Rhythmia Oven case:

- sensors measurements
- oven events
- acoustic data.

All information types are used by the Deep Learning Toolkit (DLT) for predictive maintenance and to properly train the artificial neural networks. In order to be used in a supervised machine learning framework, this information has to be organized as a list of samples. In specific, a table of 242 columns must be provided. The table is split in following parts:

Table 1: An element of a row of the feature space divided by type

0	**1**	**2**	**3**	**4**	**...**	**39**	**40**	**41**	**42**	**43**	**...**	**234**	**235**	**236**	**237**	**238**	**239**	**240**	**241**

The leftmost part of the table (green, columns from 0 to 39) contains the values sampled from different sensors, measured at the same time.

The central part of the table (red, columns from 40 to 235) contains the mapping of events, registered at the same time of the corresponding sensors readings.

The rightmost part of the table (blue, columns from 236 to 240) contains the mapping of the five acoustic data sensors decibels one per column, registered at the same time of the corresponding sensors readings.

The last column (241) contains a label to identify the oven status. This value is only used for ANN training. The label value can be 0 if the oven is working correctly or 1 for a fault (at the moment three hi warning outside the oven stabilization or hi warning + hi deviation).

In order to collect all events associated to the current sensor reading, the DLT should receive the valid sensor reading each time a new event is received. This duplication is handled by the DLT internal logic. So, if two events happen during the same sampling interval from the sensors (currently 5 minutes) the DLT expects to receive two records with identical columns from 0 to 39 and then the two events reported separately as for the events mapping. In this case, acoustic data columns from 236 to 240 must be 0.

Having following vector below, we will define some concepts according to the Learning Agent:

Table 2: An element in the "row" of the DLT feature space, plus the local ground truth

0	**1**	**2**	**3**	**4**	**...**	**39**	**40**	**41**	**42**	**43**	**...**	**234**	**235**	**236**	**237**	**238**	**239**	**240**	**241**

Input set/vector: The numbers in green are the values that represent the input of the model; and they are the values the model will receive when the model will make a prediction.

Target/label/ground truth set/vector: The number in gray are the values that represent the output of the model; and they are the values the model will predict when the model receives an input vector**.**

Learning (vector) Instance: it is the concatenation of the input vector and the target vector, in this order. In the example, is the entire vector?

Feature Vector: are the concepts/types/observed objects of the learning (vector) instance.

Table 3: A row in the feature space

Sample	**Oven sensors measurements**					**Oven events**					**Acoustic data**				
1	0	1	2	3	...	39	40	41	...	235	236	237	238	239	240
2	0	1	2	3	...	39	40	41	...	235	236	237	238	239	240
3	0	1	2	3	...	39	40	41	...	235	236	237	238	239	240
4	0	1	2	3	...	39	40	41	...	235	236	237	238	239	240
5	0	1	2	3	...	39	40	41	...	235	236	237	238	239	240
6	0	1	2	3	...	39	40	41	...	235	236	237	238	239	240
7	0	1	2	3	...	39	40	41	...	235	236	237	238	239	240
8	0	1	2	3	...	39	40	41	...	235	236	237	238	239	240
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
32	0	1	2		...	39	40	41	...	235	236	237	238	239	240

A set of 32 elements of this time series (EXCLUDING the sample column) is one “element” of the DLT Input set/vector. It can be provided to the DLT predict function that returns the prevision of a future fault. The complete feature space is a 32 - elements of 32 rows with 241 values. In other words, a matrix of 32x7712 (32x32x241).

Due to BSL maintenance team usually detects a problematic fan before a catastrophic failure, it was difficult to develop a ground truth when the fan was about to fail. Together with BSL technicians a possible fan failure heuristic was developed.

Every 32 cycles, a 241th feature must be generated for each row. It is always at 0 unless one of the following patterns provided by BSL occur:

three hi warnings (any of type 58-78) are presents in the previous thirty minutes (outside the oven stabilization)

hi warning (any of type 58-78) + hi deviation (any of type 142-162) at any stage in previous thirty minutes

The result is something like the following:

Table 4: Ground truth building

Smpl.	**Oven sensors measurements**						**Oven events**						**Acoustic data**					**Add. feature**
1	0	1	2	3	...	39	40	41	46	47	...	235	236	237	238	239	240	
2	0	1	2	3	...	39	40	41	46	47	...	235	236	237	238	239	240	
3	0	1	2	3	...	39	40	41	46	47	...	235	236	237	238	239	240	
4	0	1	2	3	...	39	40	41	46	47	...	235	236	237	238	239	240	
5	0	1	2	3	...	39	40	41	46	47	...	235	236	237	238	239	240	

6	0	1	2	3	...	39	40	41	46	47	...	235	236	237	238	239	240	
7	0	1	2	3	...	39	40	41	46	47	...	235	236	237	238	239	240	
8	0	1	2	3	...	39	40	41	46	47	...	235	236	237	238	239	240	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
32	0	1	2		...	39	40	41	46	47	...	235	236	237	238	239	240	241

For calculating the ground truth, we have to look at the next 32 values starting from the last row of the input set/vector (shown as the red zone in the example below).

- The ground truth has value 0 when **ALL THE ELEMENTS** of the red zone are 0
- The ground truth has value 1 when **AT LEAST ONE ELEMENTS** of the red zone is 1.

An example follows:

Table 5: Example of the ground truth building (1)

Sample	Oven events								Acoustics					Add. feature
1	0	1	...	39	40	41	...	235	236	237	238	239	240	0
2	0	1	...	39	40	41	...	235	236	237	238	239	240	0
3	0	1	...	39	40	41	...	235	236	237	238	239	240	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
32	0	1	...	39	40	41	...	235	236	237	238	239	240	1
33	0	1	...	39	40	41	...	235	236	237	238	239	240	0
34	0	1	...	39	40	41	...	235	236	237	238	239	240	0
35	0	1	...	39	40	41	...	235	236	237	238	239	240	0
36	0	1	...	39	40	41	...	235	236	237	238	239	240	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
61	0	1	...	39	40	41	...	235	236	237	238	239	240	0
62	0	1	...	39	40	41	...	235	236	237	238	239	240	1
63	0	1	...	39	40	41	...	235	236	237	238	239	240	0
64	0	1	...	39	40	41	...	235	236	237	238	239	240	0

The green part is the DLT **Input set/vector.** The **ground truth** for this input set is 1 because column 241 of the row 62 (inside the red area) is 1.

The next input set is shifted of one position. Now there are 2 labels with value 1 in the column 241 (row 62 and 65) of the red area but the ground truth value is still 1.

Table 6: Example of the ground truth building (2)

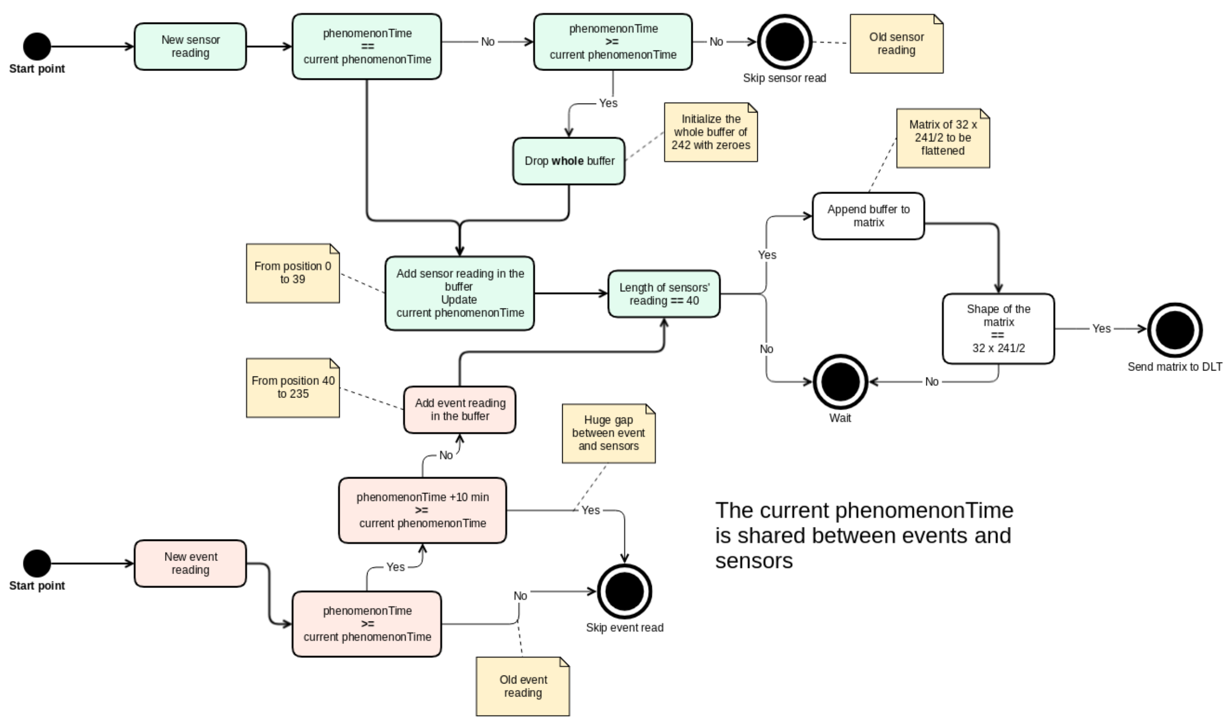
Sample	Oven events								Acoustics					Add. feature
1	0	1	...	39	40	41	...	235	236	237	238	239	240	0
2	0	1	...	39	40	41	...	235	236	237	238	239	240	0
3	0	1	...	39	40	41	...	235	236	237	238	239	240	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
32	0	1	...	39	40	41	...	235	236	237	238	239	240	1
33	0	1	...	39	40	41	...	235	236	237	238	239	240	0
34	0	1	...	39	40	41	...	235	236	237	238	239	240	0
35	0	1	...	39	40	41	...	235	236	237	238	239	240	0
36	0	1	...	39	40	41	...	235	236	237	238	239	240	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
61	0	1	...	39	40	41	...	235	236	237	238	239	240	0
62	0	1	...	39	40	41	...	235	236	237	238	239	240	1
63	0	1	...	39	40	41	...	235	236	237	238	239	240	0
64	0	1	...	39	40	41	...	235	236	237	238	239	240	0
64	0	1	...	39	40	41	...	235	236	237	238	239	240	1

The final learning feature space consist of 32x7713 (32x(32x241)+1). In other words, it is identical to the input vector plus the ground truth.

