UNIVERSITÀ DI PISA

Process Mining and Intelligence

Smartwatch Based Electrocardiogram Analysis

M. Daole - M. G. Gómez - F. Minutella - A. Schiavo

PROCESS MINING AND INTELLIGENCE

Smartwatch based electrocardiogram analysis

Professor: M. G. C. A. Cimino

2020

Students: M. Daole – M. G. Gómez – F. Minutella – A. Schiavo

TABLE OF CONTENTS

Workflow Modeling Methodology

1. Process analysis

1.1. Activity identification

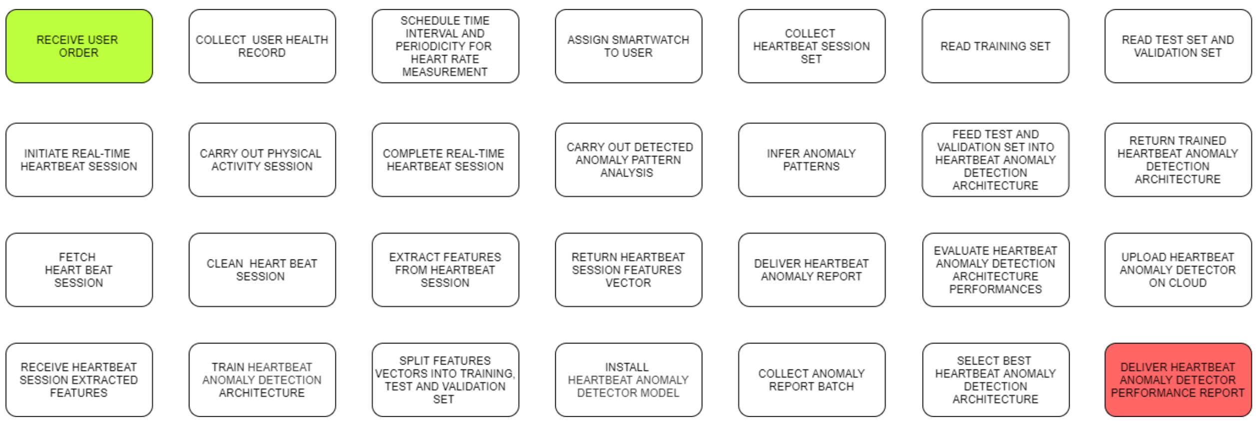


Figure 1. Partial results from activity identification.

1. Receive user order
2. Collect user health record
3. Assign smartwatch to user
4. Collect heartbeat session set
5. Read training set
6. Read test set and validation set
7. Initiate real-time heartbeat session
8. Carry out physical activity session
9. Complete real-time heartbeat session
10. Carry out detected anomaly pattern analysis
11. Infer anomaly patterns
12. Fetch heart beat session
13. Clean heart beat session
14. Extract features from heartbeat session
15. Return trained heartbeat anomaly detection architecture
16. Feed test and validation set into heartbeat anomaly detection architecture
17. Schedule time interval and periodicity for heart rate measurement
18. Return heartbeat session features vector
19. Deliver heartbeat anomaly report
20. Evaluate heartbeat anomaly detection architecture performances
21. Upload heartbeat anomaly detector on cloud
22. Receive heartbeat session extracted features
23. Train heartbeat anomaly detection architecture
24. Split features vectors into training, test and validation set
25. Install heartbeat anomaly detector model
26. Collect anomaly report batch
27. Select best heartbeat anomaly detection architecture
28. Deliver heartbeat anomaly detector performance report

1.2. Link the activities and determine business process

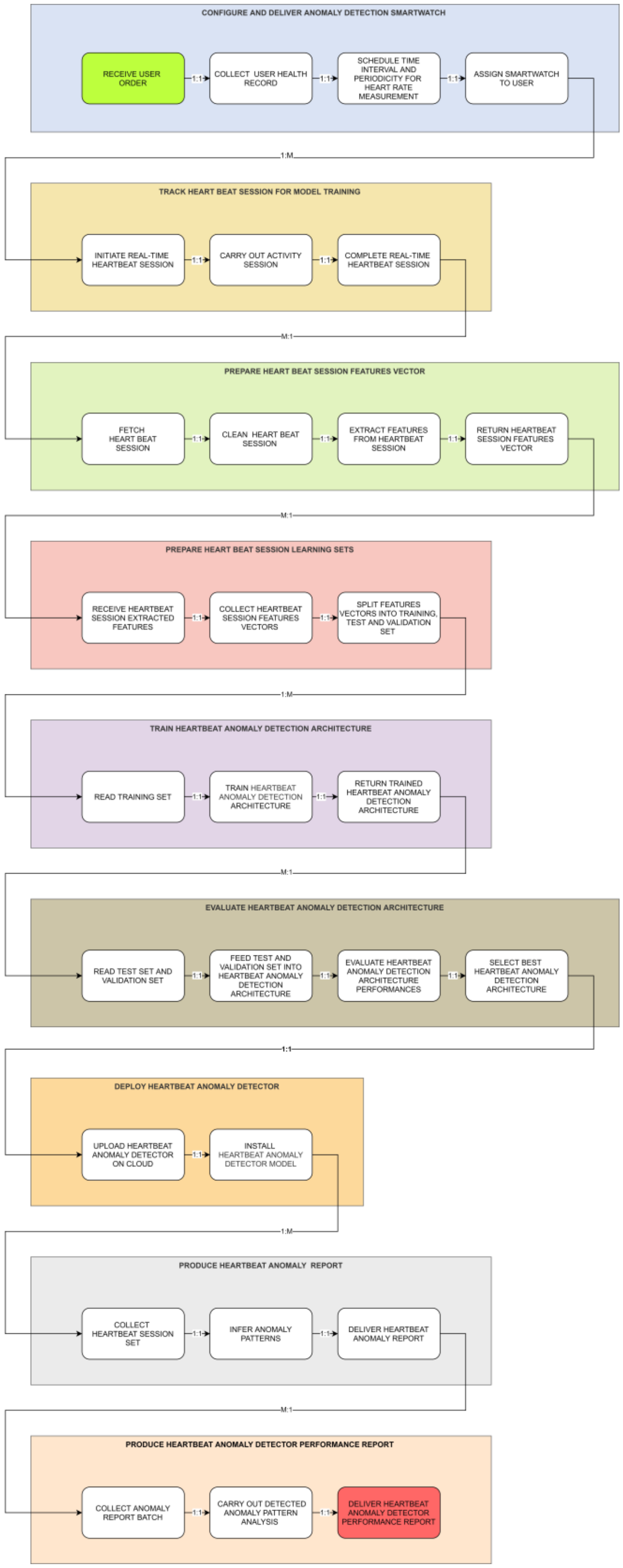


Figure 2 Sequence activities and analyze linkages.

Business Process Modeling Notation

2. BPMN Modeling

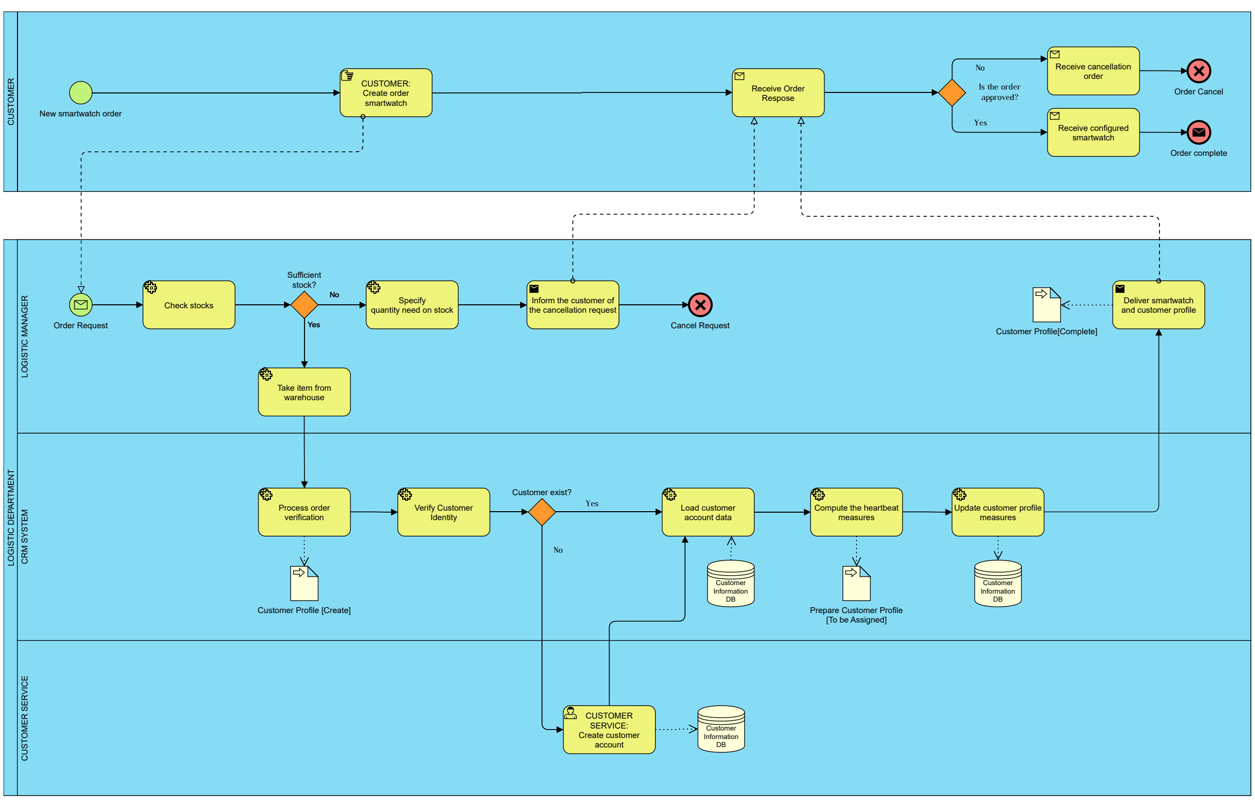
2.1. Diagram services

Introduction

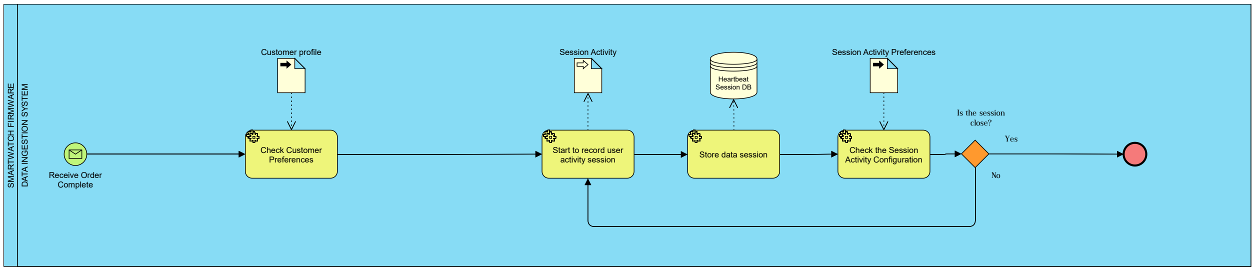
In this section, we analyze the Machine Learning pipeline steps, taking a look on this resource:

- *Problem Definition:* Configure and deliver anomaly detection smartwatch
- *Data Ingestion:* Track heart beat session for model training
- *Data Preparation:* Prepare heartbeat session features vector
- *Data Segregation:* Prepare heartbeat session learning sets
- *Model Training:* Train heartbeat anomaly detection architecture
- *Candidate Model Evaluation*: Evaluate heartbeat anomaly detection architecture
- *Model Deployment*: Deploy Heartbeat Anomaly Detector
- *Performance Monitoring*: Produce Heartbeat Anomaly Report & Produce Heartbeat Anomaly Performance Report

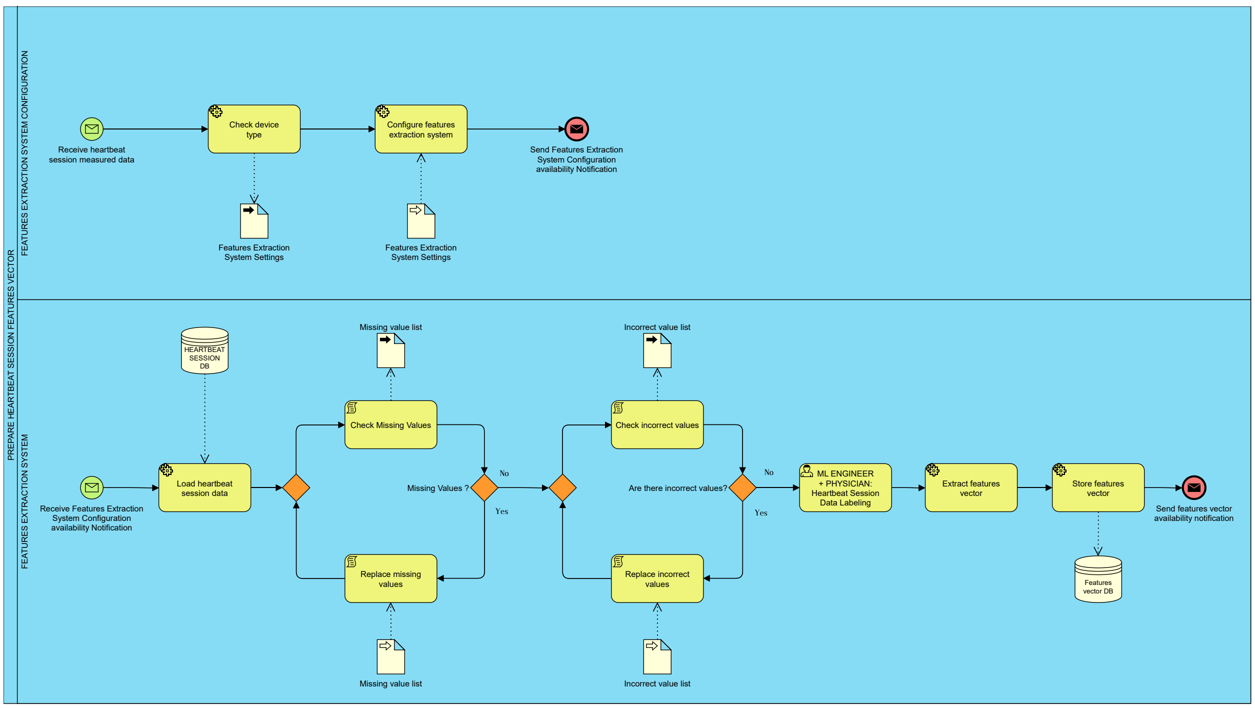
2.1.1. Configure and deliver anomaly detection smartwatch (Marsha Gómez)



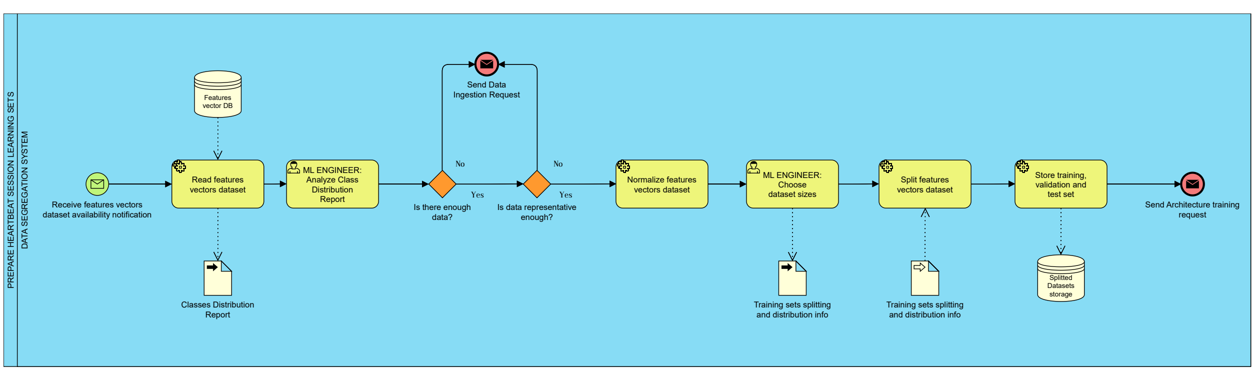
2.1.2. Track heartbeat session for model training (Marsha Gómez)



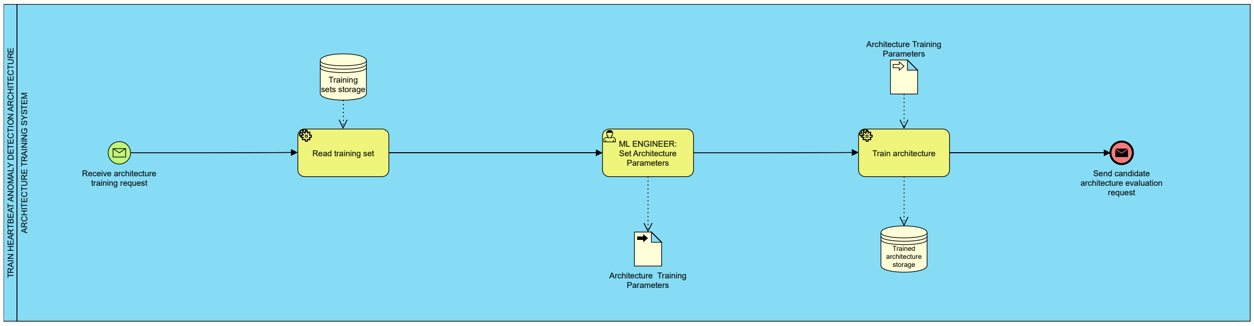
2.1.3. Prepare heartbeat session features vector (Alessio Schiavo)



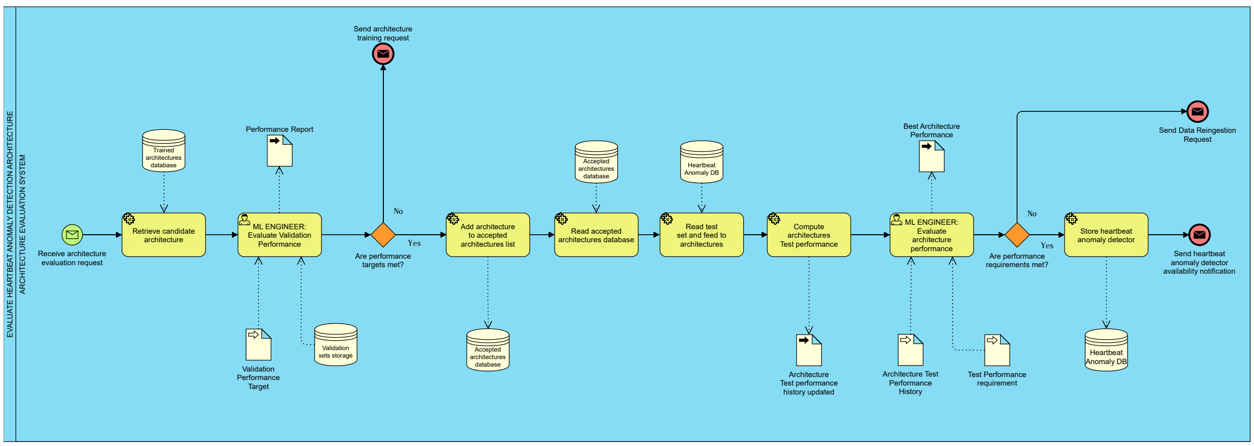
2.1.4. Prepare heartbeat session learning sets (Alessio Schiavo)



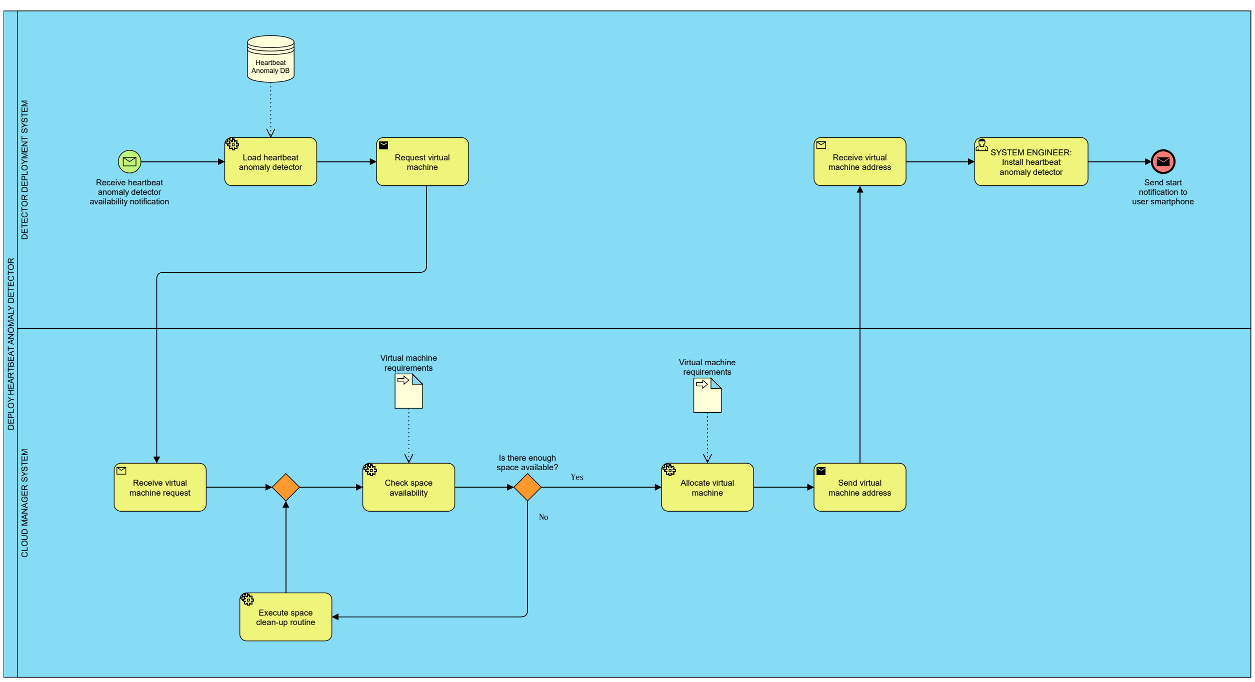
2.1.5. Train heartbeat anomaly detection architecture (Alessio Schiavo)



