University of Pisa

Marine Plastic Detection

Jacopo Cecchetti

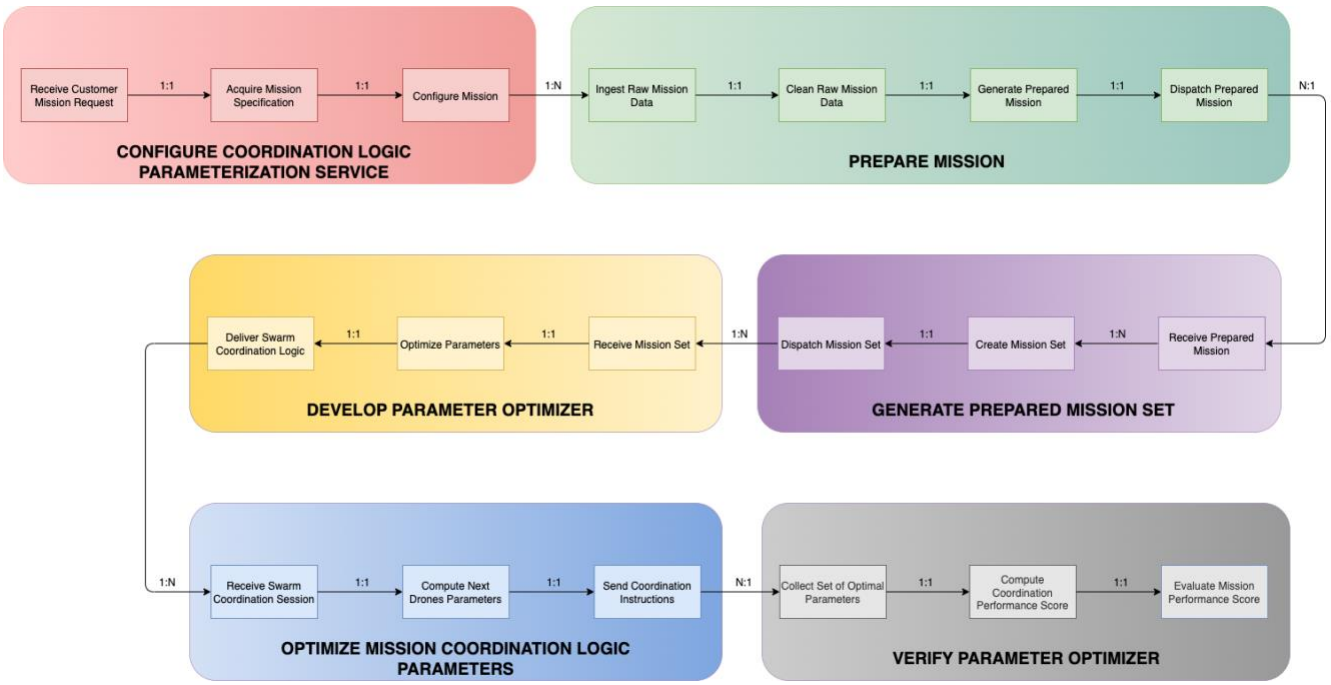
Franco Terranova

Matteo Del Seppia

Federico Minniti

Summary

1 PROCESS LANDSCAPE

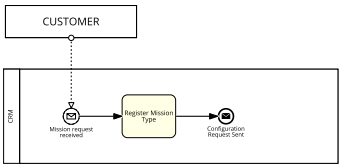


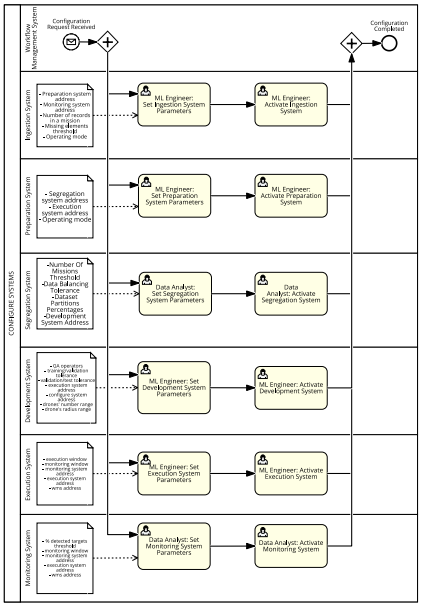
A mission is composed of three types of record:

- **Layout record** (aerial satellite photo)
- **Environmental record** (currents, coast, winds, coral reefs…)
- **Annotation record** (targets)

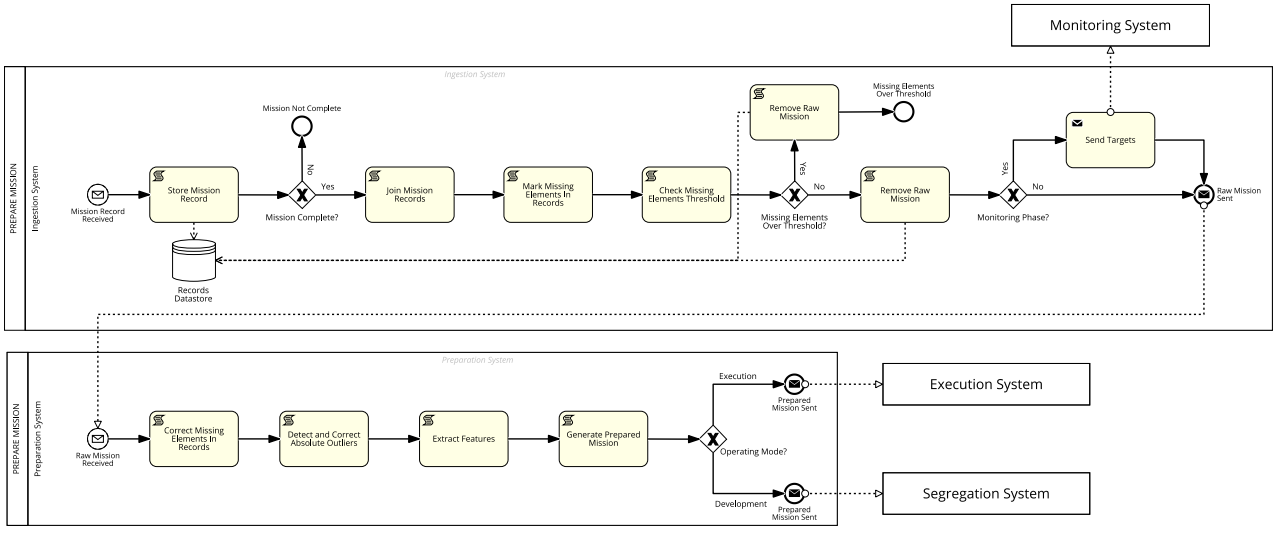
2 BPMNs

2.1 CONFIGURE COORDINATION LOGIC PARAMETERIZATION SERVICE (ALL)