4.2.2.2 LA Processing Pipeline

The events received by the DLT are coming from BSL facilities transmitted by the BMS. The events are sent one by one and unprocessed. The processing and data preparation are done by the LA. In this chapter, we present the final set of processing steps done by the LA. In the processing scheme followed by the LA

Table 7: End-to-end state machine, intra-factory BSL-2


Start point
New sensor reading
phenomenonTime == current phenomenonTime
No
phenomenonTime >= current phenomenonTime
No
Skip sensor read
Old sensor reading
Yes
Initialize the whole buffer of 242 with zeroes
Drop whole buffer
Matrix of 32 x 241/2 to be flattened
Append buffer to matrix
From position 0 to 39
Add sensor reading in the buffer Update current phenomenonTime
Length of sensors' reading == 40
Yes
No
Shape of the matrix == 32 x 241/2
Yes
Send matrix to DLT
No
Wait
From position 40 to 235
Add event reading in the buffer
Huge gap between event and sensors
No
phenomenonTime +10 min >= current phenomenonTime
Yes
The current phenomenonTime is shared between events and sensors
Start point
New event reading
Yes
phenomenonTime >= current phenomenonTime
No
Skip event read
Old event reading

The LA implements this logic in steps using EPL queries for the Esper engine. First, the LA collects readings from 40 sensors with following query:

```
insert into
       sensors
select
       Arrays.asList(
              s0.result, s1.result, …, s38.result, s39.result
       ) as readings
from
       pattern[
              every (
                 s0=Observation(datastream.id='ds_2-0')->
                 s1=Observation(datastream.id='ds_3-0')->
                     …
                 s38=Observation(datastream.id='ds_2-55')->
                 s39=Observation(datastream.id='ds_3-55')
                     )
              ]
```

In similar manner, the 5 new COMPOSITION acoustic sensor readings are collected.

```
insert into
       acoustics
select
       Arrays.asList(s0.result, s1.result, s2.result, s3.result, s4.result) as acousticReadings
from
       pattern[
              every (
                            s0=Observation(
                            datastream.id='ds_5-1')->
                            s1=Observation(datastream.id='ds_5-2')->
                            s2=Observation(datastream.id='ds_5-3')->
                            s3=Observation(datastream.id='ds_5-4')->
                            s4=Observation(datastream.id='ds_5-5')
)
              ]
```

Then the warnings are collected. This time each waring is collected individually in a vector of 196 elements. This means each warning will be inserted in a vector of zeros. The position depends on the type of events according to the DLT feature space. The query looks as following:

```
insert into
       warnings
select
       Tools.singleFillUp(*, 40, 196).selectFrom(i=> i.result) as warningFlags
from
       Observation(datastream.id.toString() like 'ds_6-%')
```

Now the LA can start building the input vector (32x7712 elements). This will be done in two steps in order to reuse it for the learning vector. First, we built a single element:

```
insert into
       buffer
select
       Tools.addAll(
              Tools.addAll(s.readings, Tools.ifNullReplaceList(w.warningFlags, 40, 196,'6')),
              Tools.ifNullReplaceList(a.acousticReadings, 236, 5, '5')
       ) as input
from
       sensors#lastevent() as s, warnings#lastevent() as w, acoustics#lastevent() as a
```

Now we build the 32 matrix:

```
insert into
       predicting
select
       Tools.flatArrayOfListToList(window(input)) as input
from
       buffer#length(32) having count(*) = 32
```

Now we will construct the Ground Truth as explained in the last chapter. First we collect the Hi warning alerts in 30 minutes:

```
insert into
       gtIntHiW
select
       window(intResult).where(i=> i > 0).countOf() as hiWarnings
from
       Observation(datastream.id.toString() in (
       'ds_6-58','ds_6-59','ds_6-60','ds_6-61','ds_6-62','ds_6-63','ds_6-64','ds_6-65','ds_6-
       66','ds_6-67','ds_6-68','ds_6-69','ds_6-70','ds_6-71','ds_6-72')
       )#time(30 min)
```

Now evaluate if there is any hi deviation in the last 30 min.

```
insert into
       gtIntHiD
select
       window(intResult).anyOf(i=> i > 0) as hiDeviation
from
       Observation(datastream.id.toString() in ('ds_6-142','ds_6-143',…,'ds_6-161','ds_6-
162'))#time(30 min)
```

Then we build the ground truth by checking if have been either 3 hi warnings or, 1 hi warning and a hi deviation in the last 30 min, We do this with the two following queries

```
insert into
       gtbuff
```

```
select
       1 as label
from
       gtIntHiW#lastevent() HiW, gtIntHiD#lastevent() HiD
where
       ((HiW.hiWarnings >= 3) or (HiW.hiWarnings > 1 and HiD.hiDeviation))

insert into
       gtbuff
select
       0 as label
from
       gtIntHiW#lastevent() HiW, gtIntHiD#lastevent() HiD
where
       not ((HiW.hiWarnings >= 3) or (HiW.hiWarnings > 1 and HiD.hiDeviation))
```

Finally, we build one element of 7713 of the 32x7713 for the learning vector/matrix:

```
insert into
       learning
select
       Tools.addAll(buf.input,t.label) as input, t.label as gt
from buffer#lastevent() as buf, gtbuff#lastevent() as t
```

We create the 32x7713 matrix for learning and use it as learning query. This query feed the DLT at the other end:

```
select
       window(input) as input, last(gt) as gt
from
       learning#length(32) having count(*) = 32
```

And we use the predict vector as deployment query. This query uses the DLT to do the prediction:

```
select
       CEML.batchPredictUsing('<id>', input) as result
from
       predicting#lastevent()
```

4.2.3 Testing/Benchmarking

In order to evaluate if the agent can handle the event rate needed as well to evaluate the possibility of the agent to scale; we made a series of tests / benchmarks.

The IoT Learning Agent will be tested / benchmarked considering the following aspects:

1. Load tests:
    a. Incrementing load test: Gradually increasing the number of events
    b. Burst load test: Simulating sudden bursts of events

Event fusion: Dynamically increase the amount of LA events/streams from different sources which a statement put together applying an operation

4.2.3.1 Burst Test A:

The test is design to test and discover the behaviour of the IoT Learning Agent (LA) in case of a burst of incoming events in a reasonable period of time.

4.2.3.1.1 Deployment Characteristics

Deployment Description: 3 Virtual machines (VM):

Table 8: Specs of the type of hosts

Types	CPU	Memory	NIC
A	Intel(R) Xeon(R) CPU E5-2670 v3 @ 2.30GHz cache 61440 KB	2.0 GB	10000baseT/Full
B	Intel(R) Xeon(R) CPU E5-2670 v3 @ 2.30GHz cache 61440 KB	4.0 GB	10000baseT/Full

Table 9: Hosts by type

Name	Type
Producer	A
Broker	B
LA	A

The data producer and consumer have the same hardware characteristics to test the performance in symmetric conditions. In this manner, the performance of capturing, handling, processing, and forwarding messages vs just generating and sending events are tested in equal hardware conditions.

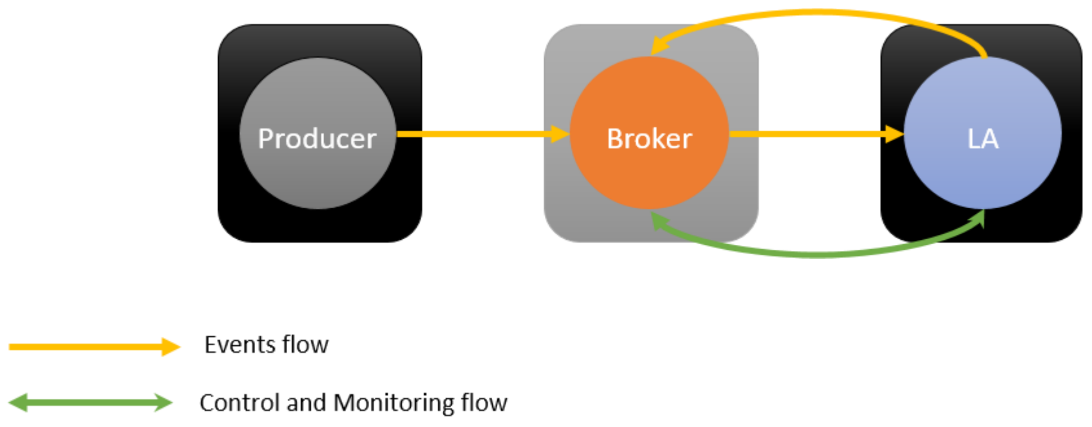


Figure 7: Deployment Sketch

4.2.3.1.2 Test Implementation

- In the Producer VM resides 10 clients producing as much load they can produce. The Broker VM resides in the MQTT Mosquitto Broker. Finally, in the LA VM the LA service is deployed. The test started when one LA statement is loaded in the LA service. The statement is a simple one (i.e. just arithmetical operations using the events payload). Afterwards, the producer clients generate the load for 15 minutes.

4.2.3.1.3 Results

Table 10: Summary of the test

Total load generated	Total load processed	Total average processed
104871868 messages (msg)	79425813 msg	82821.49426 msg/s
Total load generated	Total load processed	Total average processed

Table 11: Results per host

Host	Average	Maximum
	CPU Rate	
Broker*	84.80%*	95.35%
LA	169.60%	190.7%
	Main Memory	
Broker	52.16MB	95 MB
LA	35.96MB	37.9MB

*The CPU is measured in regards with the theoretical maximum can be reached by the process. i.e. the Brokers can just take a single core, while the LAs can take all of them. Therefore, the brokers percentage is measure using 1 core as base, while LA used 2 cores as base.

4.2.3.1.4 Analysis

The test was performed until one of the three sides reached its limit. In this test, the producer used an average of 97.7% (both cores) and was unable to generate more load. This means that, in equal conditions, the LA could outperform the producers.

The processed rate shows how many messages were processed at the end of the experiment. The remaining events were at the queue (either at the broker or at LA). This means that the system was stacking events faster than they could be processed. This happened with a rate of 2.4 MB/min. With this rate, the LA would overflow its memory in about 13.64 hours of sustained burst. The mosquito broker increased the memory in a faster rate 3.48 MB/min, but the broker VM had the double amount of memory (4 instead of 2 GB). This means first, the broker VM needed 18.71 hours before its collapse. Secondly, the broker queue was increasing faster than the LA. This indicates that in such deployments, the broker will demand more memory.

The broker shows better performance than the LA at the first glance. However, Mosquitto is a mono-thread service, this means, that the service can just leverage form one CPU. Therefore, the performance measured in all CPU is unimportant. Even more, the increase of cores will not improve performance. In contrast, the LA is a multi-thread service. Hence, the LA will probably scale better as the Mosquitto, due to it is easier to increase cores as the clock speed-rate on a CPU. This had been already observed in other experiments, where the load increased over the load used in this experiment.

4.2.3.1.5 Scalability Perspective

The test shows that the effects of a heavy load in the system affect different on other components. This point out that the first bottleneck in the processing chain is the broker. Furtherer tests confirm that. The broker is the first component to be overloaded. This can be addressed by deploying more brokers, which balance load between them. It is important that each broker is deployed in a single core machine so there is no waste of CPUs. On the other hand, the LA cannot hold more load over the one presented in this experiment. Nevertheless, the LA did not crash when the CPU reached its limits in contrast to Mosquitto. But it will when the LA reached its physical memory limit. Increasing the number of cores will further increase the performance of the LA. Additionally, deploying multiple LA on different machines will also increase the performance of the system, similar to the deployment of multiple brokers.

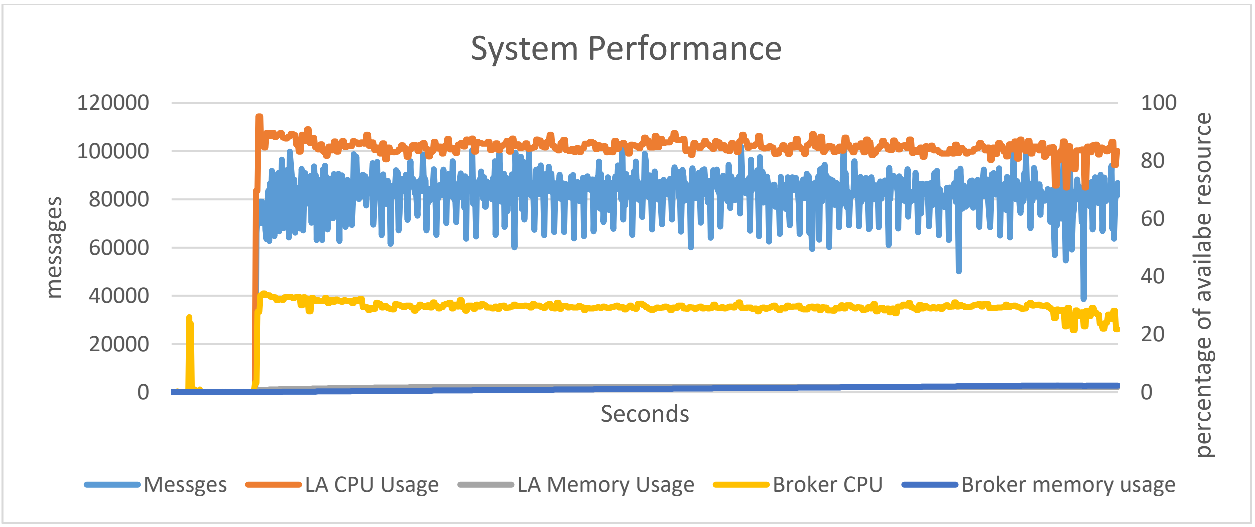


Figure 8: Plot of the Performance of Test 1

4.2.3.2 Burst Test B

The test is design to show the horizontal data processing capability by increasing the amount of LA in the deployment.