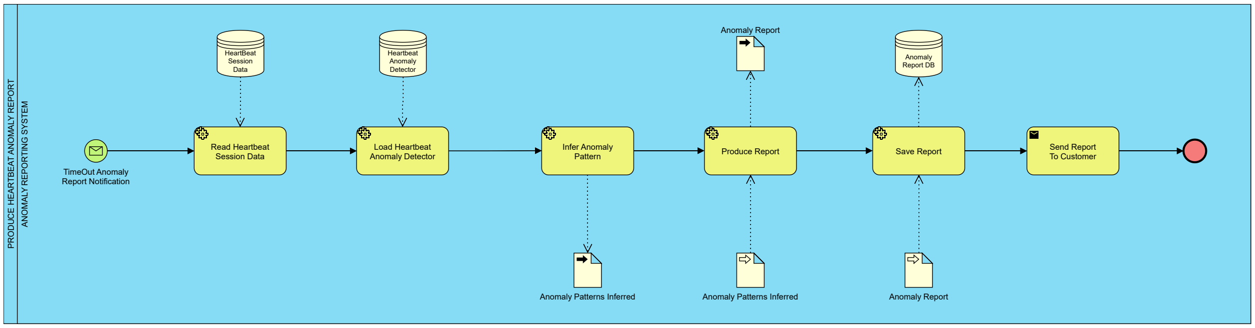
2.1.6. Evaluate heartbeat anomaly detection architecture (Mattia Daole)



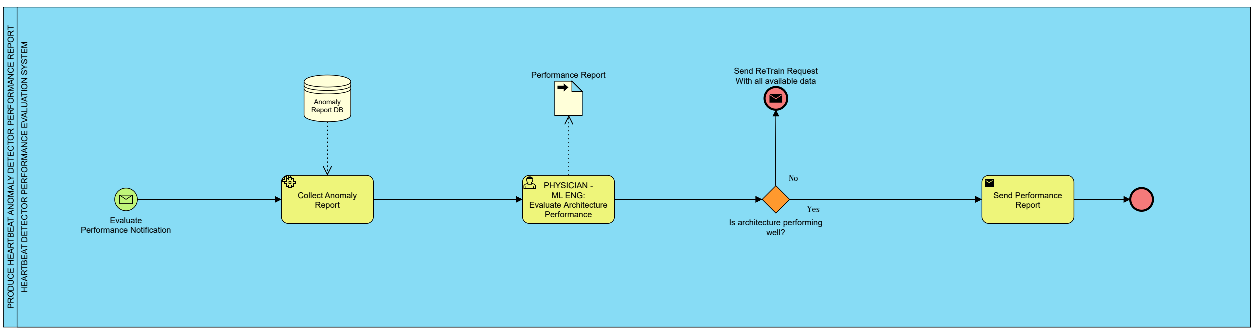
2.1.7. Deploy Heartbeat Anomaly Detector (Mattia Daole)



2.1.8. Produce Heartbeat Anomaly Report (Filippo Minutella)



2.1.9. Produce Heartbeat Anomaly Performance Report (Filippo Minutella)



3. BPMN Simulation

3.1. AS IS - Use case Diagrams

Introduction

On this section we analyze with more detail the use case diagram of the most important and complex task of our system and the costs associated with each UML:

- UML Heart Beat Session Data Labeling
- UML Analyze Class Distribution Report
- UML Evaluate heartbeat anomaly detection architecture
- UML Heartbeat detector performance evaluation system

As you can see in Table 1. AS IS - Salary ComparisonTable 1 has been defined the costs associated with the experts involved in each process.

Profession	Salary per Year	Salary per Hour	Total Cost
Physician[1]	45643	25	1.92
Customer Service Assistant[2]	22280	13	1
System Engineer[3]	34966	19	1.46
Machine Learning Engineer[4]	35930	21	1.61

Table 1. AS IS - Salary Comparison

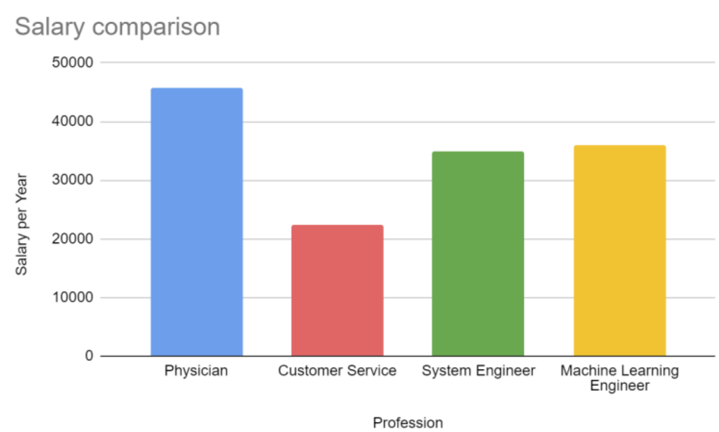


Figure 4. AS IS - Salary Comparison Graph

---

[1] Inc. "Glassdoor". (2008-2021). *Glassdoor*. Salary for https://www.glassdoor.it/Stipendi/physician

[2] Inc. "Glassdoor". (2008-2021). *Glassdoor*. Salary for Customer Service Assistant: https://www.glassdoor.it/Stipendi/customer-service-stipendio-SRCH_KO0,16.htm?clickSource=searchBtn

[3] Inc. "Glassdoor". (2008-2021). *Glassdoor*. Salary for System Engineer: https://www.glassdoor.it/Stipendi/system-engineer-stipendio-SRCH_KO0,15.htm?clickSource=searchBtn

[4] Inc. "Glassdoor". (2008-2021). *Glassdoor*. Salary for Machine Learning Engineer: https://www.glassdoor.it/Stipendi/machine-learning-engineer-stipendio-SRCH_KO0,25.htm

3.1.1. Heartbeat Session Data Labeling (Marsha Gómez)

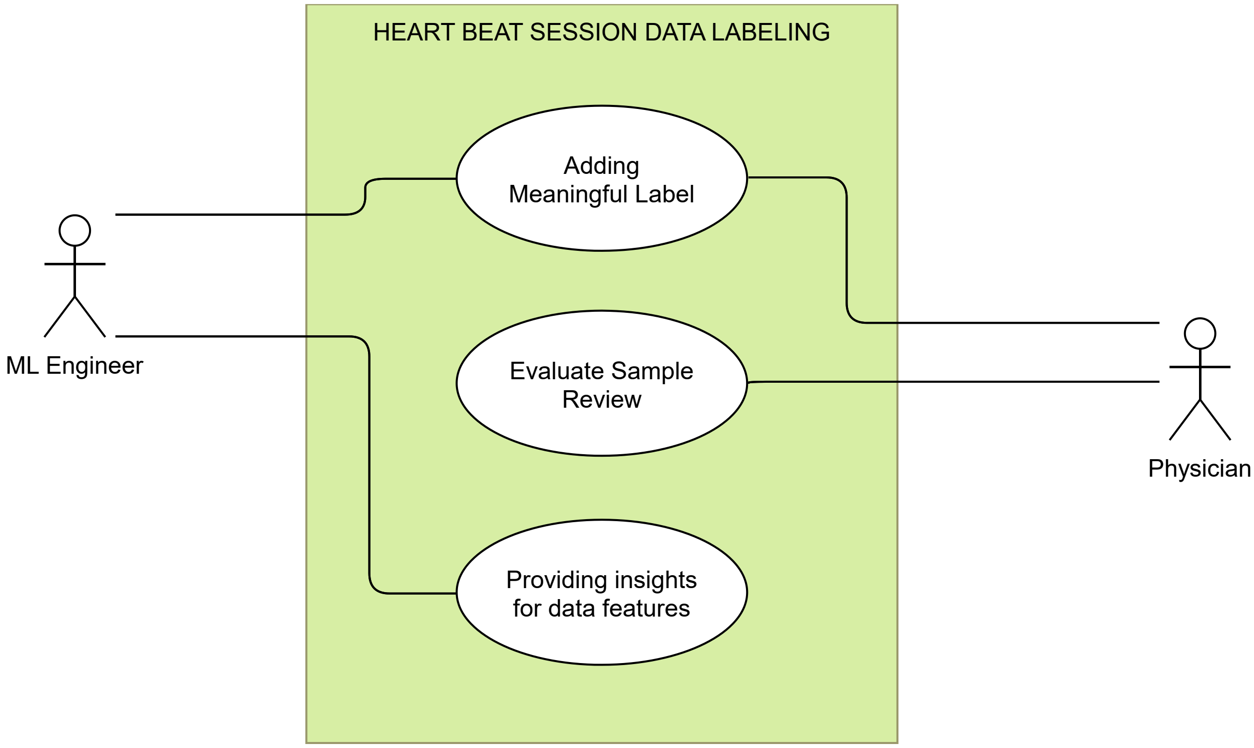


Figure 5. AS IS - UML Heartbeat Session Data Labeling

1. ML ENGINEER: Search "Heart Beat Session Data Labeling"
2. FOR EACH "Raw Data" in "Heart Beat Session Data"
    - **2.1.** ML ENGINEER: Export "Raw Data" and insert to chart viewer
    - **2.2.** SYSTEM: Produce Time Series charts from "Raw Data"
    - **2.3.** SYSTEM: Display Heartbeat Time Series charts
    - **2.4.** PHYSICIAN: Analyze the patterns
    - **2.5.** IF the pattern is "Normal"
        - **2.5.1.** PHYSICIAN: Assign Label as "Normal"
    - **2.6.** ELSE
        - **2.6.1.** PHYSICIAN: Assign "Anomaly" Label
3. SYSTEM: Produce Labeled Dataset

Table 2. AS IS - Cost Human Labor for task "Heart Beat Session Data Labeling"

Task	Actor	Taxonomy	Total Cost
Search Data Labeling	ML Engineer	1 - Remember	(1,61 x 1) = 1,61
Export "Raw Data" and insert to chart viewer	ML Engineer	1 - Remember	(1,61 x 1) = 1,61
Analyze Patterns	Physician	4 - Analyze	(1,92 x 4) = 7.68
Assign Labels	Physician	1 - Remember	(1,92 x 1) = 1,92
Produce Labeled Dataset	ML Engineer	1 - Remember	(1,61 x 1) = 1,61
Total	-	-	14,43

As you can see on the Figure 6, the principal idea of the system is to bring a comprehensive heart beat graph generated by the Machine Learning Engineer and analyzed by the physician for the manual classification and Labeling of each Data Raw pattern. The first image shows a Normal Heart Beat Patter example and the second one shows a Anomaly Heart Beat Patter example.

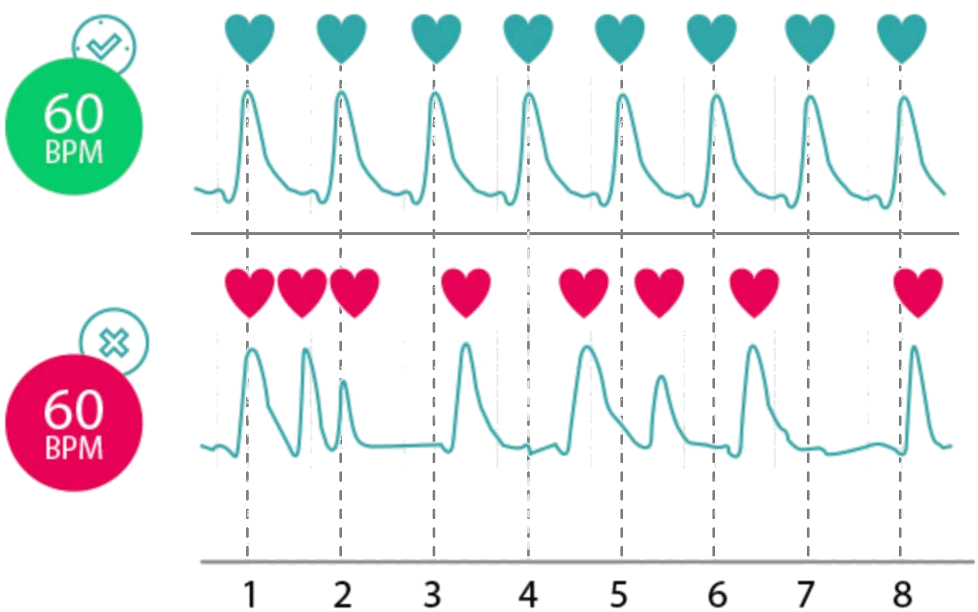


Figure 6. AS IS - The first "Raw Data" is a Normal Pattern and the second "Raw Data" is an Anomaly Pattern.

3.1.2. Evaluate heartbeat anomaly detection architecture (Mattia Daole)

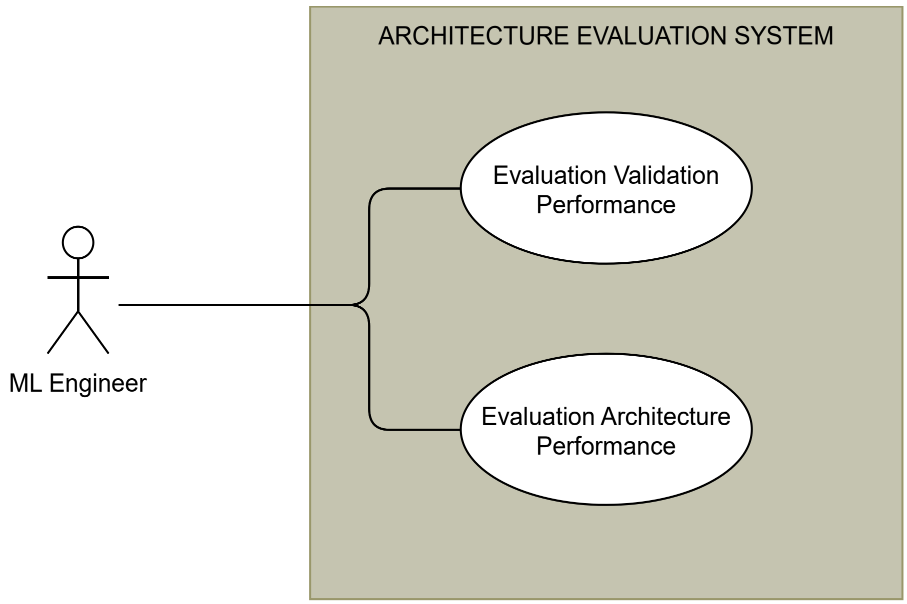


Figure 7. AS IS - UML Evaluate heartbeat anomaly detection architecture

Evaluate Validation Performance:

1. **ML ENGINEER**: Search Validation Error Rate
2. **EVALUATION SYSTEM**: Display "Validation Error rate" and "Training Error Rate"
3. **ML ENGINEER**: Compare "Validation Error rate" and "Training Error Rate"
   - 3.1. **IF** "Validation Error Rate" is smaller than "Training Error Rate"
     - 3.1.1. **ML ENGINEER:** Select "Accept Architecture"
   - 3.2. **ELSE**
     - 3.2.1. **ML ENGINEER:** Select "Reject Architecture"
4. **SYSTEM**: Produce Performance Report

Evaluate Architecture Performance:

1. **EVALUATION SYSTEM**: Display "Accepted Architectures" test performances with Table and Histogram
2. **ML ENGINEER**: Select "Best Architecture" by test performances
   - 2.1. **IF** "Best Architecture" performances satisfy "Test Performance Requirement for Deployment"
     - 2.1.1. **ML ENGINEER:** Select "Best Architecture"
   - 2.2. ELSE
     - 2.2.1. **ML ENGINEER:** Select "Reject Architecture"
3. **SYSTEM**: Produce "Architecture Report"

Table 3. AS IS – Cost Human Labor for task "Evaluate validation performance"

Task	Actor	Taxonomy	Total Cost
Search Validation Performance & Target	ML Engineer	1 – Remember	(1,61 x 1) = 1,61
Visualize Performance	ML Engineer	1 – Remember	(1,61 x 1) = 1,61
Analyze Performances	ML Engineer	2 – Understand	(1,61 x 2) = 3,22
Select Accept/Reject Architecture	ML Engineer	1 – Remember	(1,61 x 1) = 1,61
Total	–	–	8,05

Table 4. AS IS – Cost Human Labor for task "Evaluate architecture performance"

Task	Actor	Taxonomy	Total Cost
Search Test Performance & Graphs	ML Engineer	1 – Remember	(1,61 x 1) = 1,61
Visualize Graphs	ML Engineer	1 – Remember	(1,61 x 1) = 1,61
Analyze Graphs	ML Engineer	4 – Analyze	(1,61 x 4) = 6,44
Produce Report	ML Engineer	1 – Remember	(1,61 x 1) = 1,61
Total	–	–	11,27

The main objective of Machine Learning Engineer is to look the Validation Error Rate, compares it with the Training error rate and Evaluate the Performance of each Architecture for select the best one. On **Errore. L'origine riferimento non è stata trovata.**, we display a data example of the possible output values of the task Evaluate heartbeat anomaly detection architecture.

Architecture	Precision	Recall	Accuracy	Time (sec)
Architecture 1	0.8	0.75	0.8	10
Architecture 2	0.6	0.9	0.55	5
Architecture 3	0.7	0.85	0.85	15

Table 5. AS IS – Test Table (w.r.t Anomalies)

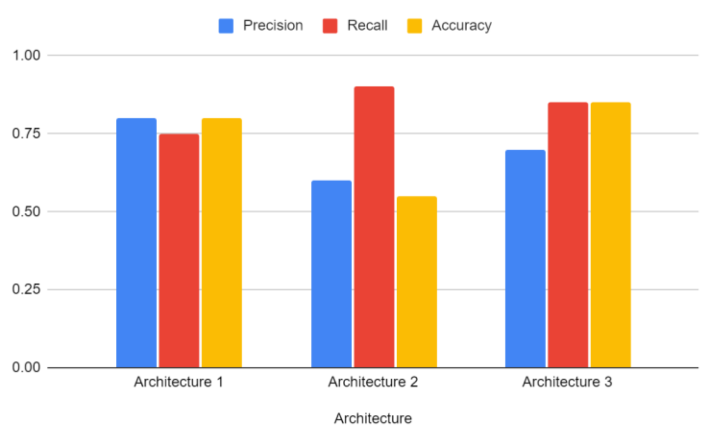


Figure 8. AS IS – Test Graph (w.r.t Anomalies)

3.1.3. Analyze Class Distribution Report (Alessio Schiavo)

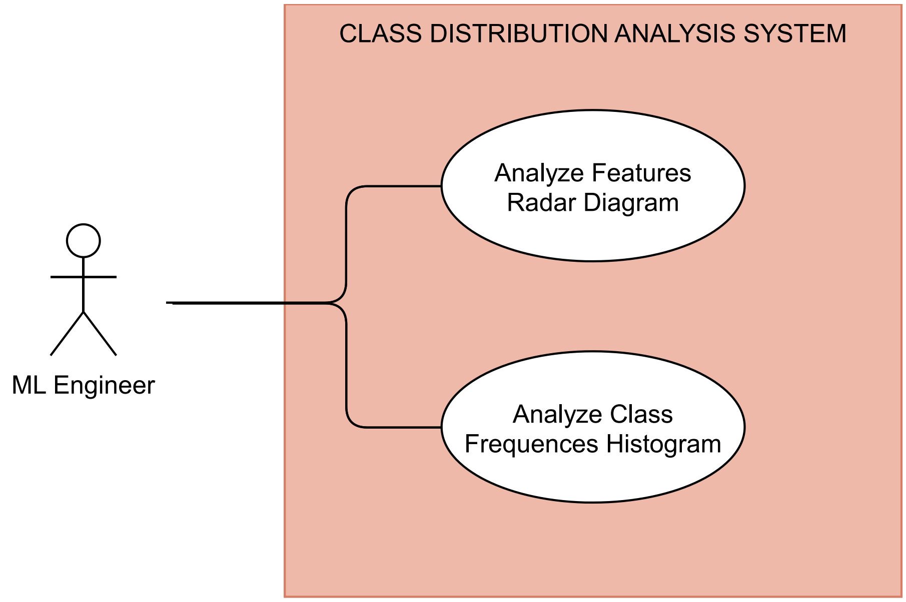


Figure 9. AS IS - UML Analyze Class Distribution Report

1. **ML ENGINEER**: Ask system for Class Distribution Diagrams
2. **SYSTEM**: Produce Features Radar Diagram
3. SYSTEM: Produce Class Frequencies Histogram
4. ML ENGINEER: Analyze Features Radar Diagram for data coverage
    - 4.1. IF data space is not uniformly covered:
        - 4.1.1. ML ENGINEER: Send data recollection request to Data Ingestion System
    - 4.2. ELSE allow the system to proceed.
        - 4.2.1. ML ENGINEER: Analyze Class Frequencies Histogram
        - 4.2.2. IF the unbalance level between the classes exceeds the threshold:
            - 4.2.2.1. ML ENGINEER: Send data recollection request to Data Ingestion System
        - 4.2.3. ELSE allow the system to proceed.