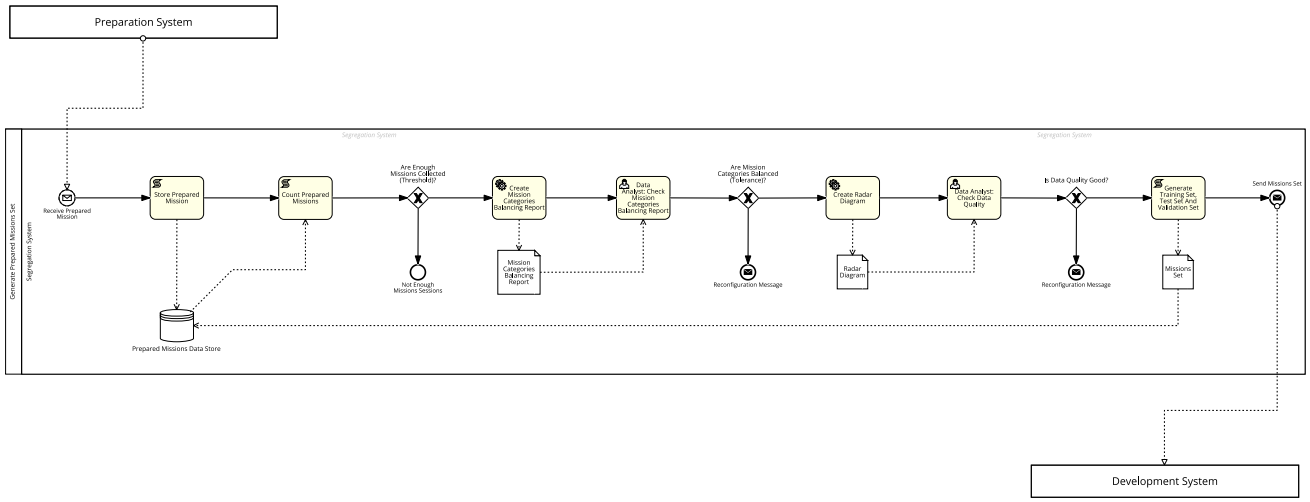




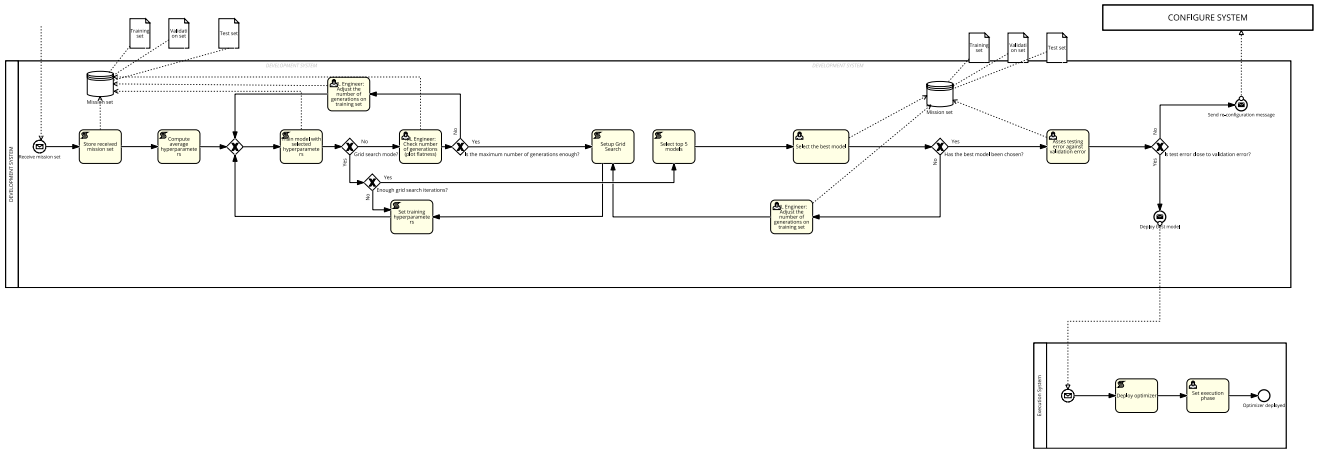
2.2 PREPARE MISSION (JACOPO)



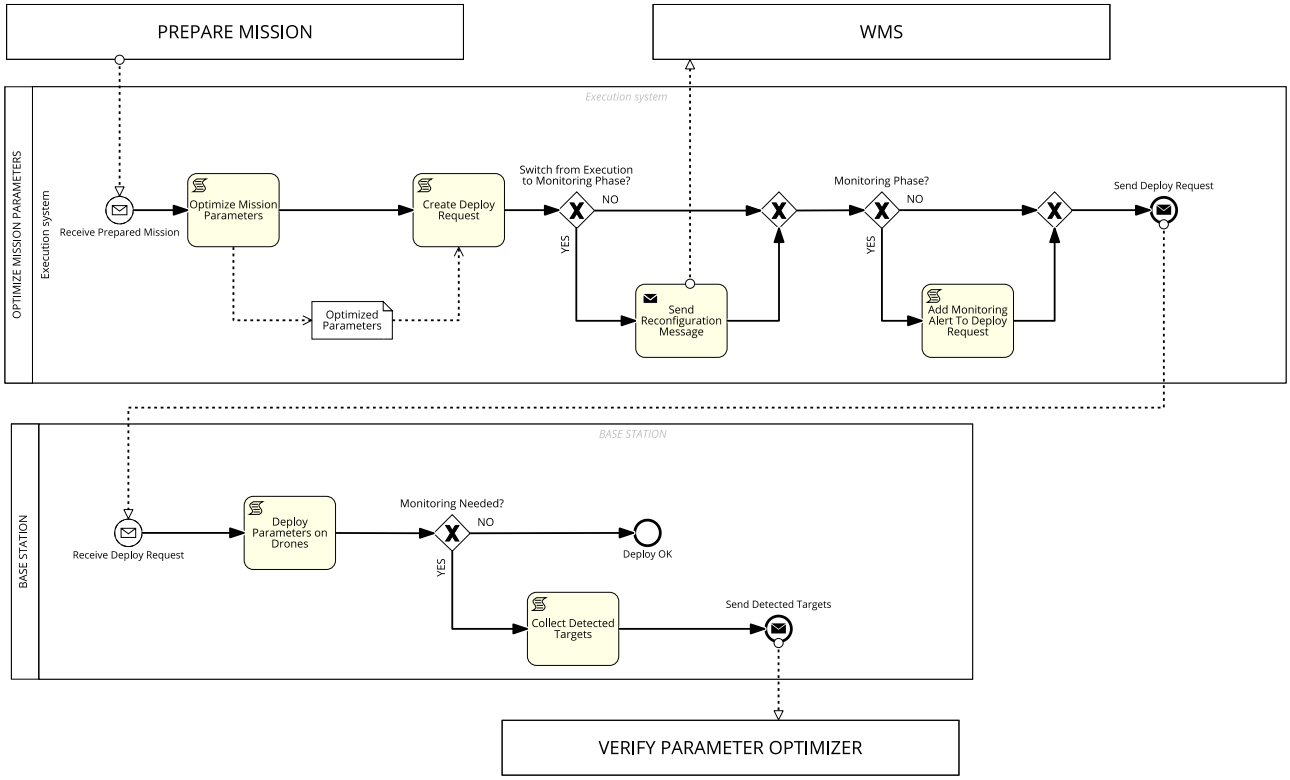
2.3 GENERATE PREPARED MISSIONS SET (FEDERICO)