4.2.3.2.1 Deployment Characteristics

Deployment Description: 10 hosts, 9 VM and one a physical machine. There are three kind of machines:

Table 12: Type of hosts for test 2

Type	CPU	Memory	NIC	Hardware
A	2 cores Intel(R) Xeon(R) CPU E5-2670 v3 2.30GHz cache 61440 KB	2.0 GB	10000baseT/Full	Virtual
B	Intel(R) Xeon(R) CPU E5-2670 v3 2.30GHz cache 61440 KB	4.0 GB	10000baseT/Full	Virtual
M	4 cores: Intel(R) Xeon(R) CPU E5-2603 0 1.80GHz cache 61440 KB	4.0 GB	1000baseT/Full	Physical

Table 13: Host by types in test 2

Name	Type
Producer 1, 2 (p1, p2)	A
Broker010 and BrokerControl	
LA000, LA010 and LA100	
Broker000 and Broker001	B
Broker100	M

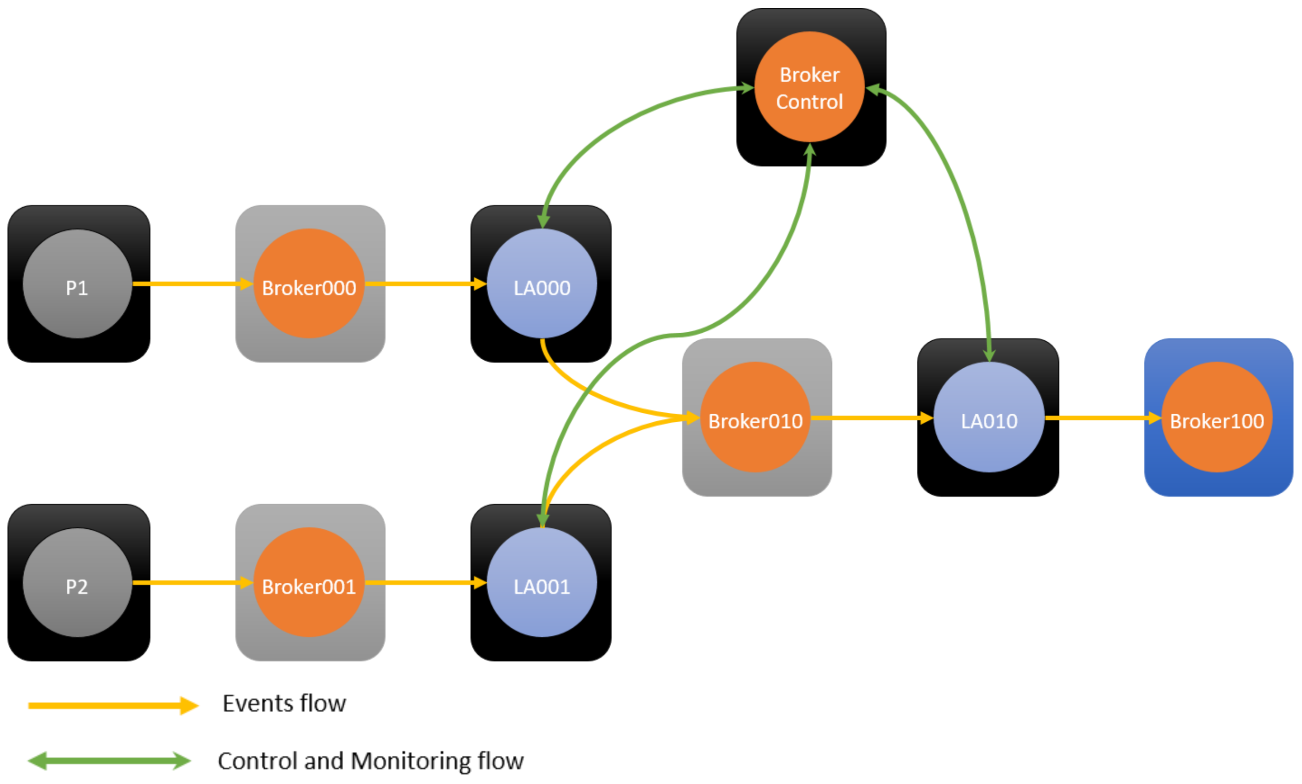


Figure 9: Deployment Sketch of Test 2

4.2.3.2.2 Test Implementation

In this test, we have two parallel data channels. The first, started by the producer one (p1) VM and the second started by producer (p2) VM. Both channels are merged by LA[0,1,0] VM, for thereafter follow a single data channel. In each Producer VMs resides 10 clients producing up to 100K msg/s (msg = messages). The p1 (producer one) publish the load on Broker[0,0,0], while p2 publish on Broker[0,0,1]. Each broker machine has a MQTT Mosquitto Broker. The broker machines forward the messages to the corresponding LA machines, Broker[0,0,0] to LA[0,0,0] and Broker[0,0,1] to LA[0,0,1]. Both LA[0,0,***i***] aggregates the data and publish their results on Broker[0,1,0]. The LA[0,1,0] aggregates the results of both (LA[0,0,1], LA[0,0,0]) and publish its results on Broker[1,0,0]. On the other hand, BrokerControl is there for deploying the LA statements and monitoring all LAs simultaneously. This means, that the statement is publish once and multi-deployed in parallel in all LAs. Finally, for the scope of this test, we will concentrate in the first aggregation layer hosts (Broker[0,0,0],Broker[0,0,1],LA[0,0,0], LA[0,0,1]), while the second is just to add the data in real-time without interfere with the test.

4.2.3.2.3 Results

Table 14: Summary of the test

	Load generated		Load processed		CPU rate
	Total (msg)	Avg (msg/s)	Total (msg)	Avg (msg/s)	Avg (%)
Channel 1	62520137	60405.92947	55986276	54093.02029	90.19
Channel 2	81452259	78697.83478	73863449	71365.65121	90.68
Channel 1&2	143972396	69551.88213	129091627	124726.2097	

Table 15: Results per host

Host	Average	Maximum
	CPU Rate (%)	
Broker000	58.65	100
Broker001	48.33	58.3
Broker010	1.42	1.6
Broker100	0.01	0.3
LA000	77.27	98.85
LA001	57.22	98.7
LA010	0.57	59.35
Channel 1 (P1)	67.96	99.43
Channel 2 (P2)	52.78	78.5
	Main Memory (%)	
Broker000	77.00	95.2
Broker001	1.42	1.6
Broker010	0.2	0.2
Broker100	<0.01	<0.01
LA000	28.73	30.2
LA001	28.89	30.4
LA010	5.75	5.9
P1	52.5	62.7
P2	15.16	16

*The CPU is measured in regards to the theoretical maximum can be reached by the process. i.e. the Brokers can just take a single core, while the LAs can take all of them. Therefore, the brokers percentage is measure using 1 core as base, while LA used 2 cores as base

4.2.3.2.4 Analysis

Before analysing the results, there is a need of take some times describe a non-symmetrical results obtained between both data channels. While both parallel data channels are symmetric in virtual hardware, they produce different results. While p2 had problems to go over the 80K msg/s, p2 was constantly close to the 100K msg/s threshold. This produced uneven load in the broker. The data channel 1 (started by p1), loaded till the limit the CPU limit of the Broker[0,0,0]. This produced that the Broker[0,0,0] needed to queue much more msg till it was close to depleted all memory. On the channel 2, the data was more balanced, the load did not overload the Broker[0,0,1] allowing better results. This effect produced a difference of 18M msg in total between both channels. This difference it is probably produced due to the whole deployment is made in virtualized hardware running in the same cluster.

Theoretically, each channel can be seen as independent simple load test. Nevertheless, the performance of both channels were not equal and both less than the first test. This probably was also problem made by the virtualized hardware running in the same machine.

In case of channel 2 (Broker[0,0,1] and LA[0,0,1]) the resources where used in a moderated manner achieving a higher processing rate than channel 1 with less usage of CPU and memory resources (52.78 CPU% and 15.16 MEM%). Additionally, the components in channel 2 do not show signs of being overloaded. On the other hand channel 1, reached the maximum capacity by reaching the maximum usage of the CPU of Broker[0,0,0] almost constantly. This triggered the overused of memory, for finally the reduction of the performance. This shows again that the Mosquitto broker is the weakest component regarding performance.

However, even than the capacity of the broker of channel 1 reached its maximum capacity after 6 min the test started, the system was able to provide stable results for another 11 more minutes. This shows the robustness of the system in a heavy load situation.

4.2.3.2.5 Scalability Perspective

Therefore, the merging of both channels provides a geometrical increment of aggregation computational capacity to the overall system. In other word, the system scale in a horizontal manner. In the first test the system was able to process 82K msg/s with maximum burst of 102 K msg/s, close to channel 2 with 71K msg/s and with maximum burst of 99K msg/s in channel 1. Furthermore, the combine deployment got an average 124K msg/s with maximum burst of 194K msg/s. This is an increment of 152% on average and 190% for the maximum burst. Moreover, there was an improvement of 15% in the processing rate.

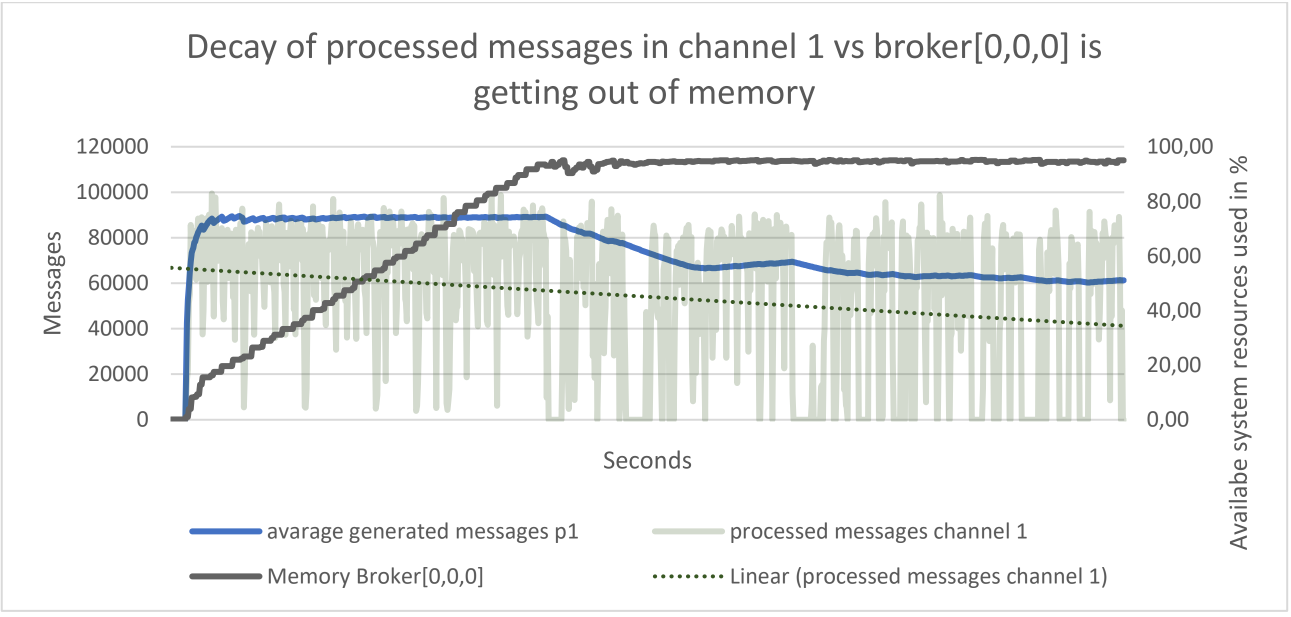


Figure 10: Plot of the Performance Results of the Test 2

4.3 Descriptive Big Data Analytics

Apart from a supervised identification of future failure states, for use cases BSL-2 and KLE-1 a detailed descriptive analysis was performed on historical data. On the one hand, this was done in order to analyse the data beforehand to identify prevailing anomalies in the data, on the other hand, to create additional statistical models for a stream-based online analysis based on the extracted knowledge.