Table 6. AS IS - Cost Human Labor for task "Analyze Class Distribution Report"

Task	Subtask	Actor	Cognitive Taxonomy	Non - automation
Analyze Class Distribution Report	Ask system for Class Distribution Diagrams	ML Engineer	1 - Remember	(1,61 x 1) = 1,61
"	Analyze Features Radar Diagram for data coverage	ML Engineer	4 - Analyze	(1,61 x 4) = 6,44
"	Send data collection request to Data Ingestion System	ML Engineer	1 - Remember	(1,61 x 1) = 1,61
"	Analyze Class Frequencies Histogram	ML Engineer	2 - Understand	(1,61 x 2) = 3,22
Total	-	-	-	12,88

Figure 10 AS IS - Class Frequencies Histogram Diagram Example

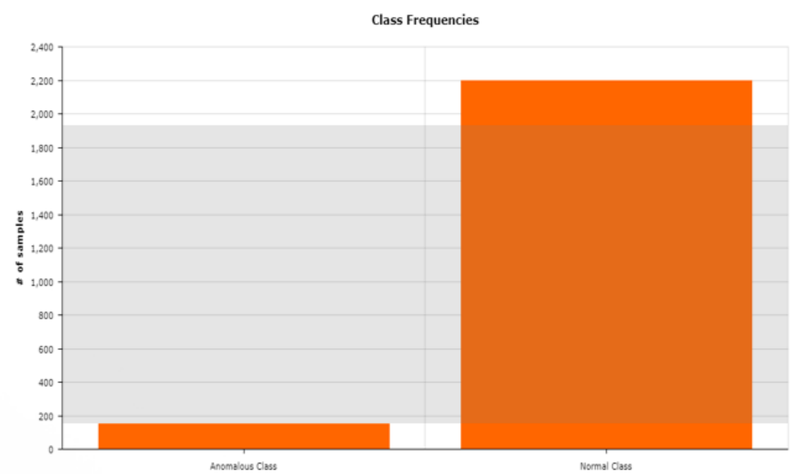
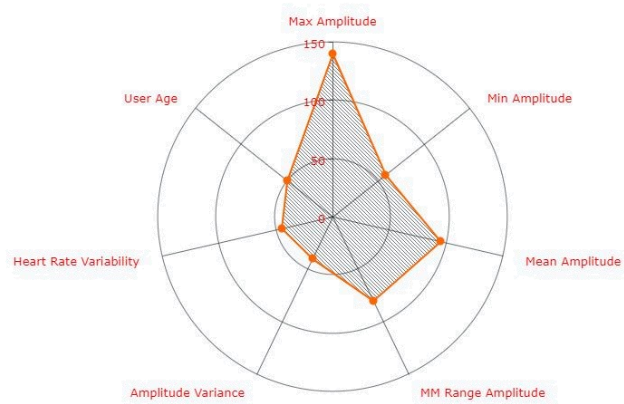


Figure 11 AS IS - Session Features Radar



As you can see on the Figure 10, the area highlighted gray in the Class Frequencies Histogram corresponds to the **max unbalancing tolerated** among the classes. If the classes distribution exceeds this threshold, as in the case reported above on the left, a new data collection phase is needed prior to train the Machine Learning architectures.

3.1.4. Heartbeat detector performance evaluation system (Filippo Minutella)

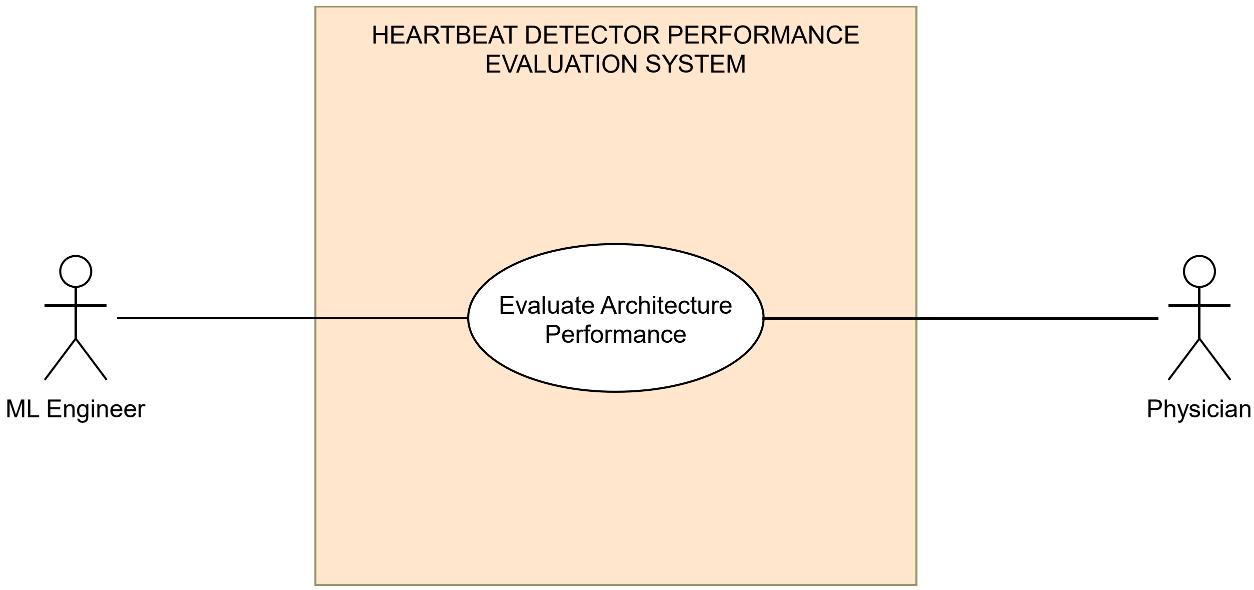


Figure 12. AS IS - UML Heartbeat detector performance evaluation system

1. **ML ENGINEER**: Ask system for predictions on data
2. **SYSTEM**: Show output predictions on data
3. **ML ENGINEER**: Produce time series charts with related predictions [5]
4. **PHYSICIAN**: Analyze labeled time series charts
5. **PHYSICIAN**: Annotate mislabeled time series charts
6. **ML ENGINEER**: Produce heartbeat detector performance report [6]
7. **ML ENGINEER**: Insert the performance report in the system

---

[5] **In point 3** the ML Engineer associates each input sample with its relative prediction. Then he produces a labeled time series chart for each sample.

[6] **In point 6** the ML Engineer checks the number of mislabeled samples (time series charts annotated by the physician) and produces the heartbeat detector performance report.

Table 7. AS IS – Cost Human Labor for task "Heartbeat detector performance evaluation system"

Task	Subtask	Actor	Cognitive Taxonomy	Non – automation
Evaluate Architecture Performance	Ask system for predictions on data	ML Engineer	1 – Remember	(1,61 x 1) = 1,61
"	Produce time series charts	ML Engineer	1 – Remember	(1,61 x 1) = 1,61
"	Analyze labeled time series charts	Physician	4 – Analyze	(1,92 x 4) = 7,68
"	Annotate mislabeled time series charts	Physician	1 – Remember	(1,92 x 1) = 1,92
"	Produce heartbeat detector performance report	ML Engineer	1 – Remember	(1,61 x 1) = 1,61
"	Insert the performance report in the system	ML Engineer	1 – Remember	(1,61 x 1) = 1,61
Total	-	-	-	16,04

The main objective of Machine Learning Engineer on this task is to produce a heartbeat detector performance report. For get this report, the Machine Learning with collaboration of an Expert Physician to label the time series chart samples with the relative predictions Figure 13 is a comprehensive heartbeat time series chart generated by the Machine Learning Engineer and analyzed by the physician for produce a correct performance report.

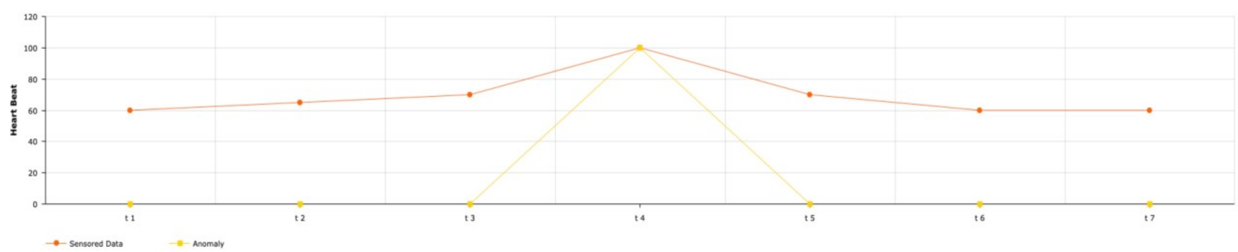
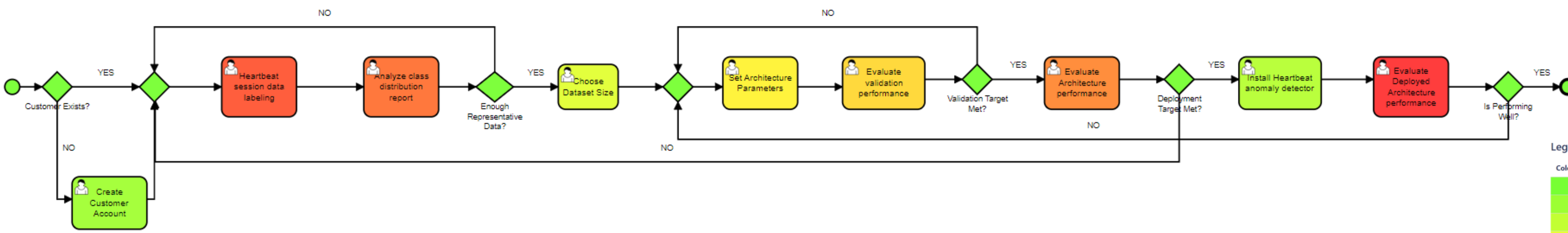


Figure 13. AS IS – Time series chart with related predictions for task "Heartbeat detector performance evaluation system"

3.1.5. AS IS – Operative Human Task Diagram



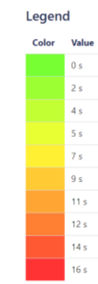


Figure 14. AS IS – Operative Human Task Diagram.

Table 8. AS IS – Operative Human Task. Activity Durations, Costs, Waiting times, Deviations from Thresholds

Name		Waiting time			Duration			Duration over threshold			Cost			Cost over threshold		
	Count	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max
Analyze class distribution report	200	0	0	0	3.2	13.2	27.1	0	0	0	0	0	0	0	0	0
Choose Dataset Size	134	0	0	0	4	5	6.1	0	0	0	0	0	0	0	0	0
Create Customer Account	96	0	0	0	1.7	2	2.2	0	0	0	0	0	0	0	0	0
Evaluate Architecture performance	139	0	0	0	0.5	11.2	27.4	0	0	0	0	0	0	0	0	0
Evaluate Deployed Architecture performance	105	0	0	0	4.7	15.5	28.6	0	0	0	0	0	0	0	0	0
Evaluate validation performance	243	0	0	0	0.3	7.9	15.1	0	0	0	0	0	0	0	0	0
Heartbeat session data labeling	200	0	0	0	9.4	14	20.2	0	0	0	0	0	0	0	0	0
Install Heartbeat anomaly detector	105	0	0	0	2.4	3	3.6	0	0	0	0	0	0	0	0	0
Set Architecture Parameters	243	0	0	0	3.1	7	9.2	0	0	0	0	0	0	0	0	0

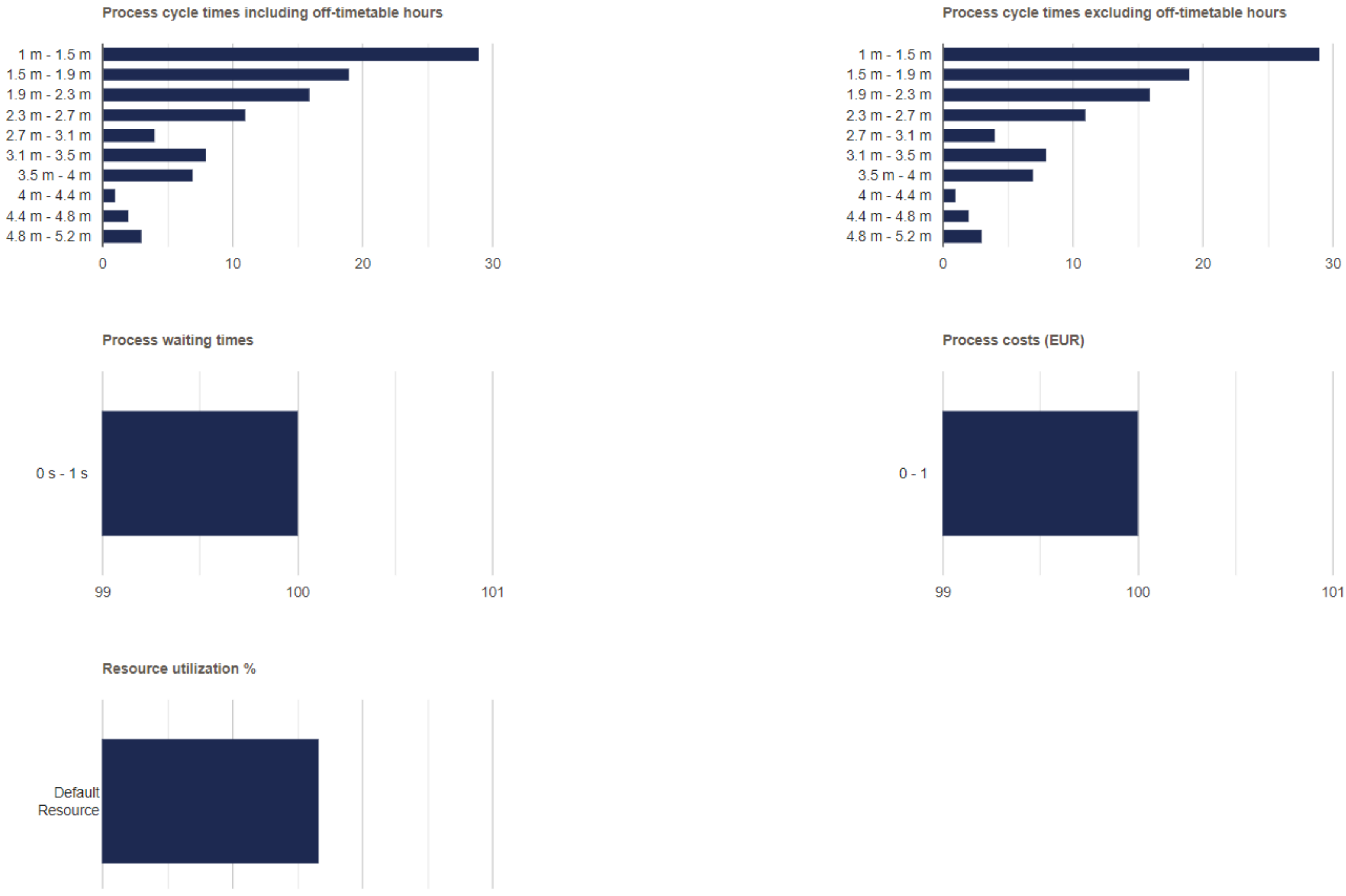


Figure 15. AS IS - Simulation Results. General Charts

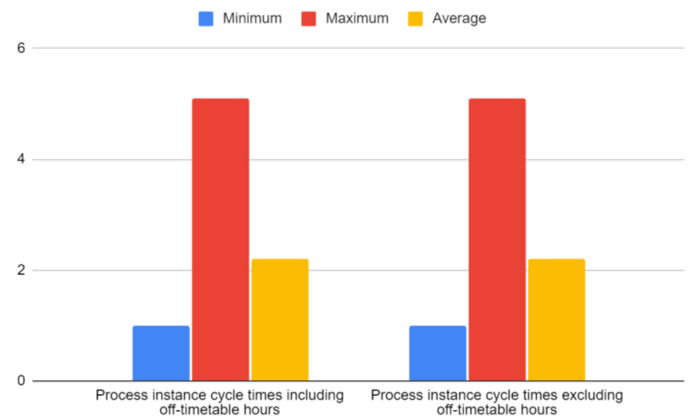
Table 9. AS IS – BIMP Simulation Results. General Information

General information	
Completed process instances	100
Total cost	0 EUR
Total simulation time	1.1 days

Table 10. AS IS – BIMP Simulation Results. Scenario Statistics

Scenario Statistics	Minimum	Maximum	Average
Process instance cycle times including off-timetable hours	1 minute	5.1 minutes	2.2 minutes
Process instance cycle times excluding off-timetable hours	1 minute	5.1 minutes	2.2 minutes
Process instance costs	0 EUR	0 EUR	0 EUR

Figure 16. AS IS – Global Scenario Statistics. Values expressed on Minutes



As can be noticed from the heatmap above, the task having the major cost is "***Evaluate Deployed Architecture Performance***". Indeed, the task requires the human contributions from: the ML Engineer and the Physician; and has a total execution cost of 16 cost units (one cost unit corresponds to 13 dollars/h). This task is executed once a month in order ensure that the system is working properly, assuming we acquire 50 new customers each month, the cost increases accordingly by a factor of 50.

Assuming to have 250 customers, the total task cost per month is $250 * 16 = 4000$ on average. Another costly part of the workflow is the "Heartbeat session data labeling" and "Analyze class distribution report" cycle which is repeated several times for each architecture: the cost for these two tasks together is on average 27,5 cost units for each cycle iteration. Assuming still that we acquire 50 new customers each month, 50 new architectures must be developed accordingly and supposing that on average we need two cycles iterations for each customer, the total cost per month of the cycle is $50 * 2 * 27{,}5 = 2750$ cost units.

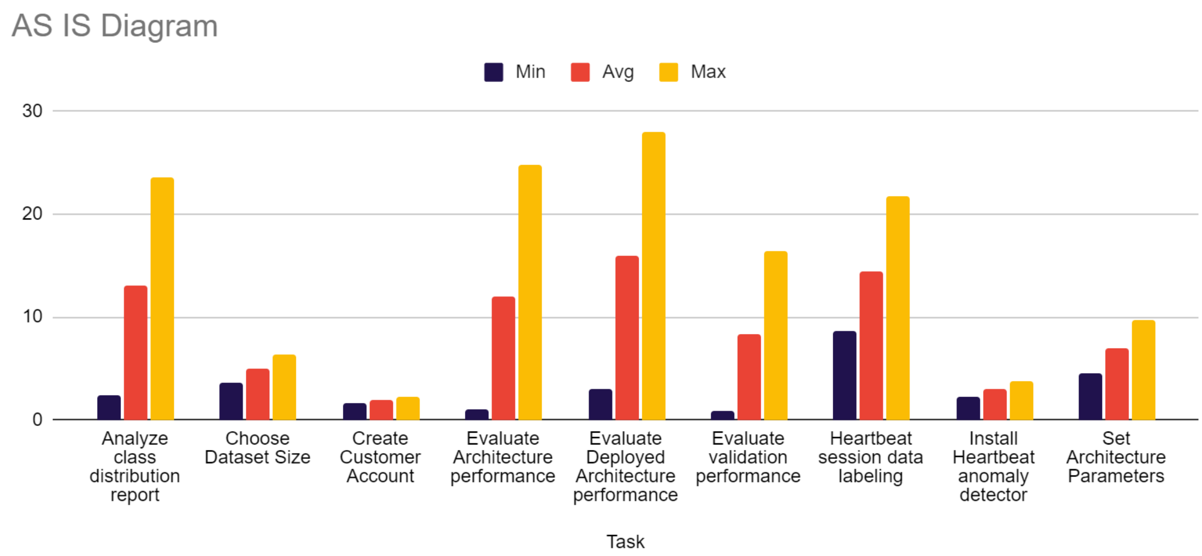


Figure 17. AS IS – Diagram Time. Min, Average and Max cost Values with 100 customers.

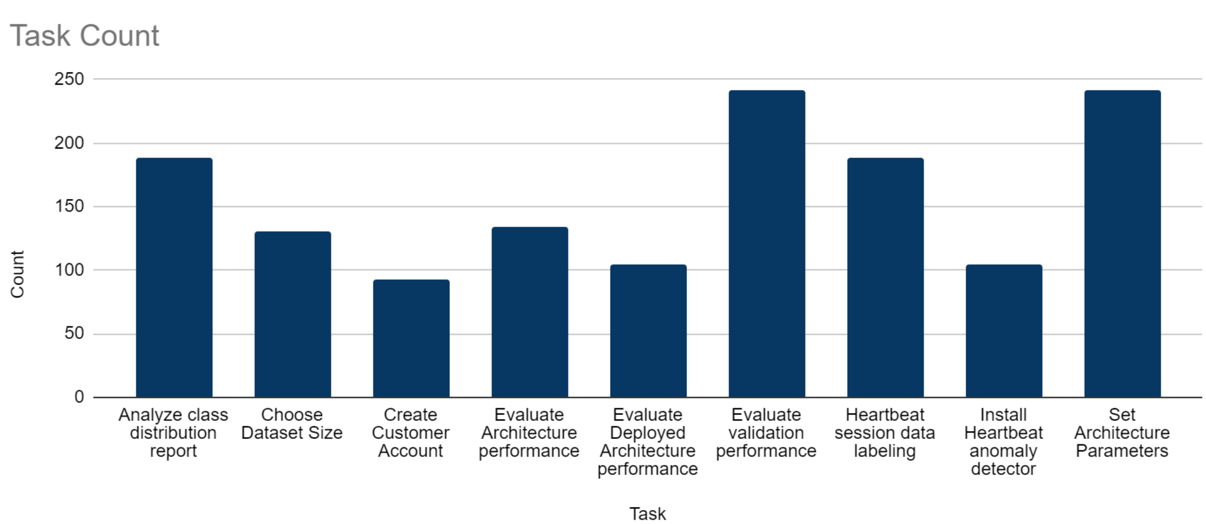


Figure 18. AS IS – Count Execution Task on 100 customers.

3.2. TO BE – Use case Diagrams

3.2.1. The hyper parameters setting is done automatically by the System

The main idea is use values from "similar users", How to do that ? Well, consist on check for existing profiles users and compare if an approximative profile can be use for an auto pre-set of the architecture parameters, avoiding and diminishing the execution times on the Manual Case "*Set Architecture Parameters*" and on the task "*Heartbeat Session data labeling*". We expected an optimize on the complexity and time execution.