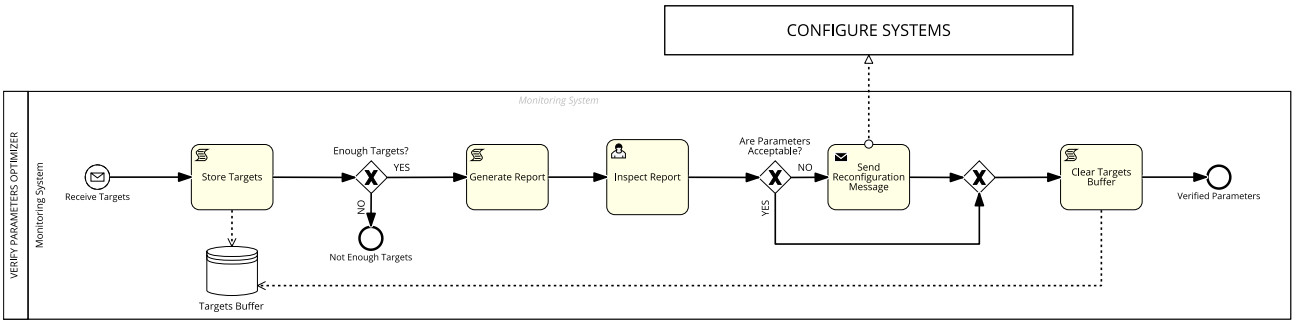
2.4 DEVELOP PARAMETERS OPTIMIZER (FRANCO)



2.5 OPTIMIZE MISSION OPTIMIZER (MATTEO)



2.6 VERIFY PARAMETERS OPTIMIZER (MATTEO)



2.7 CONFIGURE – PHASE KNOWLEDGE

System	Initial Phase	Destination Phase	Change
Ingestion System	Execution	Monitoring/Development	Ask the label again
	Monitoring/Development	Execution	No label anymore (annotation record), only 2 records instead of 3
	*	Monitoring	Send targets to the monitoring system
Preparation System	Development	Execution/Monitoring	Send mission to the execution system
	Execution/Monitoring	Development	Send mission to the segregation system
Execution System	Execution	Monitoring	Send an alert to the base station
	Monitoring	Execution	Inform that monitoring is not needed anymore
Configure System	*	Development	Set and launch Segregate and Develop
	*	Monitoring/Execution	Set and launch Deploy and Monitor

If * is present, the phase is considered regardless of the incoming phase.