4.3.1 Configuration-Specific Anomaly Detection Based on Nearest-Neighbour Similarity Search

One important fact to consider while trying to find anomalies in time series data is the inclusion of configuration changes. While a sudden change in the data behaviour might look suspicious for observations with respect to the same configuration parameters of a machine, it is common after a reconfiguration and therefore potentially no anomaly. In use case BSL-2, machine configurations are set on-the-fly. In the following, we therefore propose a multi-step approach (4.3.1.1-4.3.1.4) that was used in order to identify potential anomalies.

4.3.1.1 Configuration-Based Clustering

Given the events of a change in configuration, in a first stage the data was clustered based on each configuration. This was done to avoid a comparison of observation behaviours stemming from disjunctive machine settings. All obtained segments are consequently mapped to a specific configuration.

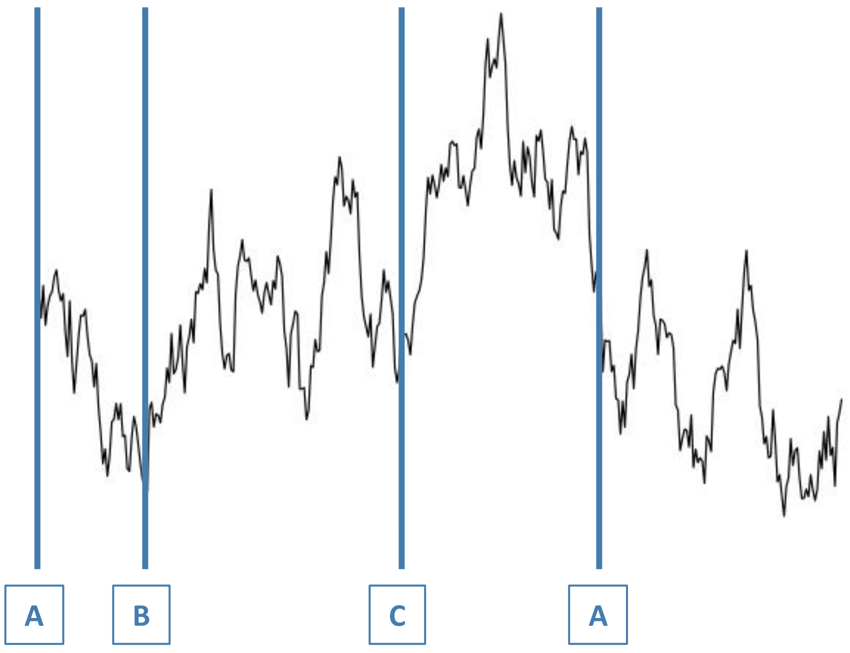


Figure 11: Segmentation of an Observation Given Exemplary Configuration Events (A, B and C)

4.3.1.2 Segmentation

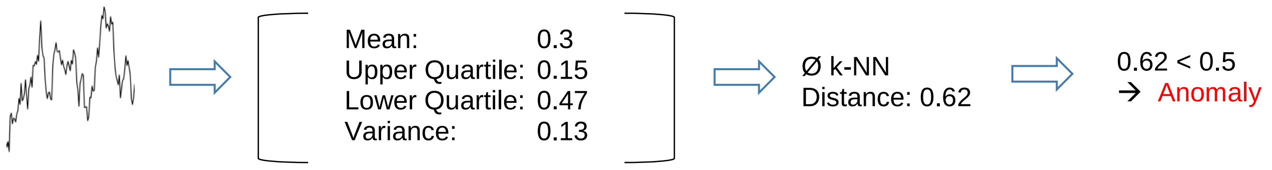
In a next stage for each group of data, any slice of the time series is further segmented to allow for a uniform synchronisation. While this for example can be done based on events or change point-detection models when no underlying process semantic is known, we used a time-interval to reflect the daily behaviour. We further segmented the data, when such an interval was not fully captured (e.g. due to missing data) and ignored segments that were below a minimal predefined size. In addition, we truncated the segments if they were immediately following a configuration change, since for most processes, like an adaption in the set temperature, the observation variable slowly adapts to the defined setting.

4.3.1.3 Feature Extraction

Given the output from 4.3.1.2, each segment was further described by a set of predefined statistical metrics. While rather complex statistical metrics or parameterized approximations by (non-)linear fittings were possible choices in order to describe individual segments, we decided to initially start with a set of well-known statistical measures as the mean, the variance and the quartiles. The reason for this choice was the computational efficiency and the avoidance of over-fitted descriptions.

4.3.1.4 Similarity Measurement

To identify anomalies in a final stage, each segment represented by a d-dimensional feature vector was compared to its k nearest neighbours regarding the average of the normalized Euclidean distance. We chose the k nearest neighbours in order to allow for different process behaviours in the same configuration group. While a segment is compared to its k nearest neighbours, segments that have no similarity to a suitable amount of other segments are considered as anomalies. The subsequent figure exemplarily shows the classification of segments with a threshold of 0.5.



4.3.1.5 Evaluation

We evaluated our approach using the historical sensor data from BSL-2 and a maintenance log file. Illustrated in the following plot showing the power consumption of a fan, the red vertical line indicates an exchange of the fan. While the sensor measurements are shown over time, different colours indicate different machine configurations. For instance, among other anomalies, the data within the black box prior to the fan exchange could successfully be identified as one anomaly.

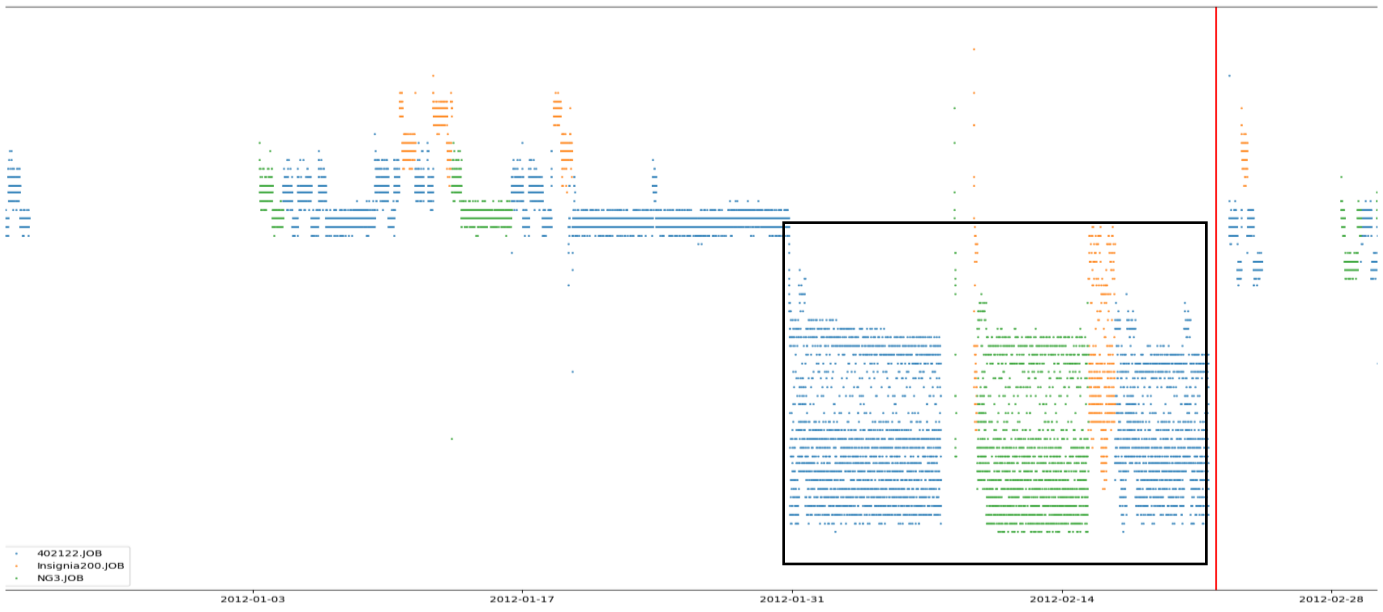


Figure 12: Evaluation of the Anomaly Analysis

4.3.2 Descriptive Analysis of Sensor Data

In use case KLE-1, a vibration sensor was attached to the motor of BOSSI machine. Additionally for test purposes, a faulty motor was implemented for a specific amount of time. The goal was to determine the differences in behaviour, to be able to describe the valid behaviour of the machine such that all other future behaviours can consequently be identified as anomaly.

Since the data has been recorded with a high sampling rate, in a first step the data was sampled down using an aggregation function (mean). After grouping the data on a daily basis, we extracted statistical features. This allowed for a distributive description of the data for each day.

In order to train the valid behaviour a multivariate Gaussian was fitted using the data prior to the motor exchange. Only defining the valid behaviour allows for a future anomaly detection. A classification also concerning the invalid behaviour, would only capture a specific invalid behaviour of the motor. The subsequent figure shows the normalized likelihood of the data, given the fitted model. As one can see, although the likelihood is sometimes low in advance to the attachment of the nearly-defect motor, the model is still able to clearly distinguish between a valid and an invalid behaviour of the motor.