3.2.2. Improvement on Heartbeat detector performance evaluation system

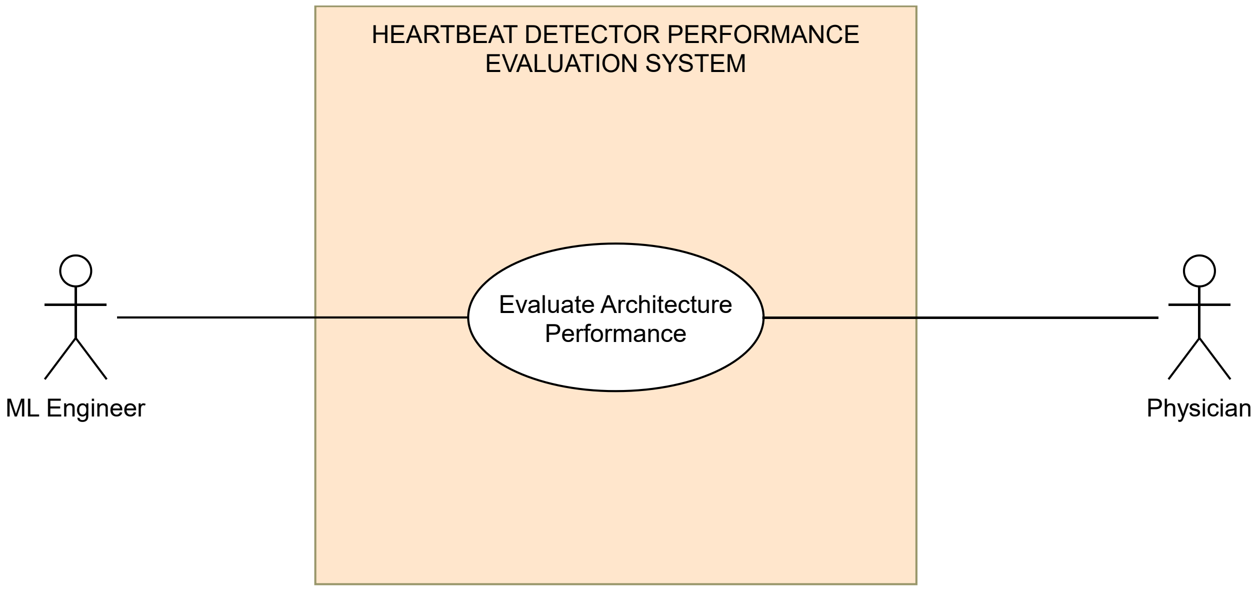


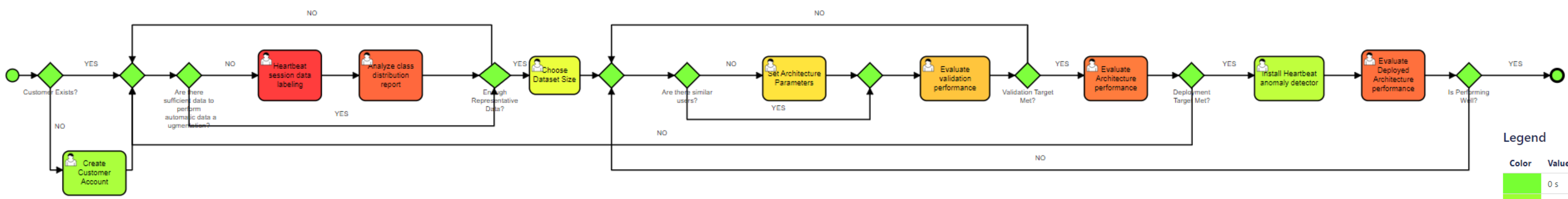
Figure 19. TO BE – UML Heartbeat detector performance evaluation system

1. **PHYSICIAN:** Ask system for time series charts with related predictions
2. **SYSTEM:** Show series charts with related predictions
3. **PHYSICIAN:** Analyze labeled time series charts
4. **PHYSICIAN:** Annotate mislabeled time series charts
5. **SYSTEM:** Produce heartbeat detector performance report

Table 11. TO BE – Cost Human Labor for task "Heartbeat detector performance evaluation system"

Task	Subtask	Actor	Cognitive Taxonomy	Non – automation
Heartbeat detector performance evaluation system	**Ask system for time** series charts with related predictions	Physician	1-Remember	(1,92 x 1) = 1,92
"	Analyze labeled time series charts	Physician	4 – Analyze	(1,92 x 4) = 7,68
"	Annotate mislabeled time series charts	Physician	1 – Remember	(1,92 x 1) = 1,92
Total	-	-	-	11,52

3.2.3. TO BE - Operative Human Task Diagram



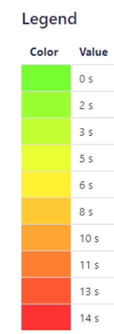


Figure 20. TO BE - Operative Human Task Diagram

Table 12. TO BE - Operative Human Task. Activity Durations, Costs, Waiting times, Deviations from Thresholds

Name	Waiting time				Duration			Duration over threshold			Cost			Cost over threshold		
	Count	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max
Analyze class distribution report	140	0	0	0	1	12.5	22.7	0	0	0	0	0	0	0	0	0
Choose Dataset Size	143	0	0	0	3.6	4.9	6.3	0	0	0	0	0	0	0	0	0
Create Customer Account	97	0	0	0	1.8	2	2.2	0	0	0	0	0	0	0	0	0
Evaluate Architecture performance	154	0	0	0	0.3	11.6	24.7	0	0	0	0	0	0	0	0	0
Evaluate Deployed Architecture performance	111	0	0	0	0.7	12.1	21.8	0	0	0	0	0	0	0	0	0
Evaluate validation performance	239	0	0	0	0.9	8.4	17.8	0	0	0	0	0	0	0	0	0
Heartbeat session data labeling	140	0	0	0	9.6	14.4	19	0	0	0	0	0	0	0	0	0
Install Heartbeat anomaly detector	111	0	0	0	2.2	3	3.8	0	0	0	0	0	0	0	0	0
Set Architecture Parameters	55	0	0	0	5.1	7	9.8	0	0	0	0	0	0	0	0	0

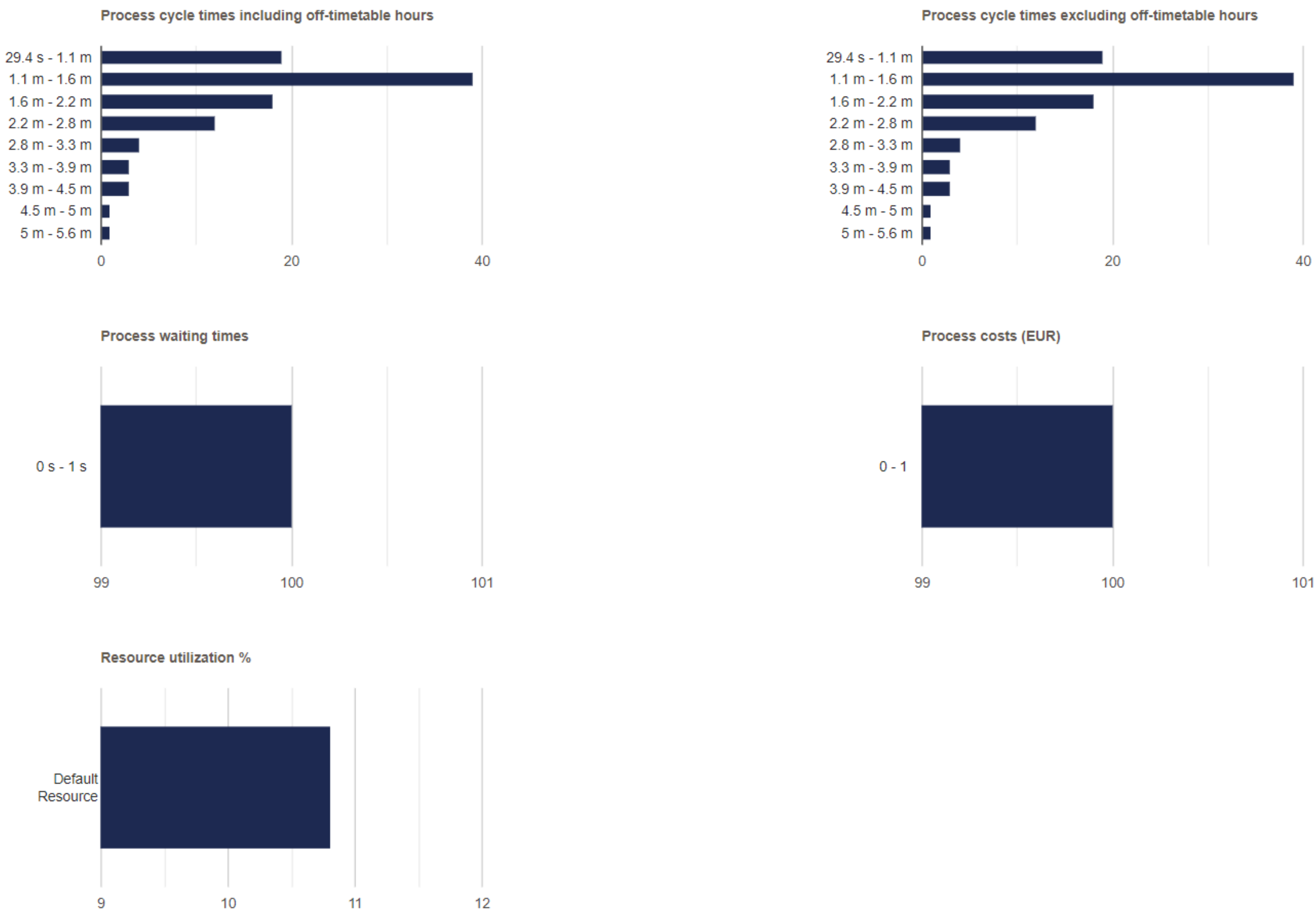


Figure 21. TO BE - Simulation Results. General Charts

Table 13. TO BE – BIMP Simulation Results. General Information

General information	
Completed process instances	100
Total cost	0 EUR
Total simulation time	1.1 days

Table 14. TO BE – BIMP Simulation Results. Scenario Statistics

Scenario Statistics	Minimum	Maximum	Average
Process instance cycle times including off-timetable hours	29.4 seconds	6.1 minutes	1.8 minutes
Process instance cycle times excluding off-timetable hours	29.4 seconds	6.1 minutes	1.8 minutes
Process instance costs	0 EUR	0 EUR	0 EUR

As can be noticed from the heatmap above, the critical parts of the AS IS workflow previously highlighted have been significantly improved. As far as the "Evaluate Deployed Architecture Performance" task, the job previously carried out by the ML Engineer has been automatized and thus done by the system itself. Therefore, the updated cost of the task is on average 9.4 cost units due to the cost of the Physician activities (check the TO BE use case for more details) with a single execution improvement of 9.4 cost units (the cost halved).

As far as the architecture parameters setting and evaluation procedure cycle is concerned, the parameters setting task has been automatized and is now carried out by the system (the architecture parameters for a new customer are set to the same values of the ones of the most similar already existing customer we have). Therefore, the updated cycle cost is 7.8 cost units: we obtained a drop in cost of 15.1-7.8 = 7.3 cost units per cycle iteration.

Table 15. TO BE – Cost Human tasks

Nome	Cost
Create Customer Account	2
Heartbeat session data labeling	15
Analyze class distribution report	10
Choose Dataset Size	5
Set Architecture Parameters	7
Evaluate validation performance	8,05
Evaluate Architecture performance	11,27
Install Heartbeat anomaly detector	3
Evaluate Architecture performance	19,26

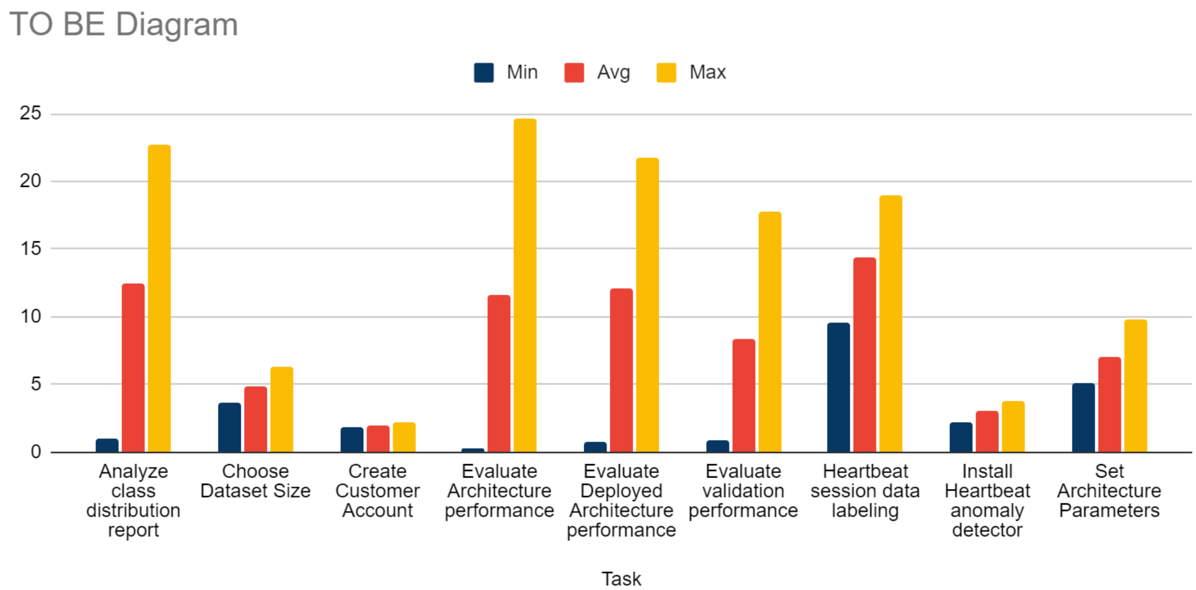


Figure 22. TO BE- Diagram Time. Min, Average and Max cost Values with 100 customers

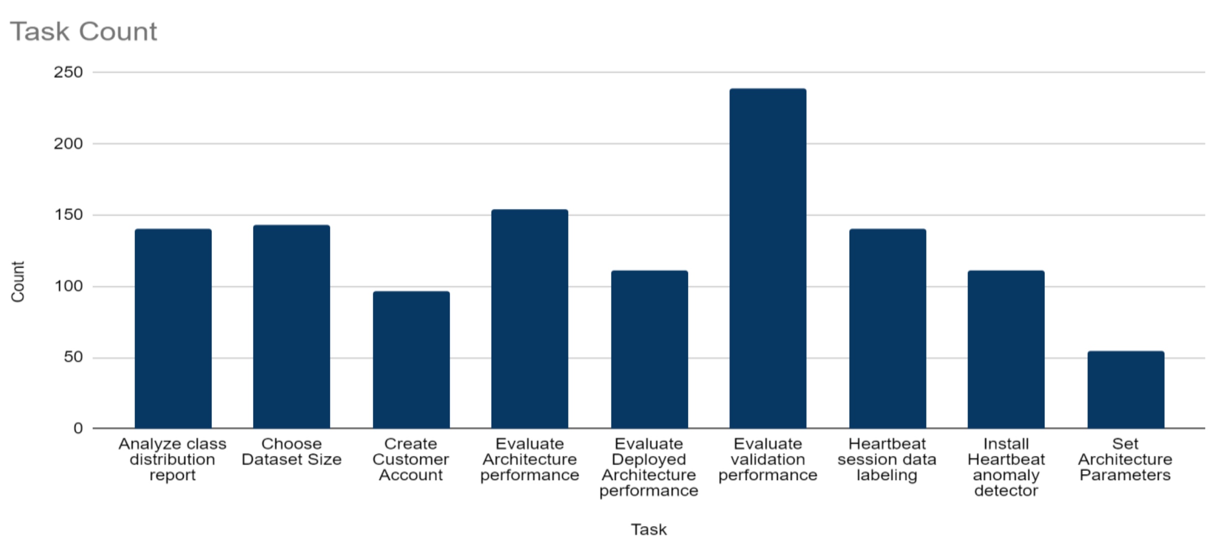


Figure 23. TO BE - Count Execution Task on 100 customers.

3.3. Comparison AS IS vs TO BE

As you can notice, the change proposed on the simulation To Be brings a better result. The charts on Figure 24 shows in a better way this comparison.

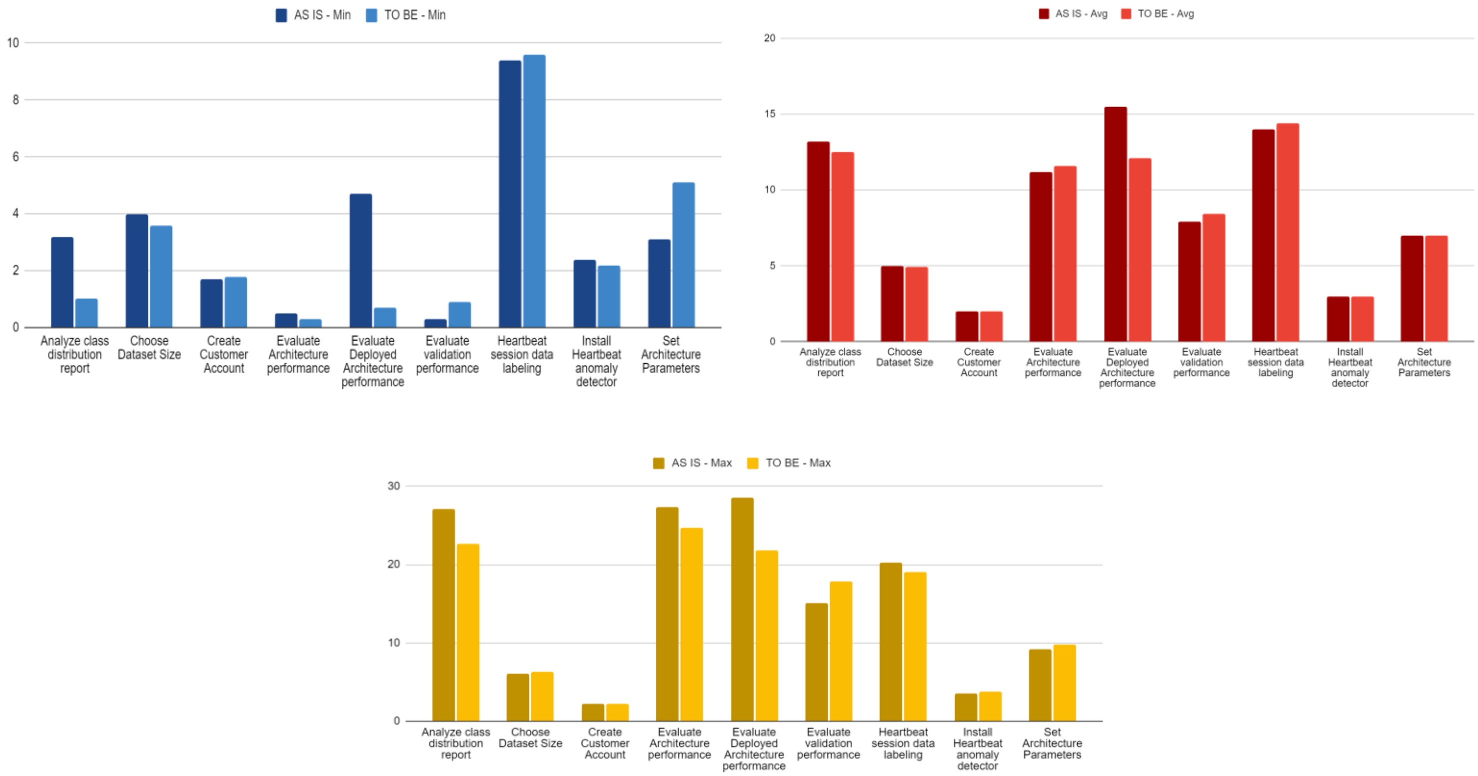


Figure 24. Comparison AS IS vs TO BE. Min, Avg and Max.

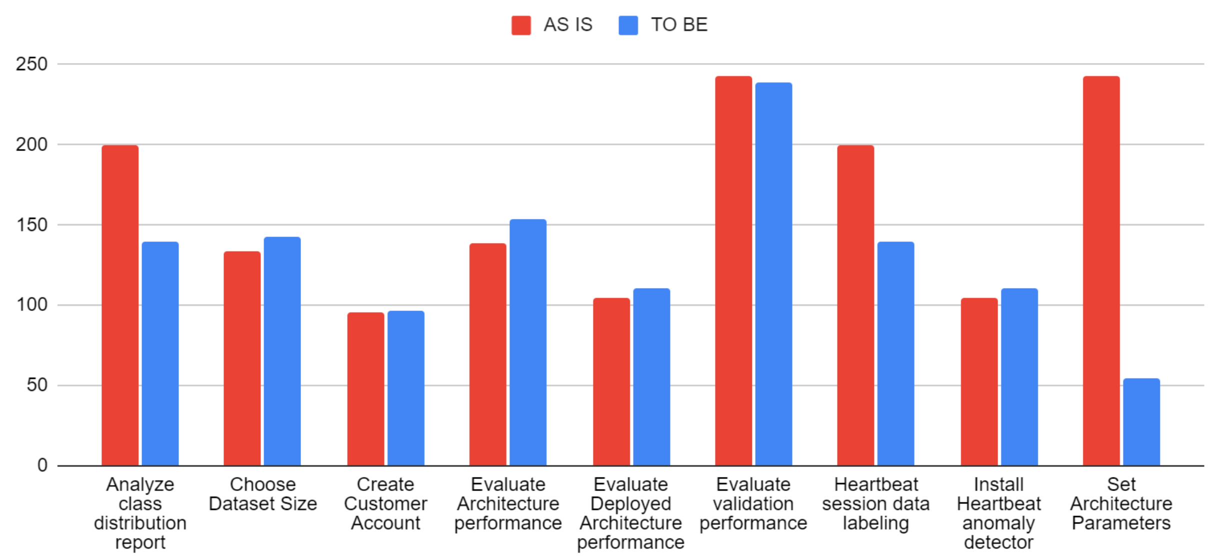


Figure 25. Comparison AS IS vs TO BE. Count Execution Task on 100 customers

4. Process Mining

In this phase of the project a new different data driven approach is adopted: by exploiting the BIMP Simulator online tool fictitious data logs are generated for each Business Process. The idea is to pretend that these logs are indeed real-world data and a process mining procedure is carried out on top of these in order to automatically derive both the BPMN Process diagrams and the process transition maps. Once obtained the mined process diagrams these are compared with the original ones in order to discuss eventual differences. The mining tool exploited in this phase are the commercial one Apromore, the open-source software ProM and Disco.

Subsequently, the generated logs are manually modified so as to introduce possibly meaningful variations with respect to normative processes (we pretend that these modifications are actual deviations of some process instances). Afterwards, conformance check is carried out on both the original and the modified process versions and then the resulting four quality measures (fitness, precision, generalization, simplicity) are compared and the differences analyzed.

4.1. Prepare heartbeat session features vector (unmodified logs)

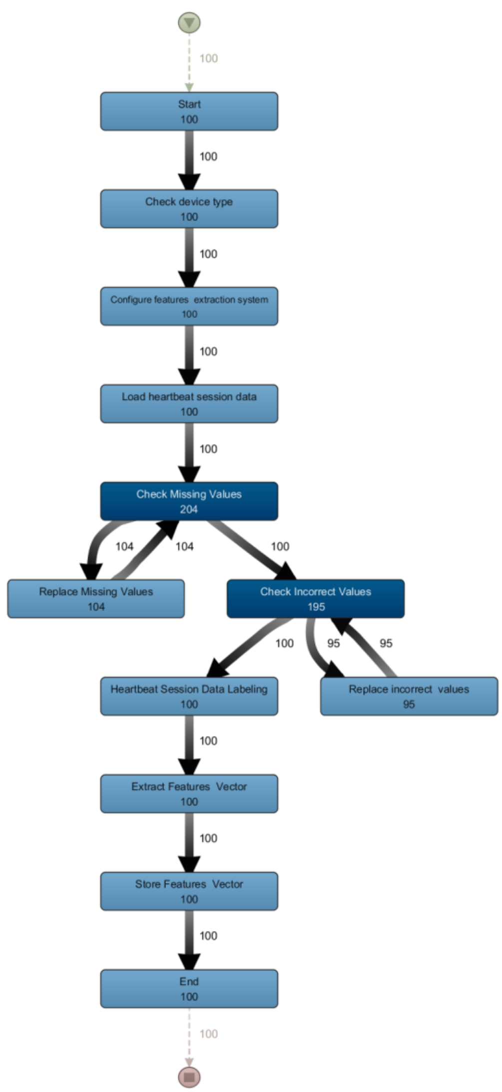


Figure 26. Transition Map (Disco generated) obtained from the simulation logs is reported. Prepare heartbeat session features vector (unmodified logs)

4.1.1. BPMN mined with ProM (unmodified logs)

The Figure 27 bellow the BPMN diagram mined with **ProM** is reported. It can be noticed that, as expected, it is equal to the normative BPMN 2.0 diagram:

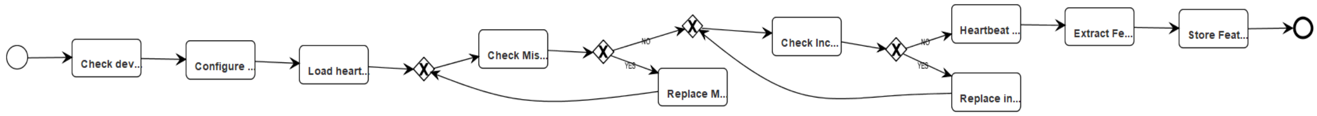


Figure 27. BPMN mined with ProM "Prepare heartbeat session features vector (unmodified logs)"

4.1.2. BPMN mined with Apromore (unmodified logs)

The Figure 28 bellow the BPMN diagram mined with **Apromore** is reported. It can be noticed that, as expected, it is equal to the normative BPMN 2.0 diagram:

Figure 28. BPMN mined with Apromore "Prepare heartbeat session features vector (unmodified logs)"

This diagram is a simple example of Mining Data using the tools ProM and Apromore. For the simplicity of the task, as you can be noticed, the two BPMN diagrams are no different.

4.1.3. Petri Net generated from BPMN (unmodified logs)

In order to compute the four process quality measures, firstly, the Petri Net is generated from the BPMN with ProM.

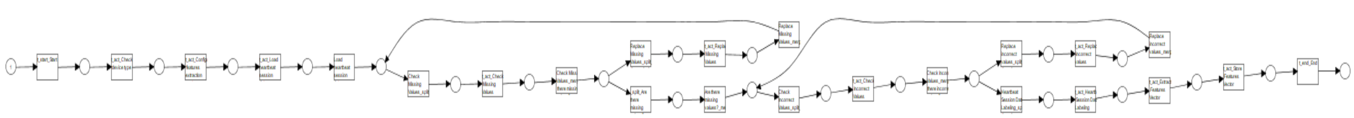


Figure 29. Petri Net generated with ProM "Prepare heartbeat session features vector (unmodified logs)"

4.1.4. Conformance check: quality measures (unmodified logs)

The first three quality measures have been computed exploiting the ProM plugins while the simplicity has been manually computed according to the formula.

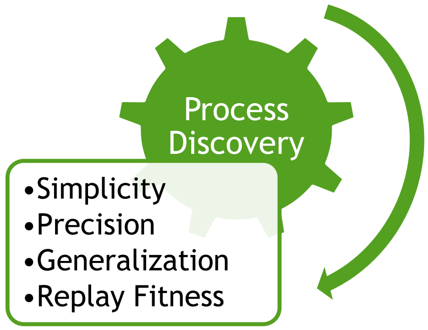


Figure 30. Quality measures

- *Replay Fitness:*

We calculate the fitness of the model using the following formula:

$$f(L, M_m) = \frac{1}{2}\left(1 - \frac{\sum_{i=1}^{k} n_i m_i}{\sum_{i=1}^{k} n_i c_i}\right) + \frac{1}{2}\left(1 - \frac{\sum_{i=1}^{k} n_i r_i}{\sum_{i=1}^{k} n_i p_i}\right)$$

k	Number of different traces
n_i	Number of traces of type k in L
m_i	Number of missing tokens (artificially added) parsing i
r_i	Number of remaining tokens parsing i
c_i	Number of consumed tokens parsing i
p_i	Number of produced tokens parsing i

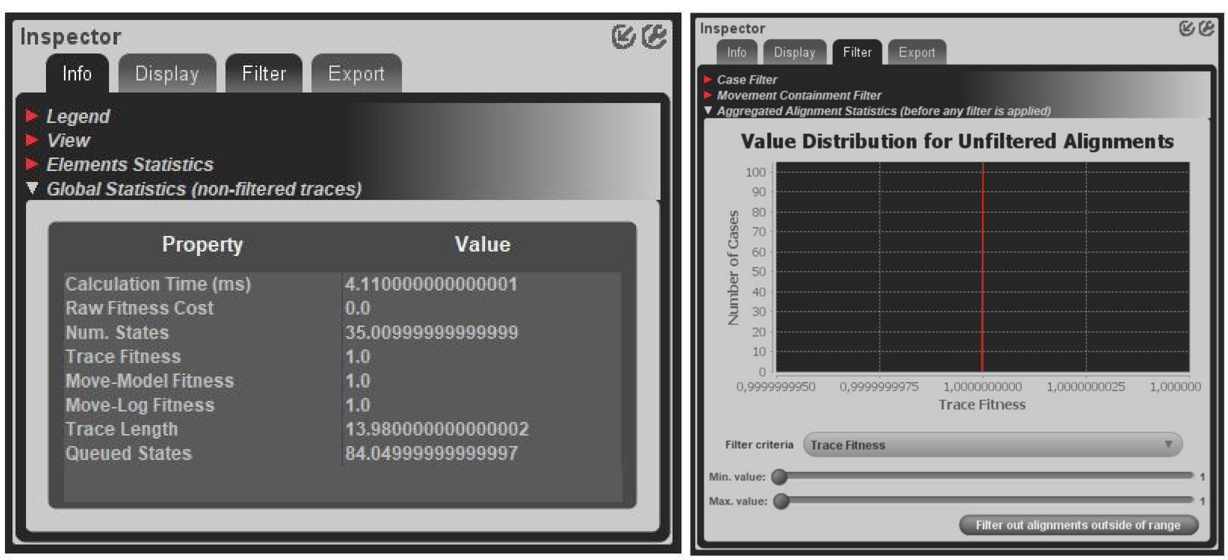


Figure 31. Fitness. Conformance check "Prepare heartbeat session features vector (unmodified logs)"

As the expected result, the trace fitness of the logs replayed on the petri net model, has value 1: the model perfectly fits the logs (all logs can be successfully replayed). (Figure 31Figure 31)

- *Normal log case example:*

Table 16. CSV Single case representation. Case 61. (Unmodified log)

Event	Case ID	Activity	Start Timestamp	elementId
	61	Start	08/02/2021 10:01	BP03_BP39
	61	Check device type	08/02/2021 10:01	BP03_BP95
	61	Configure features extraction system	08/02/2021 10:01	BP03_BP97
	61	Load heartbeat session data	08/02/2021 10:01	BP03_BP99
	61	Check Missing Values	08/02/2021 10:01	BP03_BP103
	61	Check Incorrect Values	08/02/2021 10:01	BP03_BP116
	61	Heartbeat Session Data Labeling	08/02/2021 10:01	BP03_BP124
	61	Extract Features Vector	08/02/2021 10:01	BP03_BP126
	61	Store Features Vector	08/02/2021 10:01	BP03_BP128
	61	End	08/02/2021 10:01	BP03_BP130

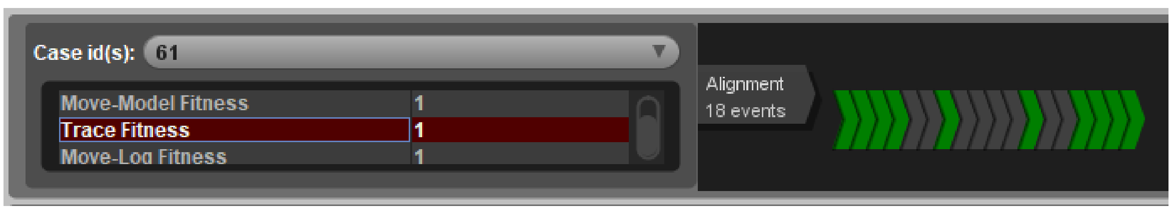


Figure 32. Project Alignment to log. Case 61. (Unmodified log)

As you can see on Figure 32, this case have an individual trace fitness equal to one because all their process are synchronous move.

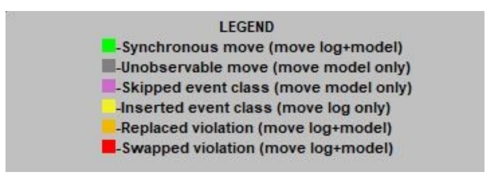


Figure 33. ProM Legend Description

4.1.5. Precision and Generalization

- *Precision*

The model should not allow for very different behavior with respect to the logs, the closer is the value to one the higher the model precision. Having a low precision value means having an underfitting model.

$$P(L,M) = \frac{1}{|\varepsilon|}\sum_{e\epsilon\varepsilon}\frac{|en_L(e)|}{|en_M(e)|}$$

ε	Collection of *unique events* in a context of the log
$en_M(e)$	Enabled activities
$en_L(e)$	Number of missing tokens (artificially added) parsing i

Precision: 0,93831

- *Generalization*

As opposed to precision, a model with a good generalization value is a model allowing some abstraction. (often there is noise or missing values in the logs). The higher the generalization value lower the model overfitting.

Generalization: 0,99827

As expected, the precision and generalization values obtained are **both very good** since computed on the model mined from the **unmodified logs**.

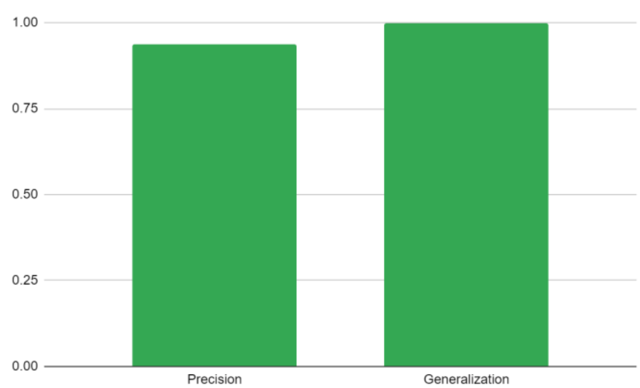


Figure 34. Precision and Generalization for "Prepare heartbeat session features vector (unmodified logs)"

4.1.6. Simplicity

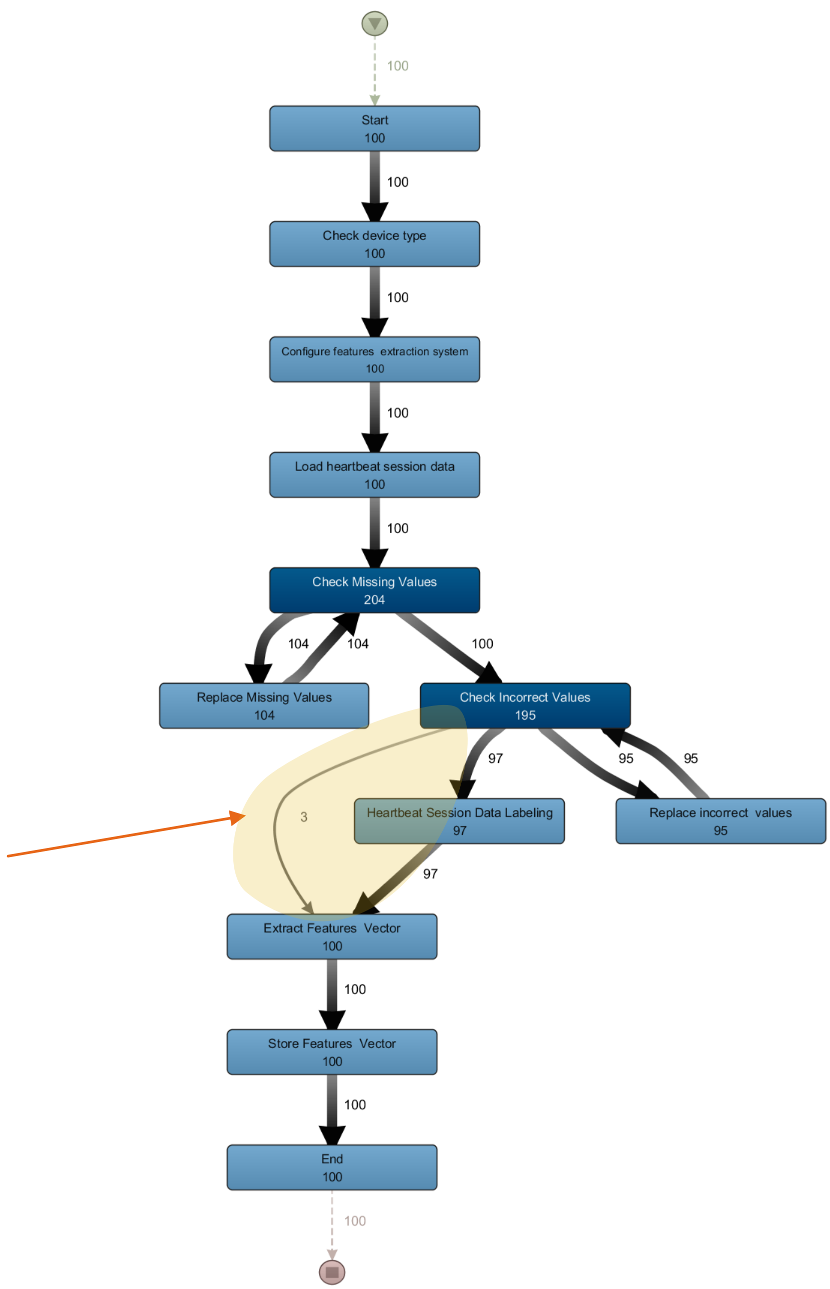
It is a measure of model complexity and it can be computed as number of gateways (g) plus number of sequence flows (f) plus number of activities (a) (**g** + **f** + **a** = **simplicity**):

- Number of gateways = 4
- Number of sequence flows = 19
- Number of activities = 12

Simplicity = 4 + 19 + 12 = 35

4.2. Prepare heartbeat session features vector (modified logs)

The BIMP generated logs have been modified by deleting (for three random different cases) the **Heartbeat Session Data Labeling** step: the idea is to simulate cases in which the data labeling task can be skipped and replaced with automatic data enrichment (as we did in the TO BE process: when there already are very similar pattern samples relative to other users, we can label new data accordingly).



By observing the Disco generated transition map reported on the left, it can be noticed that this is different with respect to the transition map mined from the unmodified logs. Specifically, there is an **additional flow** from the task *Check Incorrect Values* to the task *Extracted Features Vector* which represents the three modified cases.

Figure 35. Transition Map (Disco generated) obtained from the simulation logs is reported. Prepare heartbeat session features vector (modified logs)

4.2.1. BPMN mined with ProM (modified logs)

The Figure 36Figure 36 bellow the BPMN diagram mined with **ProM** is reported. As you can notice in the highlighted area (orange box) there is an additional edge representing the instances of modified log cases.

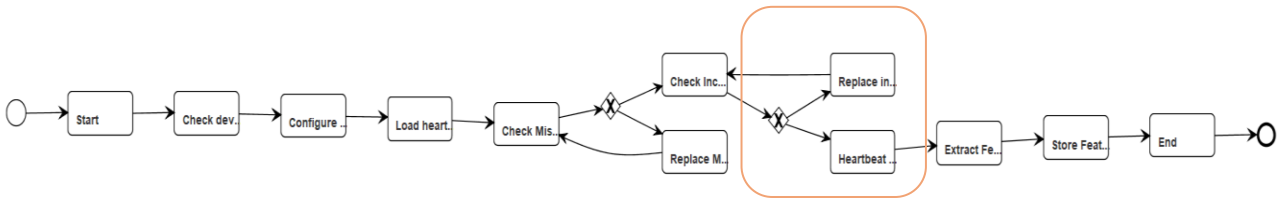


Figure 36. BPMN mined with ProM "Prepare heartbeat session features vector (modified logs)"

4.2.2. BPMN mined with Apromore (modified logs)

The Figure 37Figure 37 bellow the BPMN diagram mined with Apromore is reported. It is interesting to notice that while the Apromore tool is capable of detecting the differences in the modified logs, ProM is not. As you can notice in the highlighted area (orange box) there is an additional edge representing the 3 instances of modified log cases.

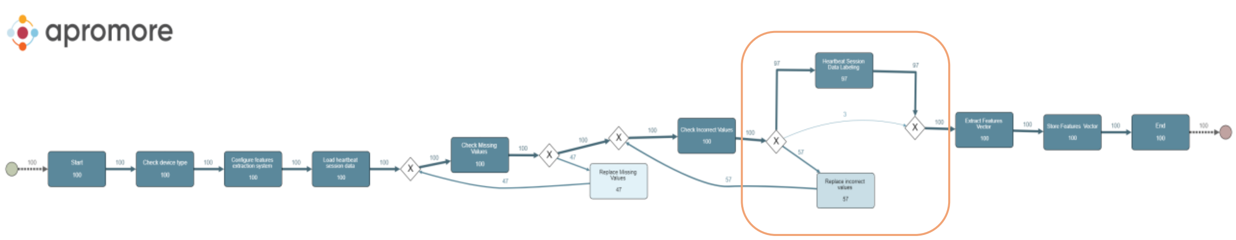


Figure 37. BPMN mined with Apromore "Prepare heartbeat session features vector (modified logs)"

4.2.3. Petri Net generated from BPMN (modified logs)

In order to compute the four process quality measures, firstly, the Petri Net is generated from the BPMN with ProM.

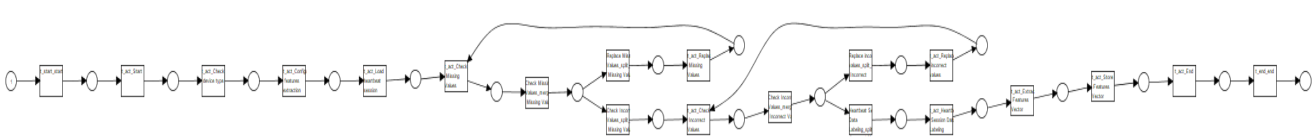


Figure 38. Petri Net generated with ProM "Prepare heartbeat session features vector (modified logs)"

4.2.4. Conformance check: quality measures (modified logs)

The first three quality measures have been computed exploiting the ProM plugins while the simplicity has been manually computed according to the formula.