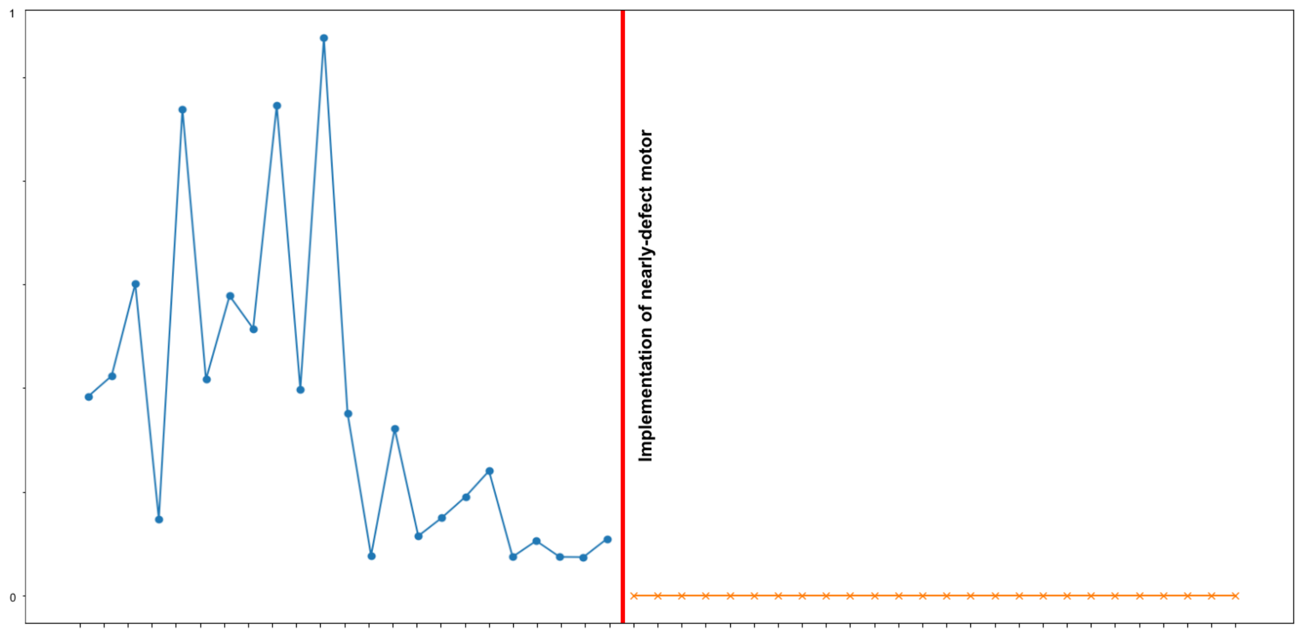


Figure 13: Likelihood of the Data, Given the Fitted Model for the Valid Behaviour

5. Big Data Visual Analytics

This chapter presents the results of Task 5.1 related to Visual Analytics tools and techniques. In the first stages of the project, the research was focused on the analysis of the existing visualisation techniques, the specification of Visual Analytics tool architecture in overall system architecture and the creation of a high-level view of the Visual Analytics tool. Afterwards, the implementation phase of the Visual Analytics tool is taken place and it is presented in this chapter as well.

The COMPOSITION Visual Analytics tool should be a web-based tool in order to follow the design guidelines and the philosophy of the project. The DSS is the tool that offers users interfaces. Therefore, the implemented Visual Analytics tool should be connected with it and should offer functionalities that are not supported from DSS. Moreover, as the DSS is focused on the visualization and decision support of the two main industrial pilots, the Visual Analytics tool aims to enable the visualization and decision making over the Marketplace and the inter-factory scenarios. This means that it is focused more on big data analytics related to ELDIA pilot.

The interactive Visual Analytics tool should be able to import data from Big Data Analytics tools and based on the input data it will apply visual analytics techniques and will present the output to the users as graphical representations. The Visual Analytics tool will provide the ability to manufacturers/end-users to evaluate the analytics and predictions results and enhance the decision-making.

5.1 Visualization Techniques Analysis

5.1.1 Overview of Visualization Techniques Categories

A thorough analysis of visualization techniques has been contacted at the first stage of this task. A summarize of the results of this analysis is presented in this section.

Based on the aforementioned analysis we have distinguished the following categories of visualization techniques:

- **Standard 1D to 3D graphics** are widely used techniques for the visualization of data models. They are also used to view the frequency distribution of an attribute or to view the estimation of the certainty about a hypothesis. The most representative examples of Standard 1D to 3D graphics techniques are the scatter and contour plots, the line and stack graphs, the histograms and the pie and bar charts. Furthermore, combinations of the previous examples such as histogram with lines and bars with lines are also wide used.
- **Graph-based or hierarchical techniques** are used in numerous applications in the field of information visualization. These are structures that represent data relationships. To achieve this representation node-link diagrams are used. Examples of graph-based or hierarchical techniques are graphs, three maps and cone trees.
- **Pixel oriented** techniques map each data value to a coloured pixel and present the data values on the display screen in separate windows. Each window presents the data values belonging to one attribute. Query-independent technique is an example of Pixel oriented techniques. In this technique, the data is sorted according to some attributes and a screen-filling pattern is used in order to arrange the data values on the display.
- **Geometric techniques** are also commonly used for the visualization of information. These techniques provide an overview of all attributes by mapping the multidimensional data into a two-dimensional plane. A scatter plot matrix is an example of Geometric technique. This matrix shows relationships and dependencies among several variables by taken a pair of them at time. A Kiviat Diagram is another well-known example of the Geometric techniques' visualization category. A Kiviat diagram is composed of axes extending from a central point and represents how multiple items compare when they are evaluated against more than two variables.

5.1.2 Parameters Define Visualization Technique's Selection

In order to select the most appropriate visualization technique to use, a set of parameters should be considered:

- **Data type** is a basic criterion for choosing a visualization technique. Moreover, the Data type parameter can be used as criterion for visualization techniques' classification. Based on the current status of the project the following data types will be into consideration:
  - continuous or discrete quantitative data

  - data represented from time series
  - multi-dimensional data
  - and spatial data.

- **Task type** is another determining factor for classifying visualization techniques. The Task type parameter is related to the activities that an end-user is able to perform. These activities refer to visualization capabilities according to his goals. Generally, the main functionalities that a user requires from a visualization tool are the following:
  - Comparison of attributes or correlation among the attributes
  - Data Overview
  - Identification of possible patterns or clusters
  - Outliers detection.

- **Dimensionality** is a parameter to be observed in the data before applying a visualization technique. It refers to the number of data attributes. Dimensionality is classified based on the number of the attributes in three basic categories:
  - Small (up to 4 attributes)
  - Medium (5 to 9 attributes)
  - High (more than 10 attributes).

- **Scalability** of data is a characteristic should be considered for the selection of visualization techniques. The Scalability of data is classified in three categories:
  - Small (10 to $10^{2)}$
  - Medium (10^3 to 10^5)
  - High (10^6 to 10^7).

5.1.3 Brief Analysis of General Visualization Tools

The COMPOSITION visual analytics tool will support the analysis of large volumes of data, related with maintenance decision support, recyclables materials management and predictive maintenance processes. These functions will allow the visualization of possible correlations among predictive maintenance features, probabilities of upcoming types of machine faults, graphs of recyclables materials management, tools for decision support based on generative algorithms etc. Because of the fact that there are different needs for each pilot use case, despite general visualizations (line plots, pies charts, scatter plots, bar charts (single and with line plots), histograms (single and with line plots), bubble charts, Kiviat diagrams etc.), the visual analytics will provide targeted analysis with tools specially crafted for each use case

A brief description of some general visualization tools is given below:

- **Line plots**: A line plot is most useful in displaying data or information that changes continuously over time
- **Pie charts**: A pie chart is a circular chart divided into sectors, illustrating proportions
- **Scatter Plots**: A scatter plot is a type of mathematical diagram using Cartesian coordinates to display values for two variables for a set of data. The data is displayed as a collection of points, each having the value of one variable determining the position on the horizontal axis and the value of the other variable determining the position on the vertical axis
- **Bar charts (single)**: A bar chart is a chart with rectangular bars with lengths proportional to the values that they represent. The bars can be plotted vertically or horizontally
- **Bar charts with line plots**: Combination of bar charts and line plots
- **Histograms (single)**: A histogram is a graphical representation showing a visual impression of the distribution of data
- **Histograms with line plots**: Combination of histogram and line plots
- **Bubble charts**: A bubble chart is a type of chart that displays three dimensions of data
- **Kiviat diagrams**: A Kiviat diagram is composed of axes extending from a central point. Each axis represents a data category. Each axis is scaled according to certain parameter values. It is used to

graphically represent, on a single diagram, how multiple items compare when they are evaluated against more than two variables.

5.2 Overview of Visual Analytics Tool Architecture and Development

The COMPOSITION Visual Analytics (VA) tool imports data from Big Data Analytics tools, Simulation and forecasting tool and Deep Learning Toolkit. The VA offers an interactive user interface for the SFT and DLT analyses and forecasts and applies visual analytics techniques in order to present the output to the end-users as graphical representations. The Visual Analytics tool will provide the ability to manufacturers/end-users to evaluate analytic tools 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] framework, which takes care of all the basic and content management features of the application. Moreover, the MongoDB[2] is used for the tool's storing requirements.

The VA tool supports different types of analysis and their corresponding method analytics:

- Forecasting:
  - Time series Forecasting
  - Time series Persistence Forecasting
  - Markov Chain Prediction
- Fill Level Analysis:
  - Trend Analysis
- Optimal Routes:
  - Optimal Routes Calculator
- Statistical Analysis:
  - Statistical Analysis
- Price Forecasting
  - Deep Learning Analysis.

Depending on the type of analysis, data can be available by uploading properly formed external files. Method Analytics are defined by configuration files, stored in the file system. These configuration files include all parameters necessary for selecting and visualizing the results of a selected method, such as file uploading, available charts, etc.

Many different widgets and directives are offered from the VA tool. A wide variety of charts, pies, line charts, tables and time series representation etc. are available in the Visual Analytics tool as it adopts open source library for these purposes. So, the graphical representations are enabled by the use of the Chart.js [3]and D3.js [4]visualization libraries. These libraries are ideal for producing dynamic, interactive data visualizations in web browsers. In contrast to many other, they allow great control over the final visual results by using the widely implemented SVG, HTML5, and CSS standards.

The Visual Analytics Tool communicates with Analytics tools using MQTT and REST protocol as both of them are supported by the aforementioned tools. In particular, the Big Data Analytics, DLT and SFT outputs that contain analysis results transferred to the VA tool using these two protocols. After that, visualizations of these results are available to the end-users. Moreover, the user is able to demand further visualization and analysis results using the interactive interface. The Visual Analytics Tool will be integrated as a micro-frontend in the HMI framework of COMPOSITION as part of both DSS and Marketplace interfaces.

The next figureFigure 14 depicts a high-level conceptual architecture of the COMPOSITION Visual Analytics Platform with its main functionalities and dependences. The results of data analysis made by the components of COMPOSITION IIMS such as the Simulations and Forecasting Tool and the Deep Learning Toolkit are the main input of COMPOSITION VA Platform. The platform will have a support pre-processing component for data cleansing, transformation and normalization prior the use of data structures. Data cleaning techniques are applied to "clean" the data by filling in missing values, smoothing noisy data, identifying or removing outliers, and resolving inconsistencies.

[1] https://angularjs.org/
[2] https://www.mongodb.com/
[3] https://www.chartjs.org
[4] https://d3js.org/

The VA Platform will support several data structures (e.g. linear structures, graphs, tables, trees etc.) so that they can be accessed and used efficiently and effectively by the general visualization modules (line plots, pies charts, scatter plots, bar charts etc.) and VA modules (clustering, probabilistic techniques, adaptive heatmaps, graphs etc.). The COMPOSITION VA Platform will provide input to the Decision Support System to improve decision making for business scenarios. The input will be related to UC-KLE-1 and vibration sensor data analysis and for supply chain use cases: UC-KLE-4 and UC-ELDIA-1. The rest visualizations and interfaces of the project use cases will be supported from DSS and Marketplace interfaces as it was decided that these tools and interfaces provides the demanded simple of visualization perspective functionality. These functionalities will be presented in their corresponding deliverables.

The output of the Visual Analytics tool will be available to DSS and Marketplace interfaces in a design format compatible with the interfaces of these tools. Actually, these three tools, DSS, VA and Marketplace will share a common menu and will be offered as an integrated tool to end-user.