- *Replay Fitness:*

We calculate the fitness of the model using the following formula:

$$f(L, M_m) = \frac{1}{2}\left(1 - \frac{\sum_{i=1}^{k} n_i m_i}{\sum_{i=1}^{k} n_i c_i}\right) + \frac{1}{2}\left(1 - \frac{\sum_{i=1}^{k} n_i r_i}{\sum_{i=1}^{k} n_i p_i}\right)$$

k	Number of different traces
n_i	Number of traces of type k in L
m_i	Number of missing tokens (artificially added) parsing i
r_i	Number of remaining tokens parsing i
c_i	Number of consumed tokens parsing i
p_i	Number of produced tokens parsing i

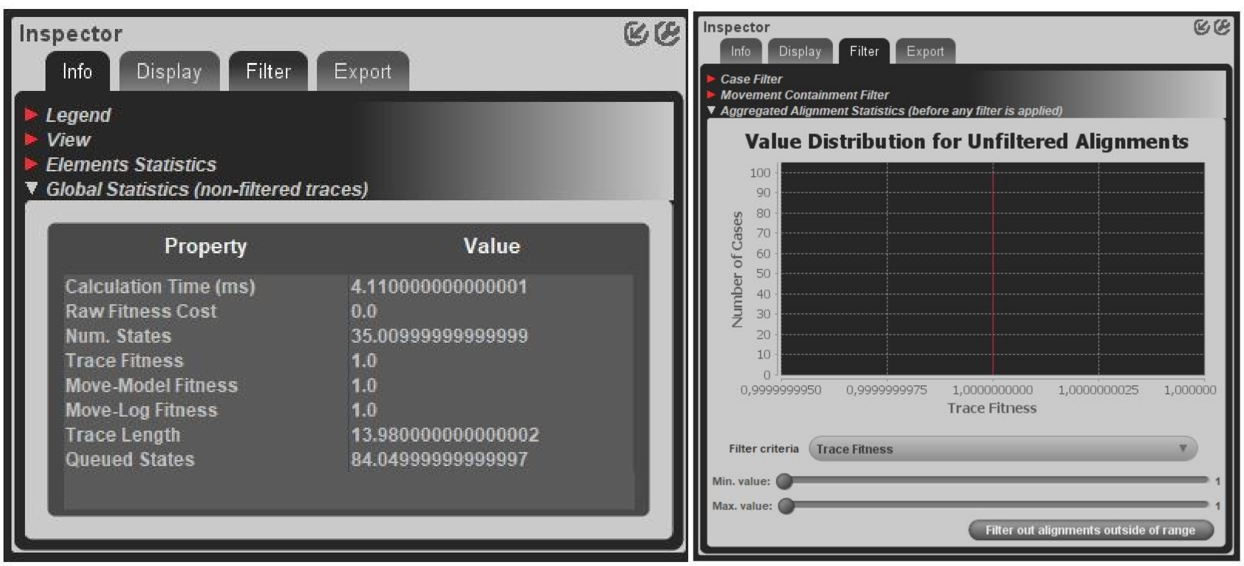


Figure 39. Fitness. Conformance check "Prepare heartbeat session features vector (modified logs)"

- *Modified log case example*

Table 17. CSV Single case representation. Case 60. (Modified log)

Event	Case ID	Activity	Start Timestamp	elementId
	60	Start	08/02/2021 10:01	BP03_BP39
	60	Check device type	08/02/2021 10:01	BP03_BP95
	60	Configure features extraction system	08/02/2021 10:01	BP03_BP97
	60	Load heartbeat session data	08/02/2021 10:01	BP03_BP99
	60	Check Missing Values	08/02/2021 10:01	BP03_BP103
	60	Replace Missing Values	08/02/2021 10:01	BP03_BP105
	60	Check Missing Values	08/02/2021 10:01	BP03_BP103
	60	Check Incorrect Values	08/02/2021 10:01	BP03_BP116
	60	Heartbeat Session Data Labeling	08/02/2021 10:01	BP03_BP124
	60	Extract Features Vector	08/02/2021 10:01	BP03_BP126
	60	Store Features Vector	08/02/2021 10:01	BP03_BP128
	60	End	08/02/2021 10:01	BP03_BP130

Figure 40. Project Alignment to log. Case 60. (Modified log)

As you can see on Figure 40Figure 40, this case have an individual trace fitness equal to 0.95 because almost their process are synchronous move, with an exception of one process (Heartbeat Session Data Labeling) that is skipped.

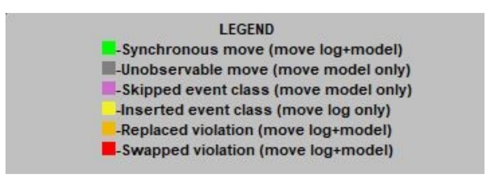


Figure 41. ProM Legend Description

4.2.5. Precision and Generalization

- *Precision*

The model should not allow for very different behavior with respect to the logs, the closer is the value to one the higher the model precision. Having a low precision value means having an underfitting model.

$$P(L,M) = \frac{1}{|\varepsilon|}\sum_{e\epsilon\varepsilon}\frac{|en_L(e)|}{|en_M(e)|}$$

ε	Collection of *unique events* in a context of the log
$en_M(e)$	Enabled activities
$en_L(e)$	Number of missing tokens (artificially added) parsing i

Precision: 0,93878

- *Generalization*

As opposed to precision, a model with a good generalization value is a model allowing some abstraction. (often there is noise or missing values in the logs). The higher the generalization value lower the model overfitting.

Generalization: 0,99849

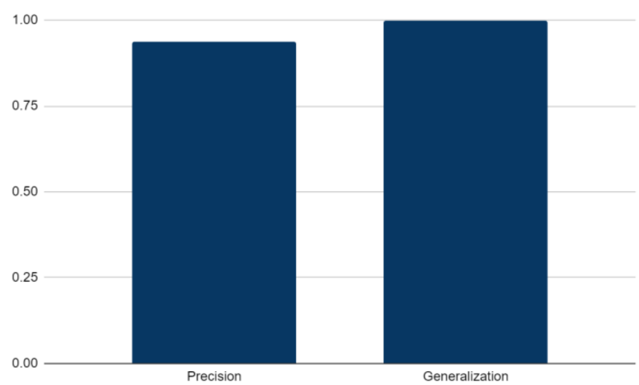


Figure 42. Precision and Generalization for "Prepare heartbeat session features vector (modified logs)"

4.2.6. Simplicity

It is a measure of model complexity and it can be computed as number of gateways (g) plus number of sequence flows (f) plus number of activities (a) (**g + f + a = simplicity**):

- Number of gateways = 5
- Number of sequence flows = 20
- Number of activities = 12

Simplicity = 5 + 20 + 12 = 37

4.2.7. Comparison between Unmodified logs and Modified logs

- *Precision and Generalization*

The Precision and Generalization values did not change significantly with respect to the original process. This may be due to the fact that the modifications introduced are not "**strong**" enough to produce substantial variations.

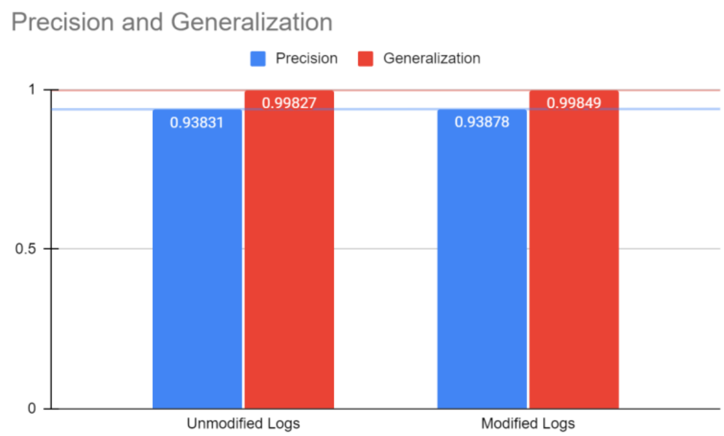


Figure 43. Precision and Generalization. Unmodified logs vs Modified logs. Prepare heartbeat session features vector

- *Simplicity*

A change of three cases produces as a result a variation on the simplicity with a higher value on modified logs with respect to an unmodified log.

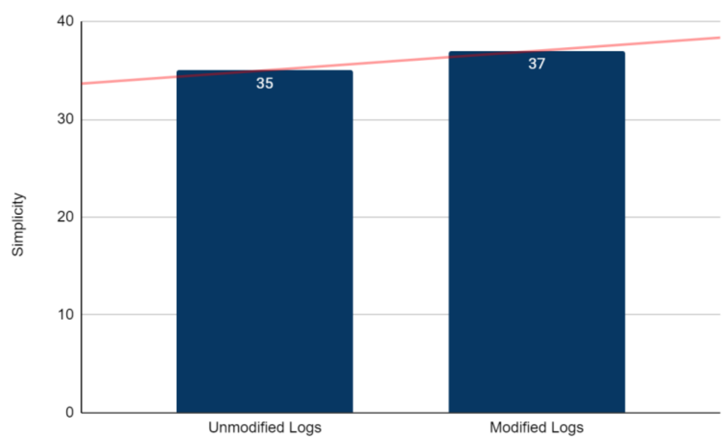


Figure 44. Simplicity. Unmodified logs vs Modified logs. Prepare heartbeat session features vector

4.3. Merged Process (unmodified logs)

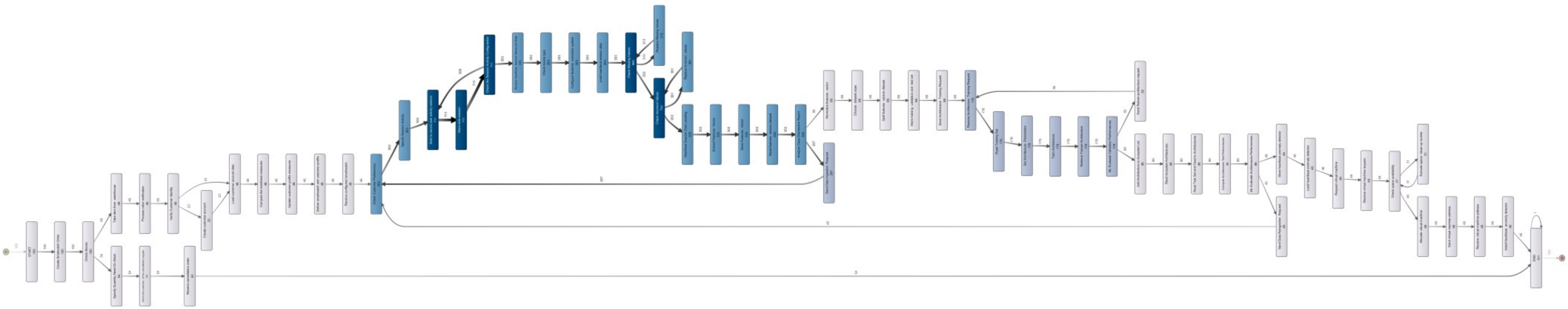


Figure 45. Transition Map (Disco generated) obtained from the simulation logs is reported. Merged Process (unmodified logs)

In order to derive a more complex business process, we merged together all the business processes starting from "*Configure and Deliver Anomaly Detection Smartwatch*" *until "Evaluate Heartbeat Anomaly Detection Architecture".* The resulting process is quite complex with a large number of activities and paths. On this Process Mining we consider the next process:

- Configure and deliver anomaly detection smartwatch
- Track heart beat session for model training
- Prepare heart beat session features vector
- Prepare heart beat session learning sets
- Train heartbeat anomaly detection architecture
- Evaluate heartbeat anomaly detection architecture
- Deploy heartbeat anomaly detector

4.3.1. BPMN mined with ProM (unmodified logs)

The Figure 46Figure 46 bellow the BPMN diagram mined with ProM is reported. The dimension and complexity of this graph is more interesting to study and compare on the next section on the modified logs.

Figure 46. BPMN mined with ProM "Merged Process (unmodified logs)"

4.3.2. BPMN mined with Apromore (unmodified logs)

The Figure 47Figure 47 bellow the BPMN diagram mined with Apromore is reported. Given the large size of the BPMN diagram, by looking at this document it is hard to understand whether the Apromore and the ProM generated BPMN are equal or not, we checked within the tools and the graphs are actually equal.

Figure 47. BPMN mined with Apromore "Merged Process (unmodified logs)"

4.3.3. Petri Net generated from BPMN (unmodified logs)

In order to compute the four process quality measures, firstly, the Petri Net is generated from the BPMN with ProM.

Figure 48. Petri Net generated with ProM "Merged Process (unmodified logs)"

4.3.4. Conformance check: quality measures

The first three quality measures have been computed exploiting the ProM plugins while the simplicity has been manually computed according to the formula.

- *Replay Fitness:*

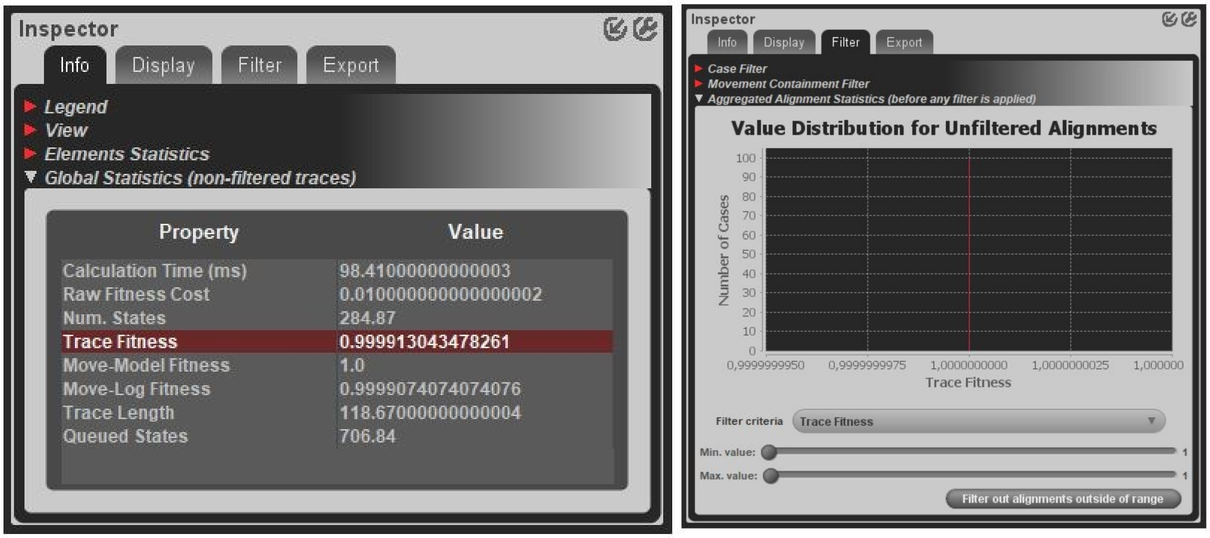


Figure 49. Fitness. Conformance check "Merged Process (unmodified logs)"

We would have expected to obtain a trace fitness value of 1 but **the actual value is 0.9999**: as you can notice on Figure 49 and Figure 50 from the statistics pieces of information reported, there is **one single case out of 100** having a trace fitness value of 0.99 (**case id 23**).

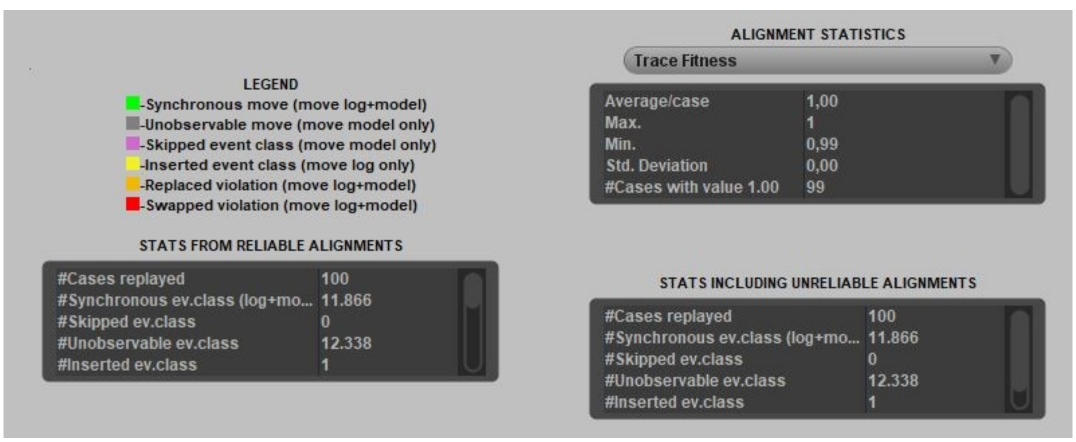


Figure 50. Statistics Resume. Merged Process (unmodified logs)

The case number 23 has an *Inserted event* violation: as last activity you can notice a yellow arrow, the name of this added activity is "END" so it looks as though for this case there are two consecutive end events.

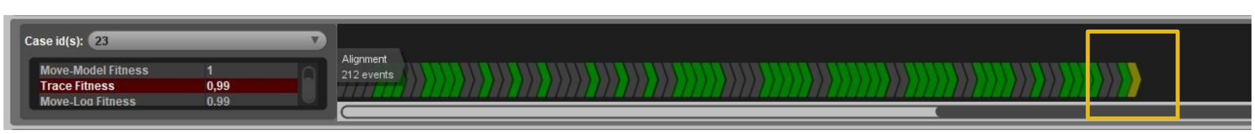


Figure 51. Project Alignment to log. Case 23 Merged Process (unmodified logs)

4.3.5. Precision and Generalization:

- *Precision*

The model should not allow for very different behavior with respect to the logs, the closer is the value to one the higher the model precision. Having a low precision value means having an underfitting model.

Precision: 0,85472

The precision value obtained is not very high: it makes sense as the process under analysis is quite complex and it comprises many different possible paths enabling different behaviors from those appearing in the logs. Therefore, we can conclude that the model is not extremely precise.

- *Generalization*

As opposed to precision, a model with a good generalization value is a model allowing some abstraction. (often there is noise or missing values in the logs). The higher the generalization value lower the model overfitting.

Generalization: 0,99967

The model achieves a very good Generalization score: it is coherent with the precision value obtained; indeed, the model is not overly precise and it retains some abstraction capability, imposing not too much constraints on the behavior.

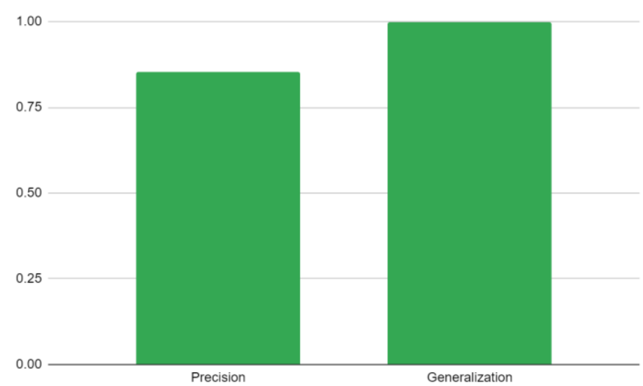


Figure 52. Precision and Generalization for "Merged Process (unmodified logs)"

4.3.6. Simplicity

It is a measure of model complexity and it can be computed as number of gateways (g) plus number of sequence flows (f) plus number of activities (a) (**g** + **f** + **a** = **simplicity**):

- Number of gateways = 17
- Number of sequence flows = 87
- Number of activities = 61

Simplicity = 17 + 87 + 61 = 165

4.4. Merged Process (modified logs)

Below are shown both the transition maps of the original process (Figure 54), so as to enable a first visual comparison.

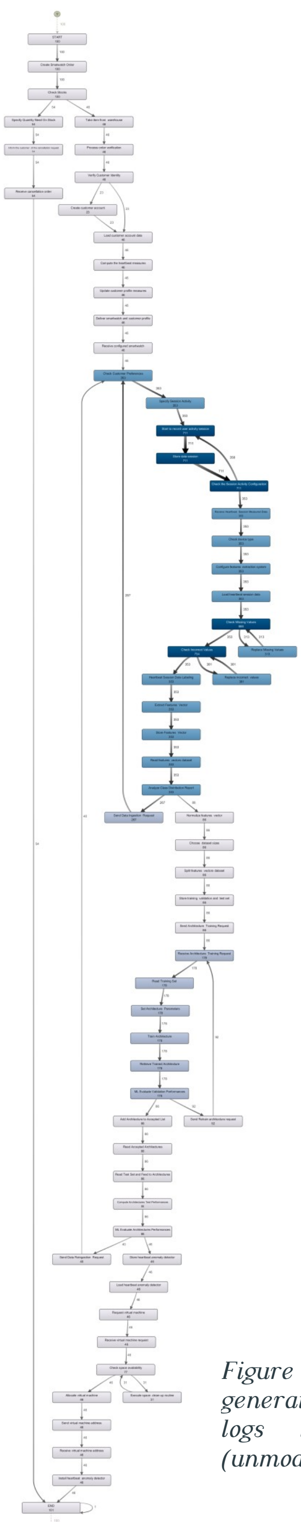


Figure 54. Transition Map (Disco generated) obtained from the simulation logs is reported. Merged Process (unmodified logs)

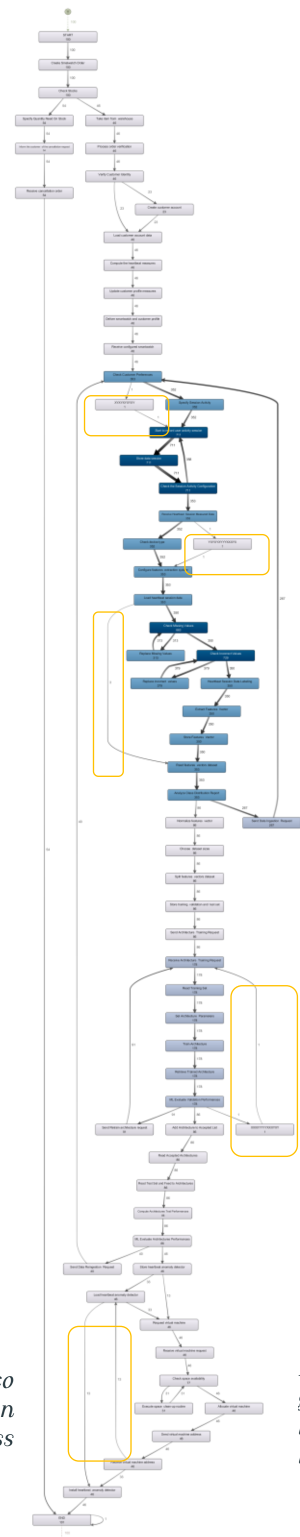


Figure 53. Transition Map (Disco generated) obtained from the simulation logs is reported. Merged Process (modified logs)

4.4.1. Modifications applied to the logs

1. As first modification simulation, we deleted from **three cases** the set of activities related to data labeling and preprocessing. The idea is to simulate some cases in which the system automatically performs data augmentation when it is possible: if for a certain data sample there is already a sufficient number of very similar cases, there is no need to analyze and manually label additional samples but more samples can be generated through automatic data augmentation. The set of cases with this modified are:

    - [7,3, 96]

2. As second modification simulation, we **swapped in the logs the order of execution of some activities**. Specifically, we swapped the execution order between the task "*Load heartbeat anomaly detector*" and the set of tasks related to the virtual machine allocation for the installation of the deployed ML architecture. The normal behavior consists in first loading the architecture, then checking for the virtual machine availability and then installing the architecture. in the **cases** we reversed this:

    - [96, 54, 97]

3. As third modification, we randomly replaced some event names with purposely meaningless names so as to simulate noise in the logs. We applied this modification to **cases:**

    - [25, 28, 89]

4. As fourth modification, we carried out modification number 1 on 10 additional cases:

    - [6, 8, 0, 51, 94, 14, 37, 38, 19, 18]

4.4.2. BPMN mined with ProM (modified logs)

Figure 55. BPMN mined with ProM "Merge Process (modified logs)"

4.4.3. BPMN mined with Apromore (modified logs)

The complexity is high and by zooming it can be noticed that the BPMN diagrams generated first with ProM and then with Apromore are no different.

Figure 56. BPMN mined with Apromore "Merge Process (modified logs)"

4.4.4. Petri Net generated from BPMN (modified logs)

Figure 57. Petri Net generated with ProM "Merge Process (modified logs)"

4.4.5. Conformance check: quality measures

The first three quality measures have been computed exploiting the ProM plugins while the simplicity has been manually computed according to the formula.

- *Replay Fitness:*

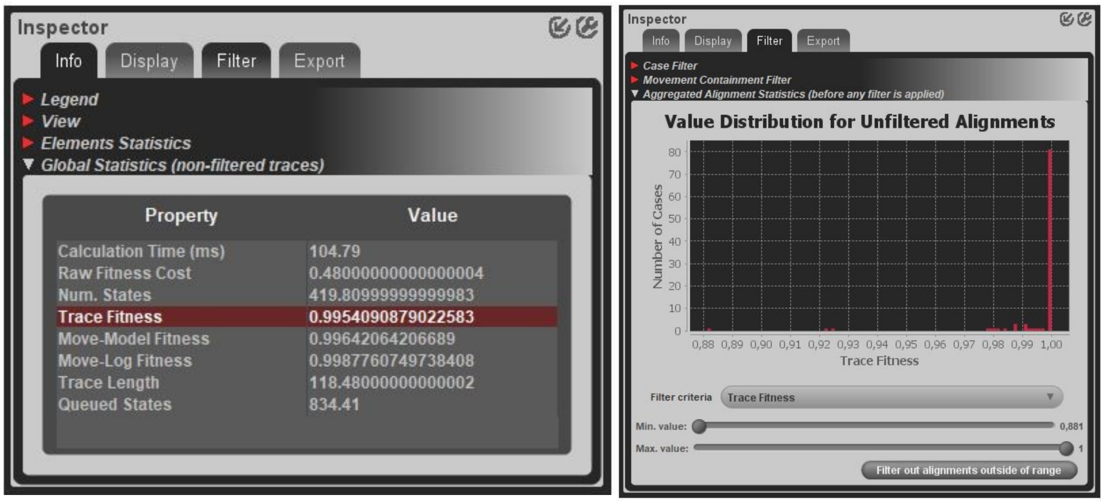


Figure 58. Fitness. Conformance check "Merged Process (modified logs)"

The trace fitness value slightly decreased, from 0,9999 to 0,9954, as expected. This is due to the log modifications introduced. As it can be noticed from the Trace Fitness Alignment Statistics (on the right) there are now 81 cases out of 100 having a perfect trace fitness score: the remaining 19 cases corresponds to the ones we modified. The minimum trace fitness value is 0,88.

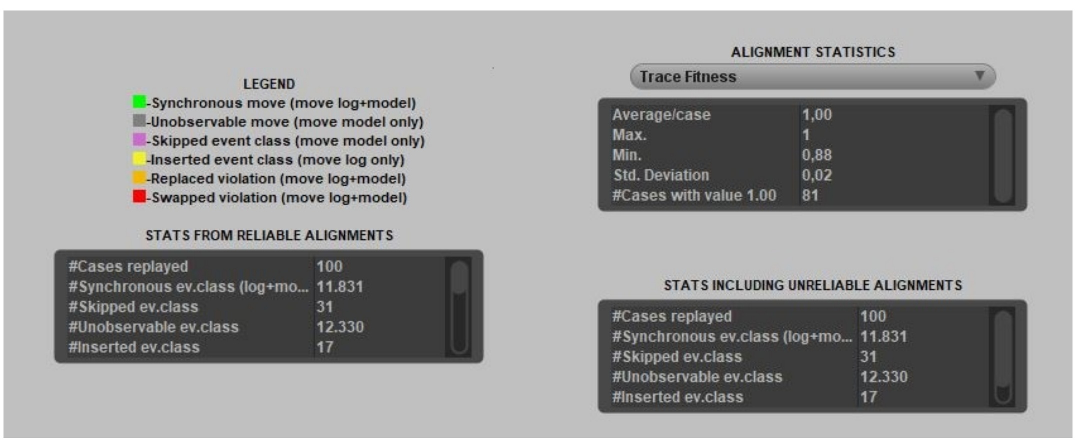
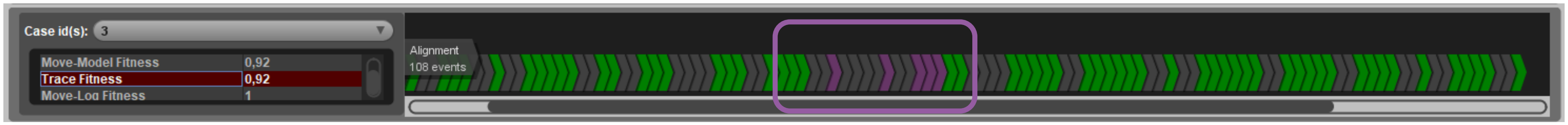


Figure 59 Statistics Resume. Merged Process (modified logs)

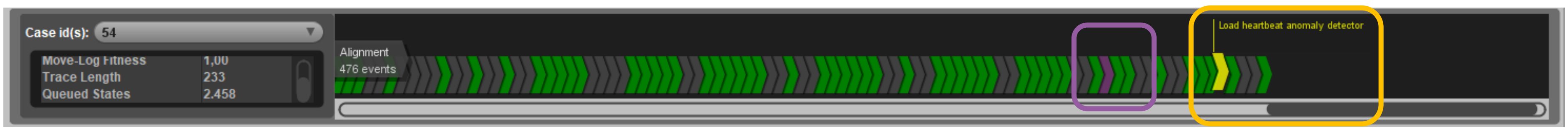
- *Modification #1 case log-model Alignment example:*

As an example, the case one is reported. In the violet highlighted area, you can notice the **skipped** set of activities related to data labeling and preprocessing.



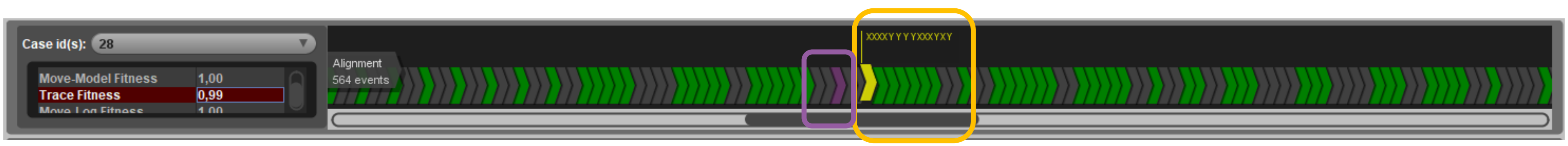
- *Modification #2 case log-model Alignment example:*

As an example, the case 54 is reported. In the yellow highlighted area, you can notice the swapped execution order between the task "*Load heartbeat anomaly detector*" and the set of tasks related to the virtual machine allocation for the installation of the deployed ML architecture. The **first violet arrow corresponds to the skipped event class** "*Load heartbeat anomaly detector*" and **the last yellow arrow correspond to the Inserted event class** of the same activity.



- *Modification #3 case log-model Alignment example:*

As an example, the case 28 is reported. In the yellow highlighted area, you can notice we randomly replaced an activity with the purposely meaningless string "*XXXXYYYYXXXYXY*" so as to simulate noise in the logs.



4.4.6. Precision and Generalization

- *Precision*

The model should not allow for very different behavior with respect to the logs, the closer is the value to one the higher the model precision. Having a low precision value means having an underfitting model.

Precision: 0,85484

The precision value obtained is not very high: it makes sense as the process under analysis is quite complex and it comprises many different possible paths enabling different behaviors from those appearing in the logs.

- *Generalization*

As opposed to precision, a model with a good generalization value is a model allowing some abstraction. (often there is noise or missing values in the logs). The higher the generalization value lower the model overfitting.

Generalization: 0,99965

The Precision and Generalization values did not change significantly with respect to the original process. This may be due to the fact that the modifications introduced are not "strong" enough to produce substantial variations.

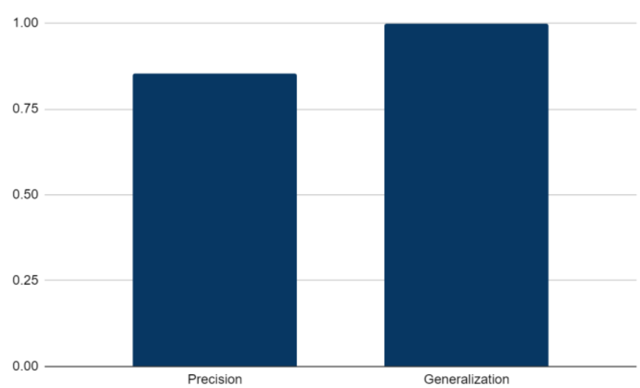


Figure 60. Precision and Generalization for "Merge Process (modified logs)"

- *Simplicity:*

It is a measure of model complexity and it can be computed as number of gateways (g) plus number of sequence flows (f) plus number of activities (a) (**g** + **f** + **a** = **simplicity**):

- Number of gateways = 29
- Number of sequence flows = 112
- Number of activities = 64

Simplicity = 29 + 112 + 64 = **205**

4.4.7. Comparison between Unmodified logs and Modified logs

- *Precision and Generalization*

The Precision and Generalization values did not change significantly with respect to the original process. This may be due to the fact that the modifications introduced are not “strong” enough to produce substantial variations.

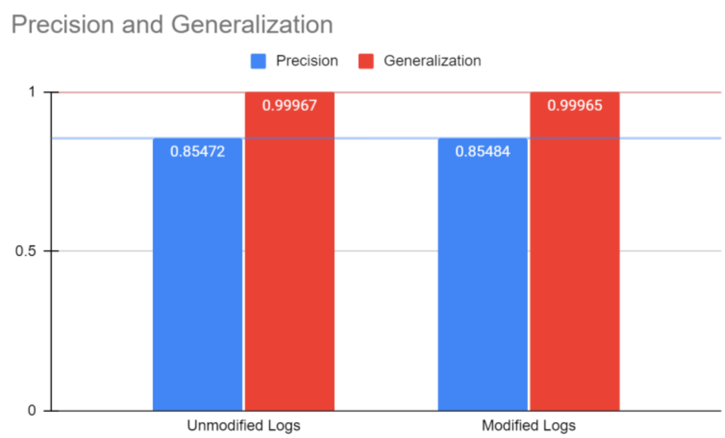


Figure 61. Precision and Generalization. Unmodified logs vs Modified logs. Merged Process

- *Simplicity*

A change of three cases produces as a result a variation on the simplicity with a higher value on modified logs with respect to an unmodified log.

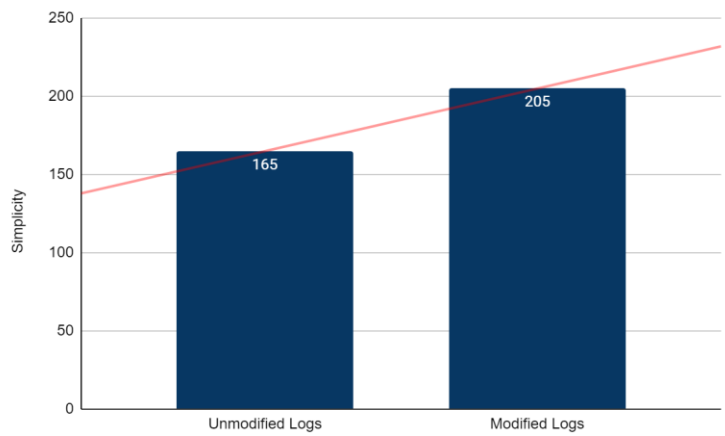


Figure 62. Simplicity. Unmodified logs vs Modified logs. Merged Process

The simplicity values increased significantly with respect to the original process. Specifically, it went **from 165 to 205 (25% increase)**: the number of gateways increased **from 17 to 29 (70% increase)**, the number of sequence flows **from 87 to 112 (29% increase)** and the number of activities increases **from 61 to 64 (6% increase).**

5. Clockify Report

5.1. Team Collaborative Tasks

This chapter is for administrative control. The team work on project "*Smartwatch based electrocardiogram analysis*" from November 24th, 2020 until February 11th, 2021 completing the 100% of time and objectives proposed by the professor.

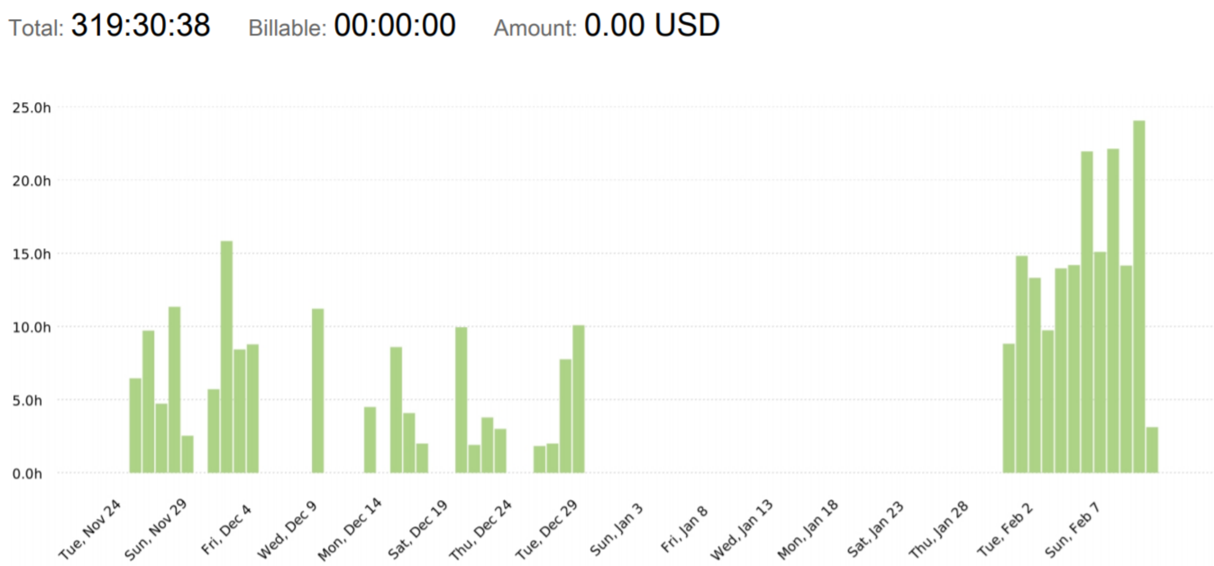


Figure 63. Clockify Hour Control. Team Collaborative Tasks Summary Report

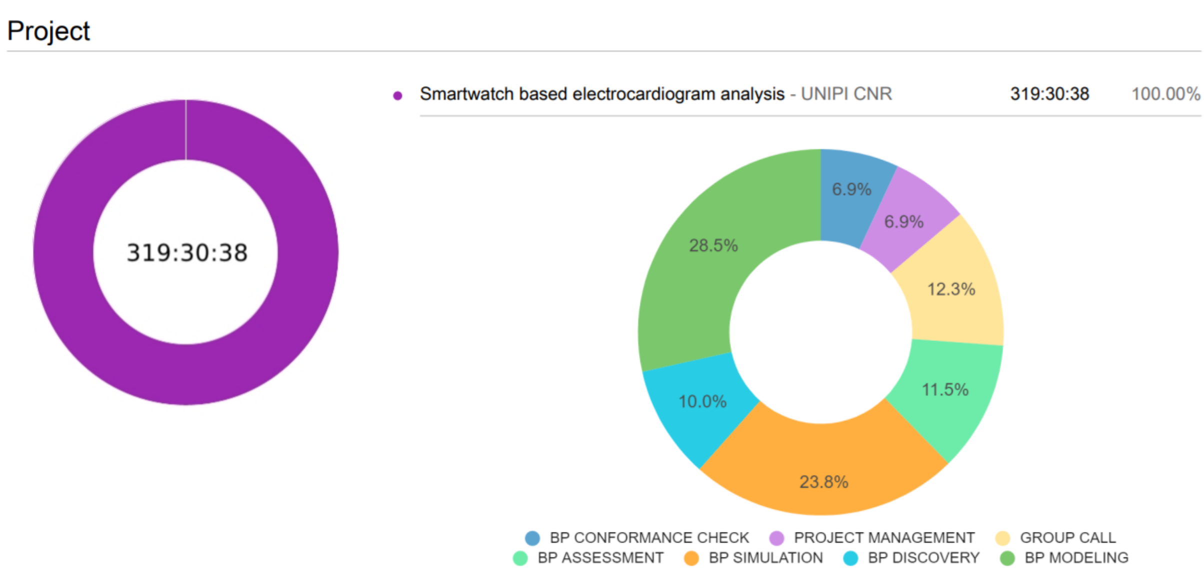


Figure 64. Clockify Hour Control. Team Collaborative Tasks Time

5.1.1. Alessio Schiavo

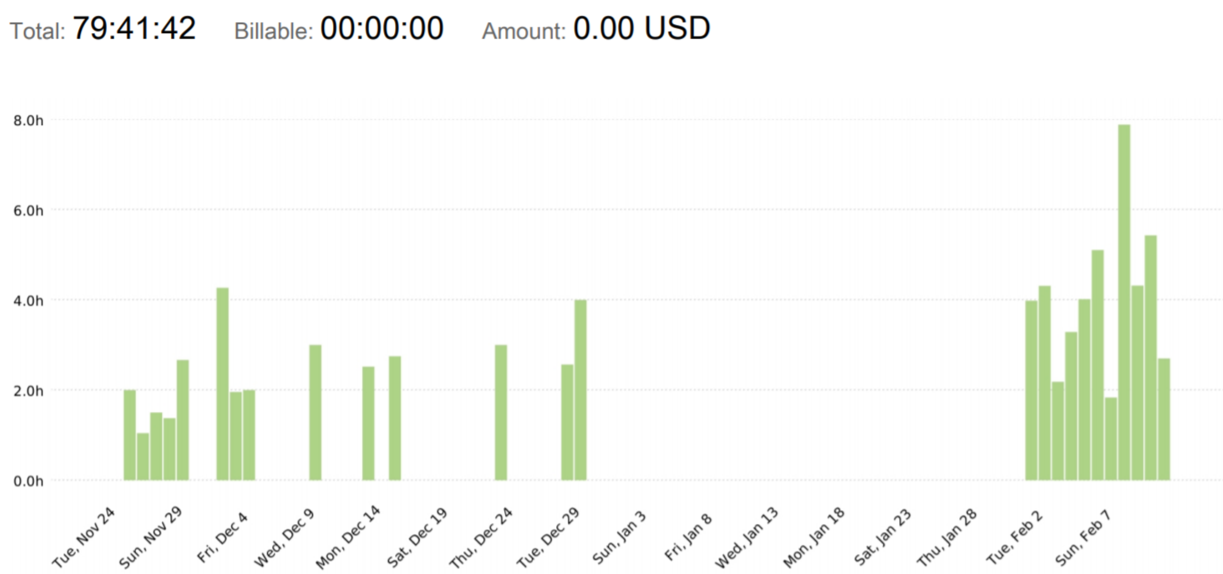


Figure 65. Clockify Hour Control. Summary Report Alessio Schiavo

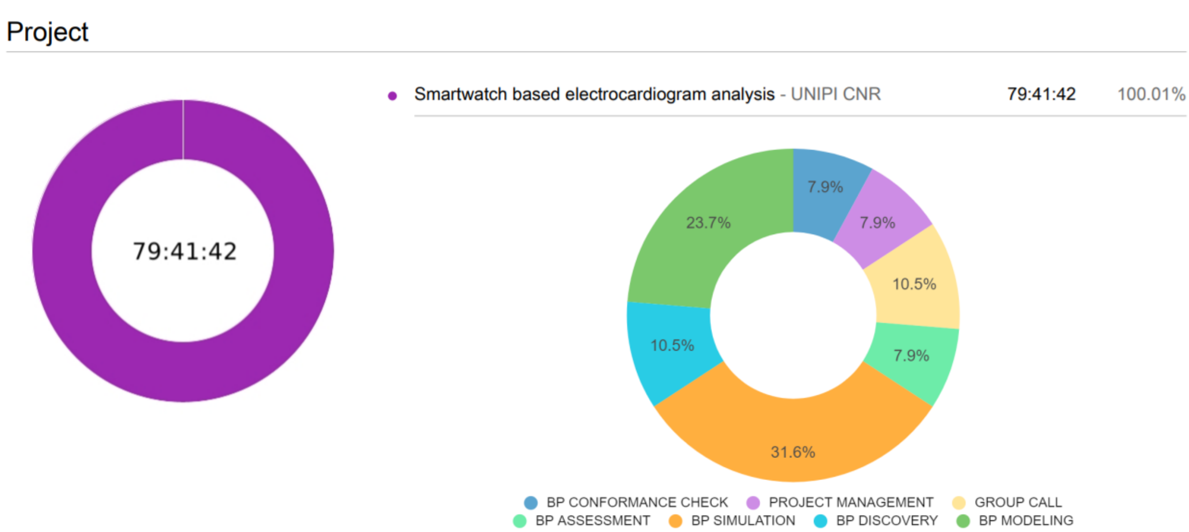


Figure 66. Clockify Hour Control. Project Chart Report Alessio Schiavo

Table 18. Project Description. Alessio Schiavo

Description	Duration	Percentage
Group Call #4	2:42:00	3.4%
Conformance Check "MERGED PROCESS" #2	3:25:52	4.3%
Conformance Check "MERGED PROCESS" #1	2:00:00	2.5%
Simulate "MERGED PROCESS" #5	1:49:00	2.3%
Simulate "MERGED PROCESS" #4	2:30:00	3.1%
Simulate "MERGED PROCESS" #3	5:33:15	7.0%
Simulate "MERGED PROCESS" #2	2:20:00	2.9%
Simulate "MERGED PROCESS" #1	1:50:00	2.3%
Discovery "MERGED PROCESS" #3	3:30:00	4.4%
Discovery "MERGED PROCESS" #2	1:36:20	2.0%
Discovery "MERGED PROCESS" #1	2:39:18	3.3%
Conformance Check "Heartbeat Detector Performance Evaluation System"	1:21:41	1.7%
Simulate "Heartbeat Detector Performance Evaluation System" #3	1:00:00	1.3%
Simulate "Heartbeat Detector Performance Evaluation System" #2	2:17:14	2.9%
Simulate "Heartbeat Detector Performance Evaluation System" #1	2:10:56	2.7%
Discovery "Heartbeat Detector Performance Evaluation System"	2:16:00	2.8%
Simulate the TO BE model #2	1:00:00	1.3%
Simulate the TO BE model #1	1:02:41	1.3%
Group Call #3	3:02:17	3.8%
Simulate the AS IS Model #2	0:56:34	1.2%
Simulate the AS IS Model #1	2:16:46	2.9%
Create use case for "Analyze Class Distribution Report" #3	1:43:06	2.2%
Create use case for "Analyze Class Distribution Report" #2	1:40:54	2.1%
Create use case for "Analyze Class Distribution Report" #1	0:53:02	1.1%
Documentation. Diagram Services	3:00:00	3.8%
Group Call #2	2:45:00	3.5%
Define "Prepare Heartbeat Session Learning Sets"	2:31:04	3.2%
Define "Prepare Heartbeat Session Features Vector" #2	3:00:00	3.8%
Define "Prepare Heartbeat Session Features Vector" #1	2:00:00	2.5%
Define "Train Heartbeat Anomaly Detection Architecture" #2	1:57:25	2.5%
Define "Train Heartbeat Anomaly Detection Architecture" #1	1:30:59	1.9%
Documentation. Process Landscape Definition	2:45:03	3.5%
Group Call #1	2:40:00	3.4%
Process Landscape Definition #2	1:22:34	1.7%
Process Landscape Definition #1	0:30:00	0.6%
Work Breakdown On Process Landscape #3	1:00:00	1.3%
Work Breakdown On Process Landscape #2	1:02:41	1.3%
Work Breakdown On Process Landscape #1	2:00:00	2.5%