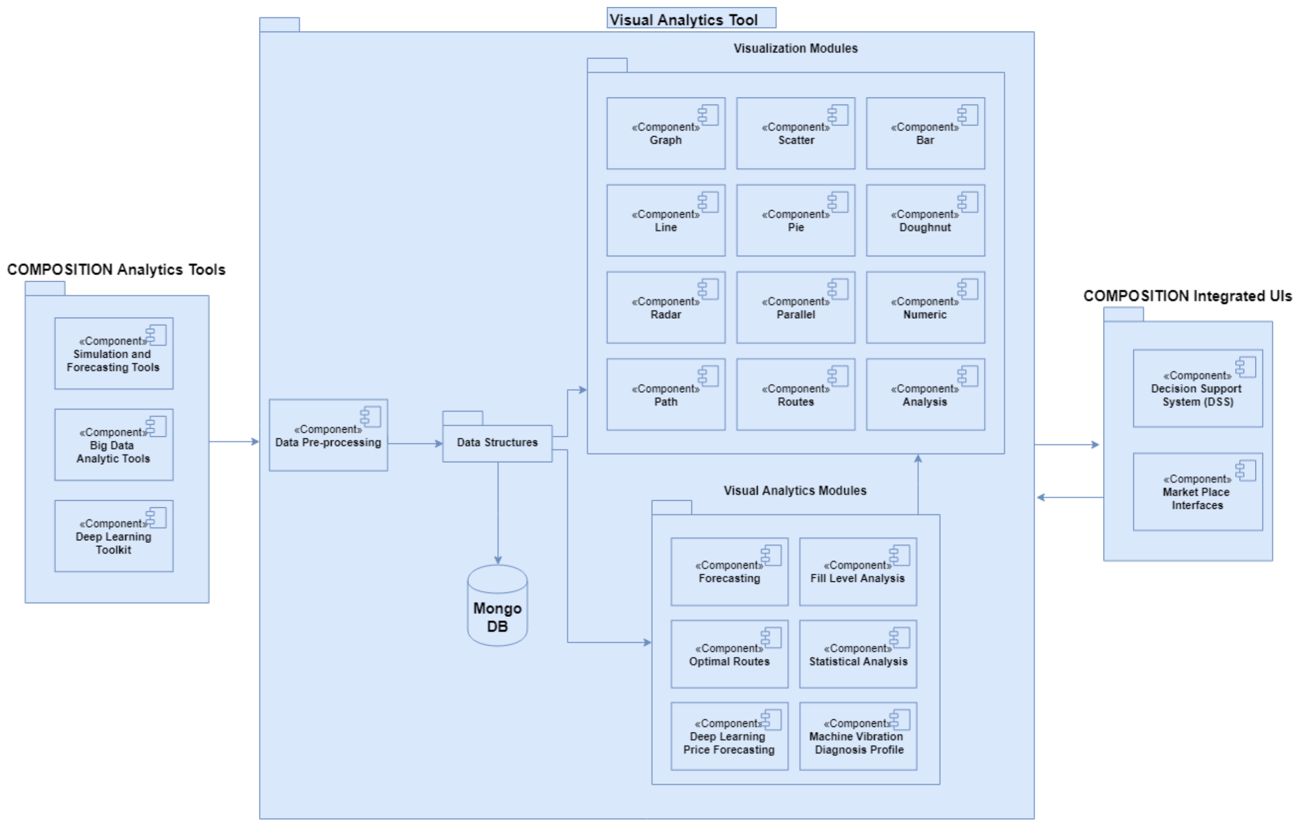

Visual Analytics Tool
Visualization Modules
«Component» Graph
«Component» Scatter
«Component» Bar
«Component» Line
«Component» Pie
«Component» Doughnut
«Component» Radar
«Component» Parallel
«Component» Numeric
«Component» Path
«Component» Routes
«Component» Analysis
COMPOSITION Analytics Tools
«Component» Simulation and Forecasting Tools
«Component» Big Data Analytic Tools
«Component» Deep Learning Toolkit
«Component» Data Pre-processing
Data Structures
Mongo DB
Visual Analytics Modules
«Component» Forecasting
«Component» Fill Level Analysis
«Component» Optimal Routes
«Component» Statistical Analysis
«Component» Deep Learning Price Forecasting
«Component» Machine Vibration Diagnosis Profile
COMPOSITION Integrated UIs
«Component» Decision Support System (DSS)
«Component» Market Place Interfaces

Figure 14: COMPOSITION Visual Analytics Platform Architecture

5.3 Visual Analytics Tools Functionality and Interfaces

The implemented Visual Analytics tool enable the end-user to select analytics category, to import dataset (if required) and then to select the analytics algorithm, which are available for the selected case or dataset. After that, the user is able to select a visualization module/graph from the list of the supported module for this analysis. The following figures describe gradually the user's interaction with the VA interfaces:

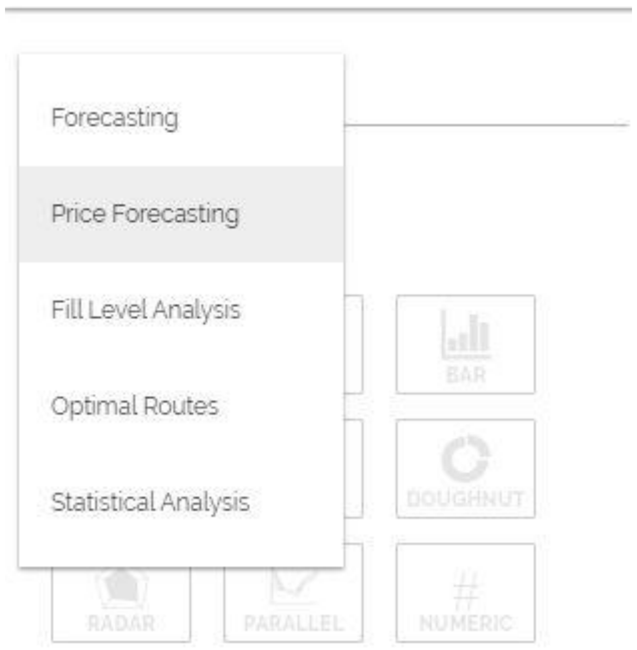


Figure 15: Selection of Data Analysis

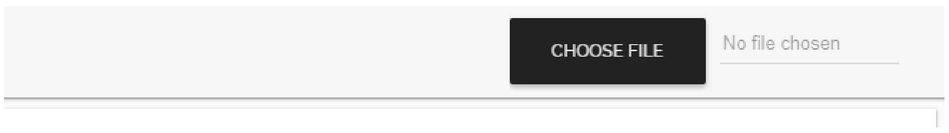


Figure 16: Selection of External Dataset (if any)

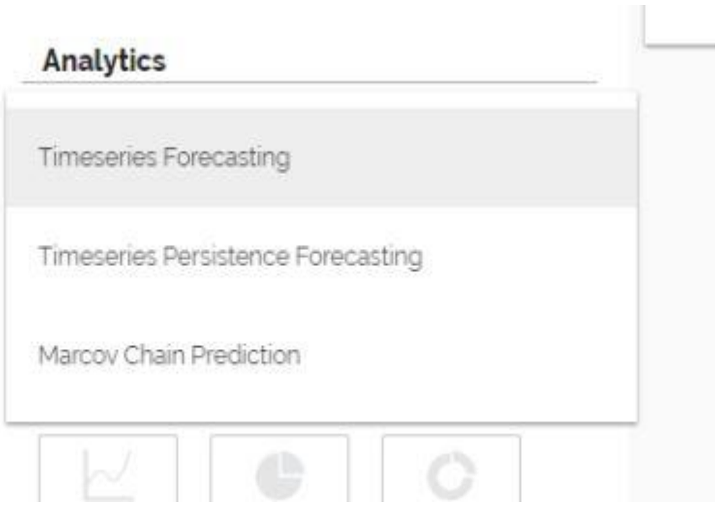


Figure 17: Selection of the Available Analytics Method

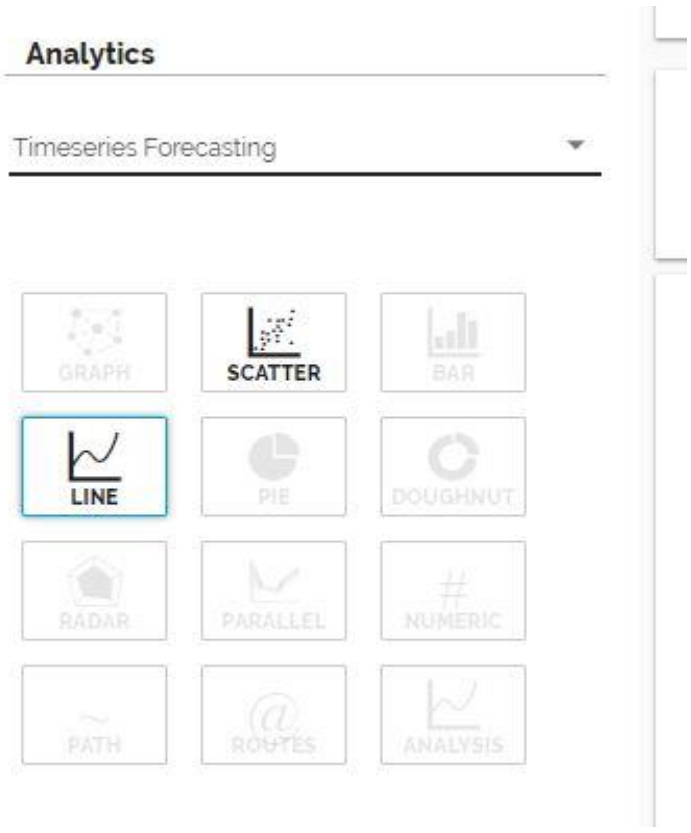
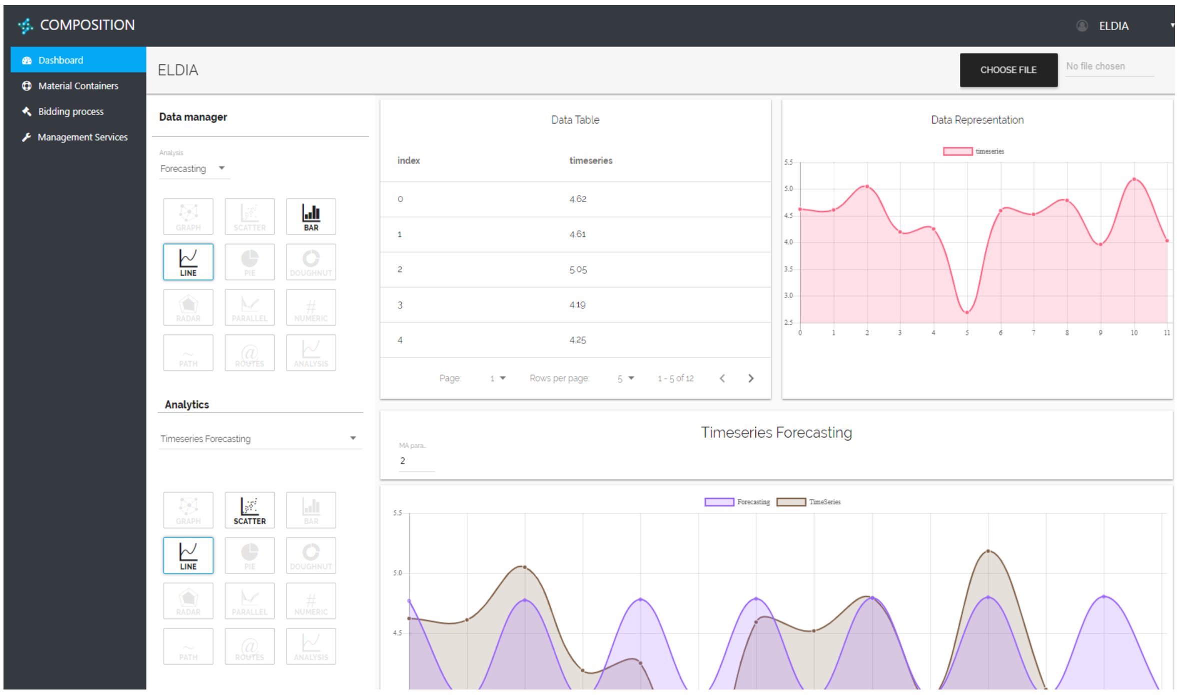


Figure 18: Selection of Available Chart

A complete overview of the COMPOSITION Visual Analytics tool user interface is presented in the following figure:



Data Table

index	timeseries
0	4.62
1	4.61
2	5.05
3	4.19
4	4.25

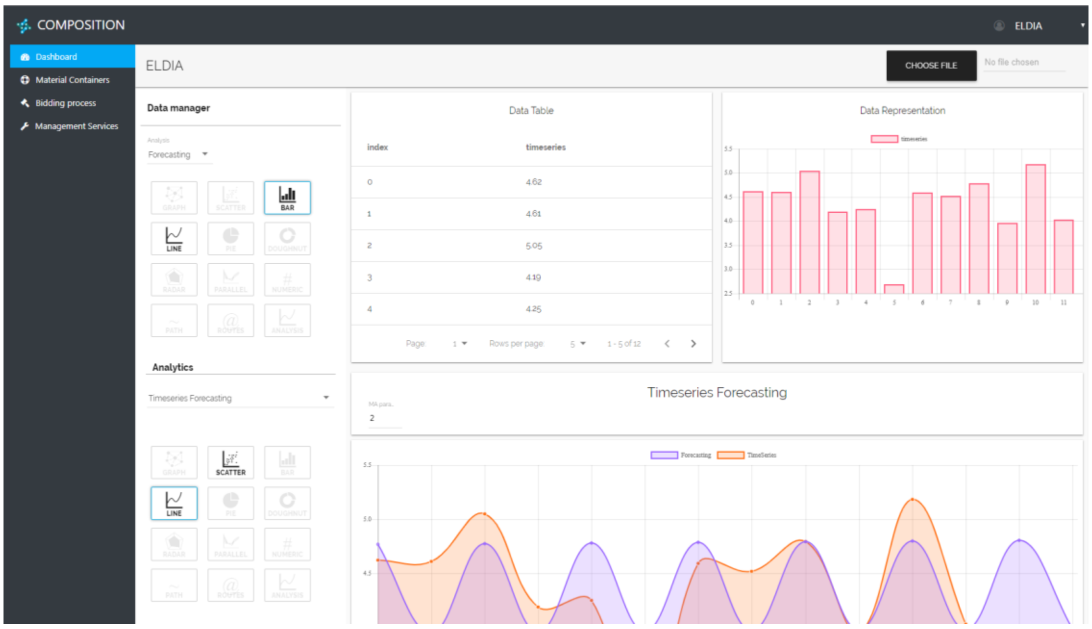
Page: 1 Rows per page: 5 1 - 5 of 12

Timeseries Forecasting

MA para.: 2

Figure 19: COMPOSITION Visual Analytics Tool

Some indicative examples of the visual analytics tool in accordance with the analytic methods are listed below in order to explain better the connection with the analytics and the interaction with the end user:

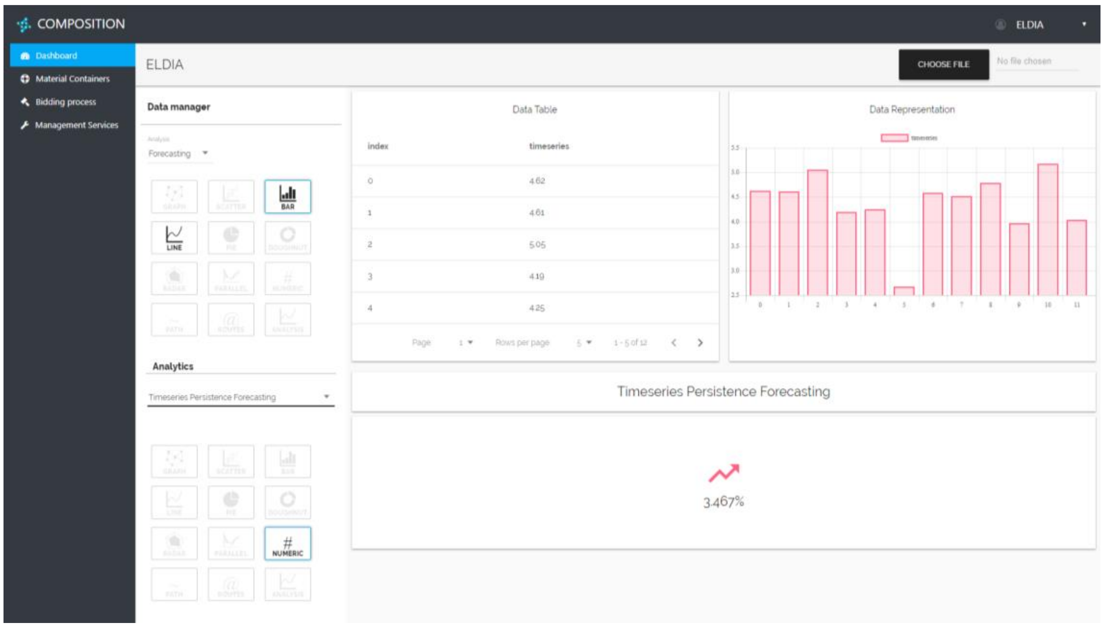


index	timeseries
0	4.62
1	4.61
2	5.05
3	4.19
4	4.25

Page 1 Rows per page 5 1 - 5 of 12

Figure 20: Time Series Forecasting Ribbon

In Figure 21, a time series forecasting methodology based on moving average (MA) time series models are presented. The end user has the ability to choose among five different parameters of a basic MA time series model so as to find the best prediction models for the imported data.



index	timeseries
0	4.62
1	4.61
2	5.05
3	4.19
4	4.25

Page 1 Rows per page 5 1 - 5 of 12

Figure 21: Time Series Persistence Forecasting Ribbon

In Figure 21, the methodology of persistence forecasting prediction model for time series. This methodology uses the value of previous time step so as to predict the expected outcome of the next time step. Unlike the previous case, (Time series forecasting) in this one the end user is not able to make any adjustments on the methodology

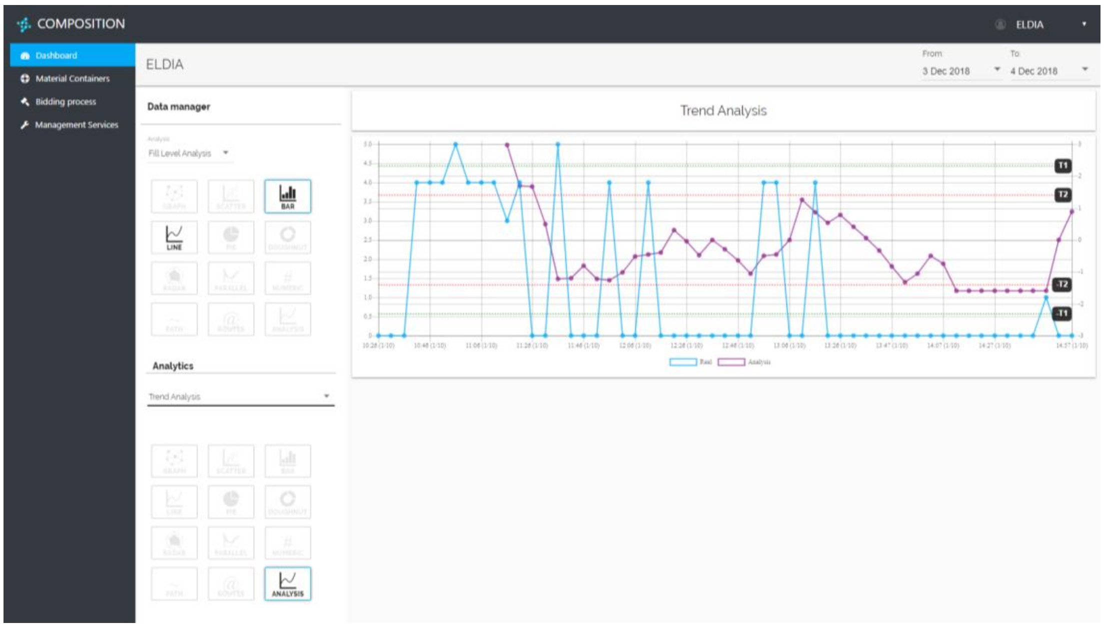
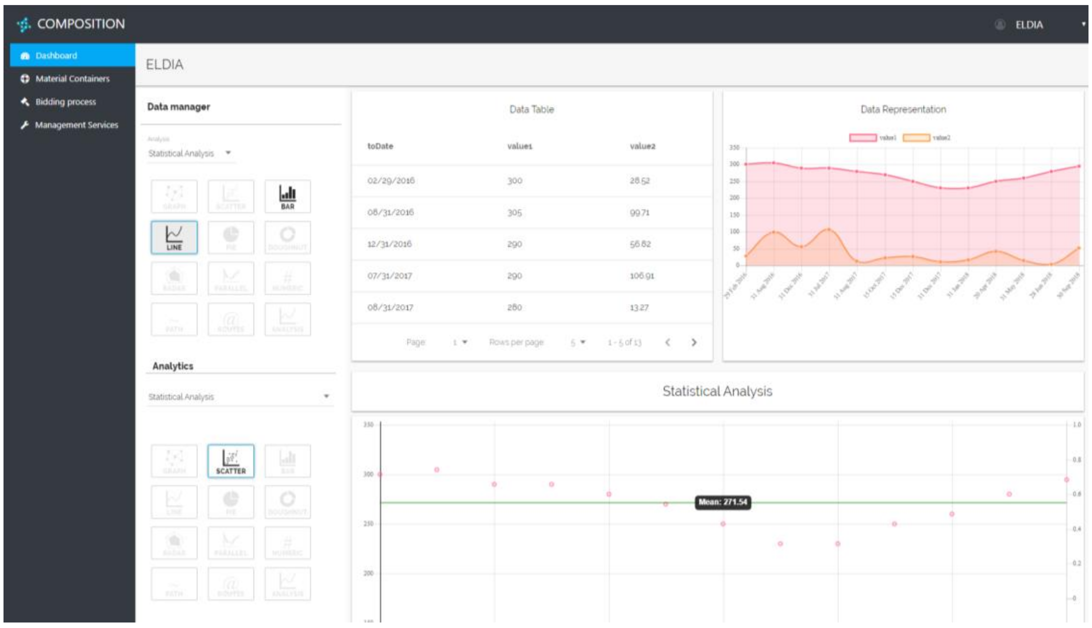


Figure 22: Trend Analysis Ribbon

In Figure 22, a visualization of Slope Statistic Profile methodology for time series linear trend analysis is presented. The end user has the ability to choose the dates of time series to be analysed and the sliding window size parameter, so as to test all available options and find the one that fits optimally on the selected time series.



Data Table

toDate	value1	value2
02/29/2016	300	28.52
08/31/2016	305	99.71
12/31/2016	290	56.82
07/31/2017	290	106.91
08/31/2017	280	13.27

Page 1 Rows per page 5 1 - 5 of 13

Figure 23: Basic Statistical Analysis Ribbon

In Figure 23, the end user has the ability to find a brief statistical analysis of the data to be analysed furtherly on other ribbons.

6. Conclusion

6.1 COMPOSITION Contribution and Technologies Used

6.1.1 Main Technologies Used

6.1.1.1 Siddhi/WSO2 CEP

Siddhi CEP is a lightweight, easy-to-use Open Source Complex Event Processing Engine (CEP) released as a Java Library under Apache Software License v2.0. Siddhi CEP processes events which are generated by various event sources, analyses them and notifies appropriate complex events according to the user specified queries. This project was initiated as a research project at University of Moratuwa, Sri Lanka, and now being improved by WSO2 Inc. The CEP engine has been rebranded as WSO2 CEP engine.

WSO2 Complex Event Processor (WSO2 CEP), aka Siddhi CEP, helps identify the most meaningful events and patterns from multiple data sources, analyse their impacts, and act on them in real time. 100% open source, it allows you a set up a more agile connected business, responding to urgent business situations with both speed and precision.