5.1.2. Mattia Daole

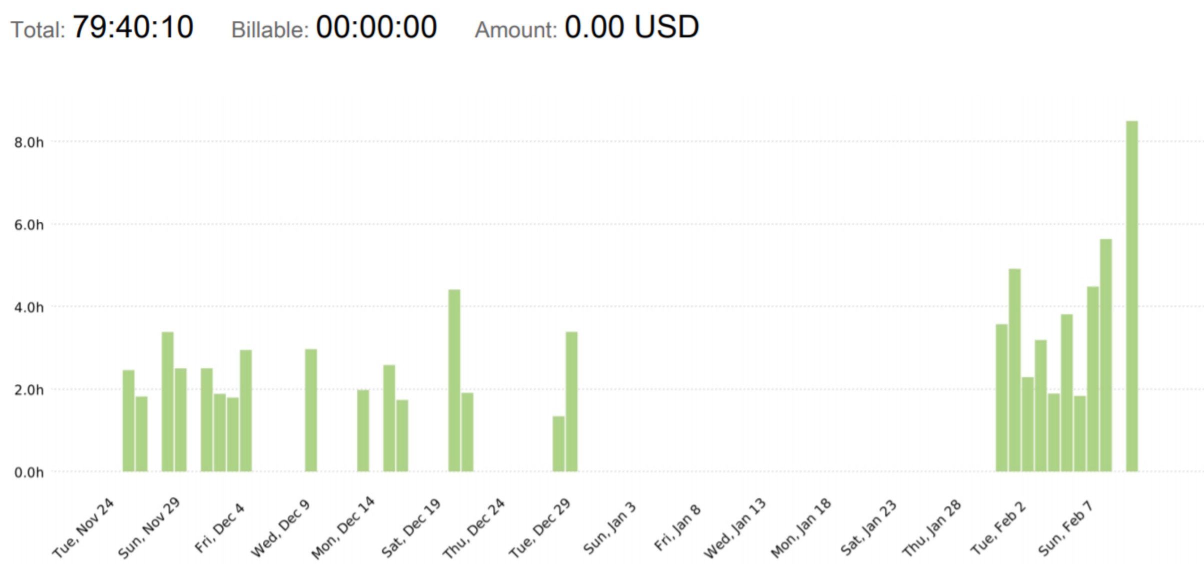


Figure 67. Clockify Hour Control. Summary Report Mattia Daole

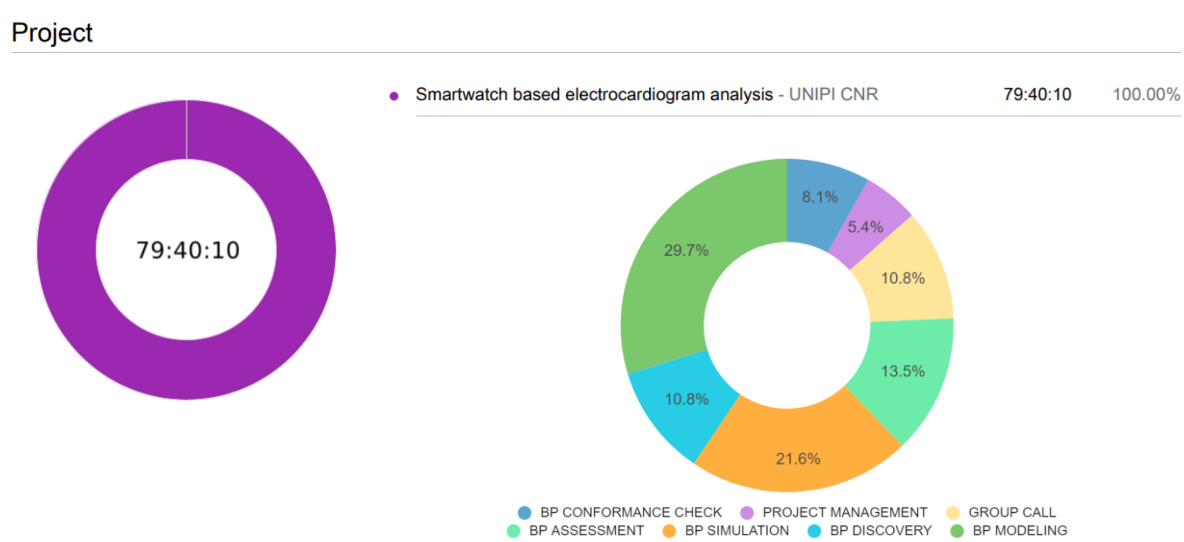


Figure 68. Clockify Hour Control. Project Chart Report Mattia Daole

Table 19. Project Description. Mattia Daole

Description	Duration	Percentage
Conformance Check "MERGED PROCESS"	3:36:00	4.5%
Simulate "MERGED PROCESS" #4	1:53:42	2.4%
Group Call #4	3:00:00	3.8%
Simulate "MERGED PROCESS" #2	2:45:20	3.5%
Simulate "MERGED PROCESS" #1	2:52:39	3.6%
Discovery "MERGED PROCESS" #2	2:31:13	3.2%
Discovery "MERGED PROCESS" #1	1:57:45	2.5%
Conformance Check "Heartbeat Detector Performance Evaluation System" #2	1:49:56	2.3%
Conformance Check "Heartbeat Detector Performance Evaluation System" #1	2:10:56	2.7%
Simulate "Heartbeat Detector Performance Evaluation System" #2	1:37:43	2.0%
Simulate "Heartbeat Detector Performance Evaluation System" #1	1:53:22	2.4%
Discovery "Heartbeat Detector Performance Evaluation System" #2	3:11:07	4.0%
Discovery "Heartbeat Detector Performance Evaluation System" #1	0:56:15	1.2%
Simulate the TO BE model	1:21:05	1.7%
Group Call #3	3:44:16	4.7%
Simulate the AS IS Model #2	1:10:28	1.5%
Simulate the AS IS Model #1	3:34:11	4.5%
Documentation. UML Cases	2:18:00	2.9%
Create use case for "Evaluate Architecture Performance" on Evaluation #4	1:05:03	1.4%
Create use case for "Evaluate Architecture Performance" on Evaluation #3	1:20:16	1.7%
Create use case for "Evaluate Architecture Performance" on Evaluation #2	1:54:22	2.4%
Create use case for "Evaluate Architecture Performance" on Evaluation #1	2:49:03	3.5%
Documentation. Diagram Services	1:35:26	2.0%
Define "Evaluate Heartbeat Anomaly Detection Model" #3	1:44:08	2.2%
Group Call #2	2:34:48	3.2%
Define "Evaluate Heartbeat Anomaly Detection Model" #2	1:58:32	2.5%
Define "Evaluate Heartbeat Anomaly Detection Model" #1	2:57:53	3.7%
Define "Deploy Heartbeat Anomaly Detector" #2	2:56:42	3.7%
Define "Deploy Heartbeat Anomaly Detector" #1	1:47:34	2.2%
Documentation. Process Landscape Definition	1:52:58	2.4%
Process Landscape Definition #4	2:30:06	3.1%
Group Call #1	2:30:00	3.1%
Process Landscape Definition #3	2:00:01	2.5%
Process Landscape Definition #2	1:22:48	1.7%
Process Landscape Definition #1	1:49:03	2.3%
Work Breakdown on Process Landscape #2	1:38:29	2.1%
Work Breakdown on Process Landscape #1	0:49:00	1.0%

5.1.3. Filippo Minutella

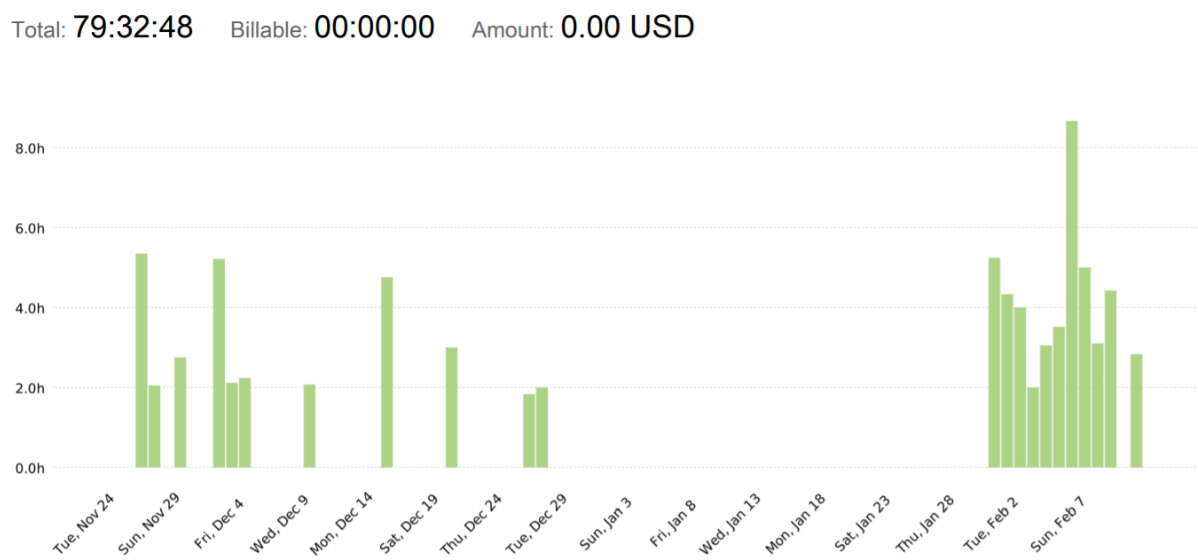


Figure 69. Clockify Hour Control. Summary Report Filippo Minutella

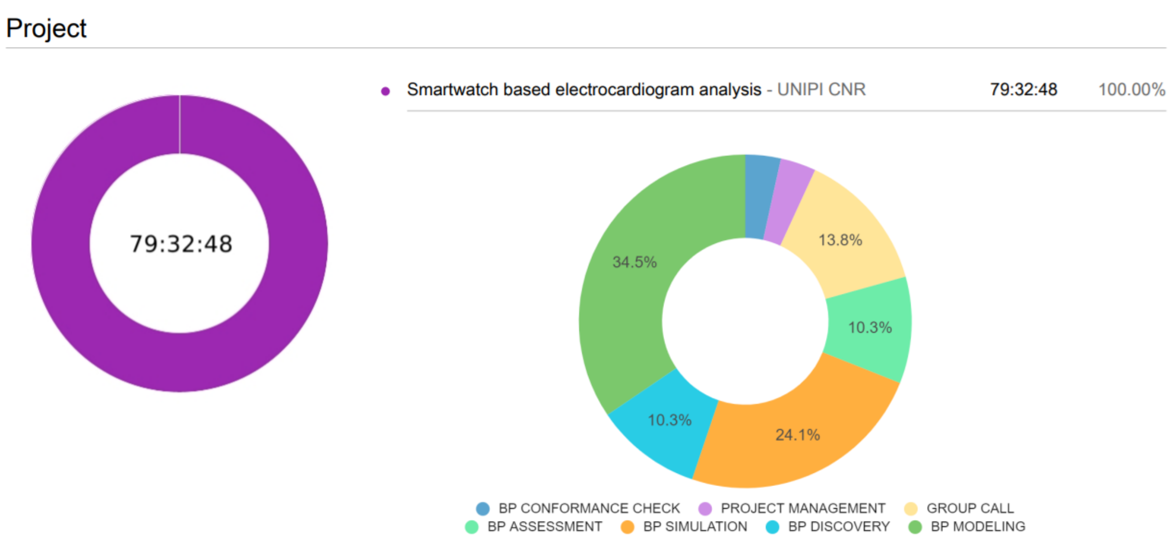


Figure 70. Clockify Hour Control. Project Chart Report Filippo Minutella

Table 20. Project Description. Filippo Minutella

Description	Duration	Percentage
Group Call #4	2:50:00	3.6%
Simulate "MERGED PROCESS"	2:40:20	3.4%
Discovery "MERGED PROCESS"	1:45:07	2.2%
Conformance Check "Heartbeat Detector Performance Evaluation System"	3:06:07	3.9%
Simulate "Heartbeat Detector Performance Evaluation System"	1:00:00	1.3%
Discovery "Heartbeat Detector Performance Evaluation System" #2	4:00:00	5.0%
Discovery "Heartbeat Detector Performance Evaluation System" #1	5:33:00	7.0%
Simulate the TO BE model #3	3:07:00	3.9%
Simulate the TO BE model #2	3:31:00	4.4%
Simulate the TO BE model #1	3:03:00	3.8%
Simulate the AS IS Model #2	2:00:00	2.5%
Simulate the AS IS Model #1	4:00:00	5.0%
Group Call #3	3:10:00	4.0%
Documentation. UML Cases	1:10:00	1.5%
Create Table Salaries #2	1:42:33	2.1%
Create Table Salaries #1	3:32:00	4.4%
Create use case for Task "Evaluate Architecture Performance" on Performance Report #2	2:00:00	2.5%
Create use case for Task "Evaluate Architecture Performance" on Performance Report #1	1:50:00	2.3%
Documentation. Diagram Services	3:00:00	3.8%
Define "Produce Heartbeat anomaly Report" #2	2:45:34	3.5%
Group Call #2	2:00:00	2.5%
Define "Produce Heartbeat anomaly Report" #1	2:04:22	2.6%
Define "Produce Heartbeat Anomaly Detection Performance Report" #2	2:14:00	2.8%
Define "Produce Heartbeat Anomaly Detection Performance Report" #1	2:07:00	2.7%
Documentation. Process Landscape Definition	5:12:50	6.6%
Group Call #1	2:45:00	3.5%
Process Landscape Definition #2	2:02:55	2.6%
Process Landscape Definition #1	2:31:00	3.2%
Work Breakdown On Process Landscape	2:50:00	3.6%
Group Call #4	2:50:00	3.6%

5.1.4. Marsha Gómez

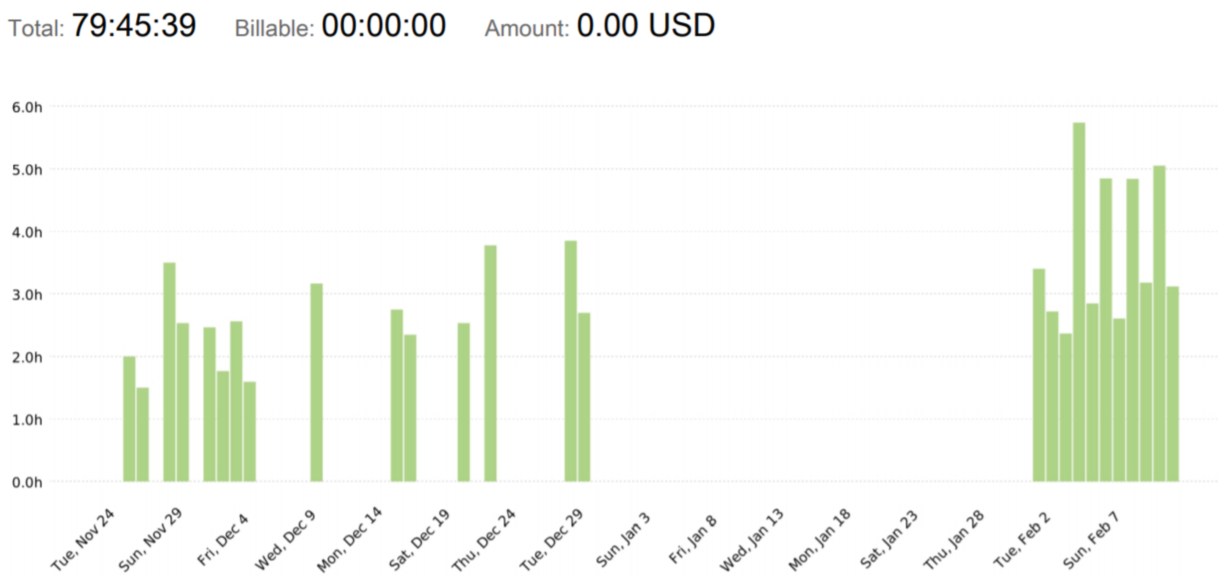


Figure 71. Clockify Hour Control. Summary Report Marsha Gómez

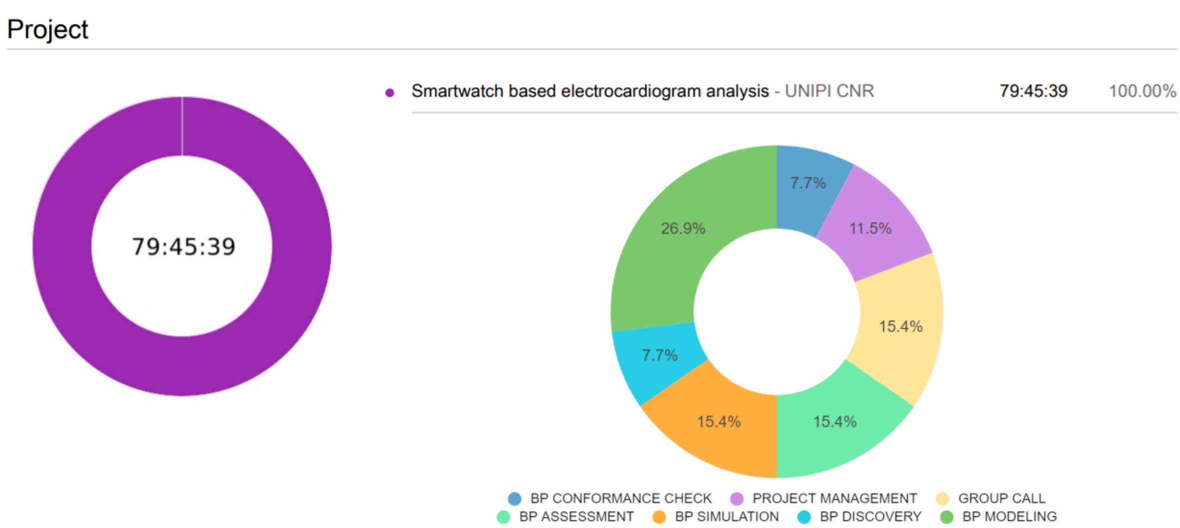


Figure 72. Clockify Hour Control. Project Chart Report Marsha Gómez

Table 21. Project Description. Marsha Gómez

Description	Duration	Percentage
Group Call #4	3:07:10	3.9%
Conformance Check "MERGED PROCESS"	5:03:07	6.3%
Simulate "MERGED PROCESS"	3:10:50	4.0%
Discovery "MERGED PROCESS"	4:50:27	6.1%
Conformance Check "Heartbeat Detector Performance Evaluation System"	2:36:24	3.3%
Simulate "Heartbeat Detector Performance Evaluation System"	4:50:50	6.1%
Discovery "Heartbeat Detector Performance Evaluation System"	2:50:50	3.6%
Simulate the TO BE model	3:10:11	4.0%
Simulate the AS IS Model	2:34:10	3.2%
Documentation. UML Cases	2:21:59	3.0%
Create use case for "Heartbeat Session data labelling"	2:43:09	3.4%
Group Call #3	3:24:09	4.3%
Documentation. Diagram Services	2:41:50	3.4%
Define "Configure and Deliver Anomaly Detection Smartwatch" #2	3:51:00	4.8%
Define "Configure and Deliver Anomaly Detection Smartwatch" #1	3:46:37	4.7%
Define "Track Heart Beat session for Model Training" #2	2:32:00	3.2%
Define "Track Heart Beat session for Model Training" #1	2:20:47	2.9%
Group Call #2	2:45:00	3.4%
Documentation. Process Landscape Definition	3:10:00	4.0%
Process Landscape Definition #4	1:35:40	2.0%
Process Landscape Definition #3	2:33:39	3.2%
Process Landscape Definition #2	1:45:55	2.2%
Process Landscape Definition #1	2:27:55	3.1%
Group Call #1	2:32:00	3.2%
Work Breakdown on Process Landscape #3	3:30:00	4.4%
Work Breakdown on Process Landscape #2	1:30:00	1.9%
Work Breakdown on Process Landscape #1	2:00:00	2.5%