From enabling healthcare providers develop real-time clinical decision support systems, to monitoring transactions and tracking vehicles in smart cities, WSO2 CEP is highly performant, massively scalable, and offers significant savings in both cost and time.

WSO2 CEP is available on-premise, on top of WSO2 Private PaaS or on any public cloud such as Amazon AWS. It is an integral part of the WSO2 Analytics Platform, where it provides the real time analytics pipeline (WSO2, 2016), (Siddhi, 2016).

6.1.1.2 Esper

Esper is an open-source (GPL licenced) Java-based software for complex event processing (CEP) and event series analysis. It enables rapid development of applications that process large volumes of incoming messages or events, regardless of whether incoming messages are historical or real-time in nature.

Esper and Event Processing Language (EPL) provide a highly scalable, memory-efficient, in-memory computing, SQL-standard, minimal latency, real-time streaming-capable Big Data processing engine for historical data, or medium to high-velocity data and high-variety data.

6.1.1.3 MQTT

MQTT stands for Message Queueing Telemetry Transport. It is a publish/subscribe, extremely simple and lightweight messaging protocol, designed for constrained devices and low-bandwidth, high-latency or unreliable networks. The design principles are to minimise network bandwidth and device resource requirements whilst also attempting to ensure reliability and some degree of assurance of delivery. These principles make the protocol ideal for IoT platforms like ALMANAC where bandwidth and battery power are at a premium. (MQTT, 2016), (MQTT, 16).

6.1.1.4 Eclipse Paho Java Client

The Paho project has been created to provide reliable open-source implementations of open and standard messaging protocols aimed at new, existing, and emerging applications for Machine-to-Machine (M2M) and IoT. Paho reflects the inherent physical and cost constraints of device connectivity. Its objectives include effective levels of decoupling between devices and applications, designed to keep markets open and encourage the rapid growth of scalable Web and Enterprise middleware and applications (Paho, 2016).

6.1.1.5 Pyro - Python Remote Objects

It is a Python library for remote process calling. The technology allows efficient and cross technology communication between a python component and other component regardless of the technology used. The technology network TCP based solution.

6.1.1.6 Docker

Docker is a computer program that performs operating-system-level virtualization, also known as "containerization". It was first released in 2013 and is developed by Docker, Inc.

Docker is used to run software packages called "containers". Containers are isolated from each other and bundle their own application, tools, libraries and configuration files; they can communicate with each other through well-defined channels. All containers are run by a single operating-system kernel and are thus more lightweight than virtual machines. Containers are created from "images" that specify their precise contents. Images are often created by combining and modifying standard images downloaded from public repositories.

6.1.1.7 JSON Web Signature (JWS)

A JSON Web Signature (abbreviated JWS) is an IETF proposed standard [RFC7515] for signing arbitrary data. This is used as the basis for a variety of web-based technologies including JSON Web Token.

6.1.1.8 Visualization

The Visual Analytics tool is developed in AngularJS framework, which takes care of all the basic and content management features of the application. Moreover, the MongoDB is used for the tool's storing requirements. The graphical representations are enabled by the use of the Chart.js and D3.js visualization libraries. They allow great control over the final visual results by using the widely implemented SVG, HTML5, and CSS standards

6.1.1.9 COMPOSITION Contribution

The LA started to be developed in 2014 in the ALMANAC project as a simple CEP for Smart Cities and presented in [1]. Since then, the LA has been developed and transformed in a self-managed learning orchestrator service that combined Complex-Event Processing and Machine Learning and other techniques. Specifically, in COMPOSITION there have been the following improvements:

- Python interoperability layer for programmers or Python SDK
- Micro-batch learning handling for non-iterative learning models
- Implementation and testing of a default detection model for SMTs using the Python SDK and Random Forest model
- Implementation of the JWS standard for the I/O API
- Full Dockerized distribution
- Introduction of CI for quality assurance using automatic testing. This includes
  - Development of Docker based Integration Test for Statement API
  - Development of Docker based Integration Test for CEML API
- Other smaller improvements and fixes had been done. For more detailed information, please check the LinkSmart® project documentation[5] and source[6] code release notes
- For the visual analytics part, COMPOSITION offers a completely web-based tool that is able to support real-time analyses' visualization and interaction with the user. Furthermore, the tool supports visualizations for a wide variety of analytics methods such as time series forecasting, deep learning predictions, Markov-chain predictions, genetic algorithms, trend analysis and statistics.

6.2 Conclusion

In this deliverable, we describe how the LA enables other services in order to execute the COMPOSITION use cases effectively. Additionally, we present evidence of the capacity and scalability of the LA for processing a big amount of data. It is important to mention that the LA is not only able to big data analytics and orchestrate continuous machine learning. The LA allows doing this almost in any environment from the cloud to embedded gateways in an autonomous manner. Additionally, the LA enables descriptive machine learning models; which allows the possibility to redistribute models. This altogether allows the possibility of ubiquitous big data machine

[5] https://docs.linksmart.eu/display/LA
[6] https://code.linksmart.eu/projects/LA/

learning that is private and confidential safe. Moreover, it enables reproducible and re-deployable machine learning process.

On the other hand, a Visual Analytics tool has been implemented for the purposes of the project. The tool is connected with the LA and Data Analytics tools of the project coming from WP3 and WP5 and aims to visualize their output. The VA tool is an interactive tool that enables end-users to visualize, analyse and explore industrial data derived from multiple sources and enhance the decision support, which is provided by the COMPOSITION IIMS. In opposite to other traditional analytics tools the COMPOSITION VA tool is completely web-based. Moreover, it is effortlessly connected with analytics tools as the communication is based on HTTP/MQTT protocols and JSON format. Therefore, newly deployed analytics can connected in a standard way to this tool in order to visualize their outputs.

7. References

[1] D. Bonino, J. A. Carvajal Soto, M. T. Delgado Alizo, A. Alapetite, T. Gilbert, M. Axling, H. Udsen and M. Spirito, “Almanac: Internet of things for smart cities,” *2015 3rd IEEE International Conference on Future Internet of Things and Cloud (FiCloud),* pp. 309-316, 2015.

[2] M. Axling, D. Bonino, D. Ioannidis, N. K. Kaklanis, A. Nizamis, P. Vergori, M. Pardi, G. Insolvibile, J. Benedicto, J. Romero, J. Á. Carvajal Soto, J. Liang, P. Kool, M. Ahlsen and P. Rosengren, “D2.3 The COMPOSITION Architecture Specification I,” COMPOSITION Consortium, 2017.

[3] COMPOSITION, “GRANT AGREEMENT 723145 — COMPOSITION: Annex 1 Research and innovation action,” 2016.

[4] R. J. Calantone and C. A. Di Benedetto, “Performance and time to market: Accelerating cycle time with overlapping stages,” *IEEE Transactions on Engineering Management,* pp. 232-244, 2000.

[5] S. M. Davis, “From “future perfect”: Mass customizing,” *Planning review,* pp. 16-21, 1989.

[6] F. T. Piller, K. Moeslein and C. M. Stotko, “Does mass customization pay? An economic approach to evaluate customer integration,” *Production planning & contro,* pp. 435-444, 2004.

[7] McKinsey Digital, “Industry 4.0 How to navigate digitization of the manufacturing sector,” McKinsey Digital, 2015.

[8] P. D. H. Kagermann, P. D. W. Wahlster and D. J. Helbig, “Recommendations for implementing the strategic initiative INDUSTRIE 4.0,” National Academy of Science and Engineering, Frankfurt (Main), Germany, 2013.

[9] M. Cheeseman, P. Swann, G. Hesketh and S. Barnes, “Adaptive manufacturing scheduling: a flexible and configurable agent-based prototype,” *Production Planning \& Control,* pp. 479-487, 2005.

[10] P. Leitão and F. Restivo, “ADACOR: A holonic architecture for agile and adaptive manufacturing control,” *Computers in industry,* pp. 121-130, 2006.

[11] A. Nassehi, S. Newman and R. Allen, “TEP-NC compliant process planning as an enabler for adaptive global manufacturing,” *Robotics and Computer-Integrated Manufacturing,* pp. 456-467, 2006.

[12] M. Pellicciari, A. O. Andrisano, F. Leali and A. Vergnano, “Engineering method for adaptive manufacturing systems design,” *International Journal on Interactive Design and Manufacturing (IJIDeM),* pp. 81-91, 2009.

[13] J. Lee, B. Bagheri and H.-A. Kao, “A cyber-physical systems architecture for industry 4.0-based manufacturing systems,” *Manufacturing Letters,* pp. 395-412, 2015.

[14] V. Vyatkin, Z. Salcic, P. S. Roop and J. Fitzgerald, “Now That's Smart!,” *IEEE Industrial Electronics Magazine,* pp. 17-29, 2007.

[15] F. S. Fogliatto, G. J. Da Silveira and D. Borenstein, “The mass customization decade: An updated review of the literature,” *International Journal of Production Economics,* pp. 14-25, 2012.

[16] D. Pham and A. Afify, “Machine-learning techniques and their applications in manufacturing,” *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture,* pp. 395-412, 2005.

[17] D.-M. Tsai and R.-H. Yang, “An eigenvalue-based similarity measure and its application in defect detection,” *Image and Vision Computing,* pp. 1094-1101, 2005.

[18] M.-H. C. Li, A. Al-Refaie and C.-Y. Yang, “DMAIC approach to improve the capability of SMT solder printing process,” *IEEE Transactions on Electronics Packaging Manufacturing,* pp. 126-133, 2008.

[19] T.-W. Hui and G. K.-H. Pang, “Solder paste inspection using region-based defect detection,” *The International journal of advanced manufacturing technology,* pp. 725-734, 2009.

[20] G. Acciani, G. Brunetti and G. Fornarelli, “Application of neural networks in optical inspection and classification of solder joints in surface mount technology,” *IEEE Transactions on industrial informatics,* pp. 200-209, 2006.

[21] G. J. Vachtsevanos, I. M. Dar, K. E. Newman and E. Sahinci, “Inspection system and method for bond detection and validation of surface mount devices”. USA Patent 5,963,662, 1999.

[22] J. Á. Carvajal Soto, M. Jentsch, D. Preuveneers and E. Ilie-Zudor, “CEML: Mixing and Moving Complex Event Processing and Machine Learning to the Edge of the Network for IoT Applications,” *Proceedings of the 6th International Conference on the Internet of Things,* pp. 103-110, 2016.

[23] G. Cugola and A. Margara, “Processing flows of information: From data stream to complex event processing,” *ACM Computing Surveys (CSUR),* p. 15, 2012.

[24] C. Andrieu, N. De Freitas, A. Doucet and M. I. Jordan, “An introduction to MCMC for machine learning,” *Machine learning,* pp. 5-43, 2003.

[25] M. M. Gaber, “Gaber, Mohamed Medhat; Krishnaswamy, Shonali; Zaslavsky, Arkady,” *On-board Mining of Data Streams in Sensor Networks,* pp. 307-335, Advanced Methods for Knowledge Discovery from Complex Data.

[26] N. A. Syed, S. Huan, L. Kah and K. Sung, “Incremental learning with support vector machines,” *Citeseer,* 1999.

[27] A. P. Dawid, “Present position and potential developments: Some personal views: Statistical theory: The prequential approach,” *Journal of the Royal Statistical Society. Series A (General),* pp. 278-292, 1984.

[28] S. V. Stehman, “Selecting and interpreting measures of thematic classification accuracy,” *Remote Sensing of Environment,* pp. 77 - 89, 1997.

[29] O. K. Mont, “Clarifying the concept of product- service system,” *Journal of cleaner production,* vol. X, no. 3, pp. 237-245, 2002.