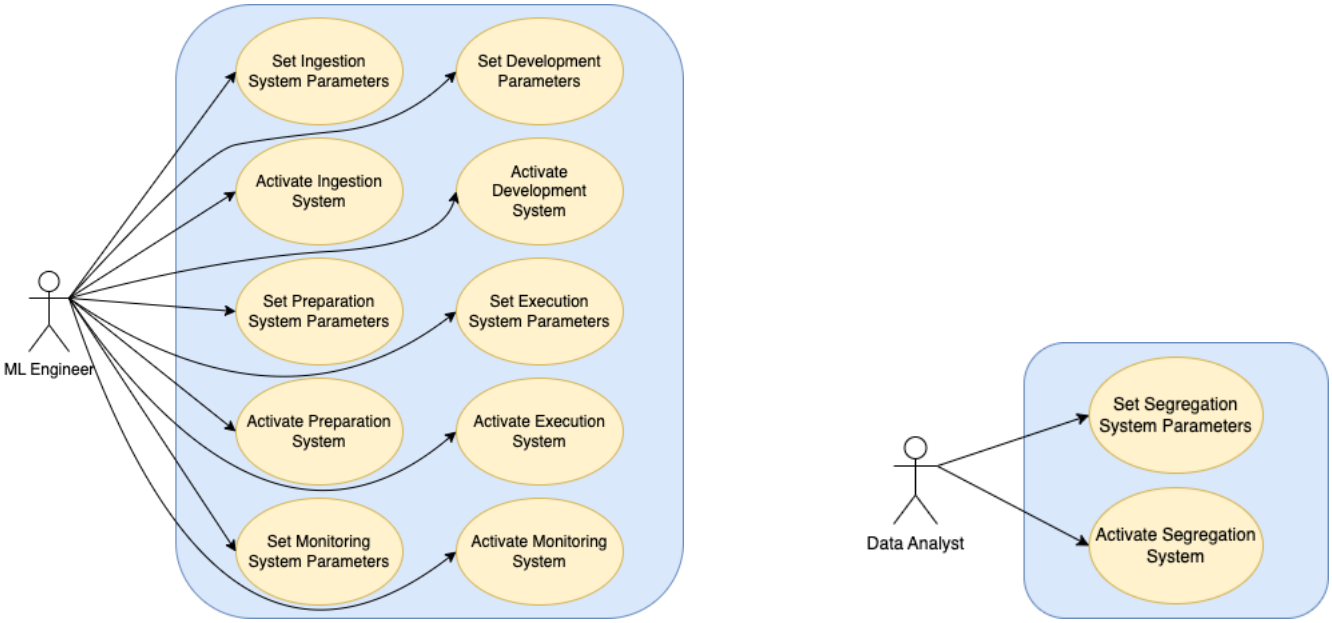
2.8 ASSUMPTIONS MADE

- Static plastic scenario, one static frame
- The sea current carries the plastic, approximately, in recurring positions

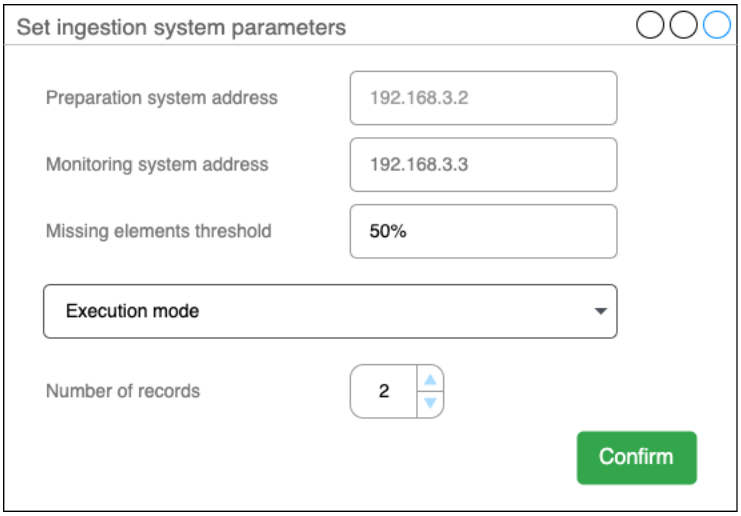
3 USE CASES

Job Profile	Source of information	Annual salary	Cost
Machine Learning Engineer	https://www.prospects.ac.uk/job-profiles/machine-learning-engineer#salary	£35,000	1.4
Data Analyst	https://www.prospects.ac.uk/job-profiles/data-analyst	£25,000	1

3.1. CONFIGURE COORDINATION LOGIC PARAMETERIZATION SERVICE (Tutti)

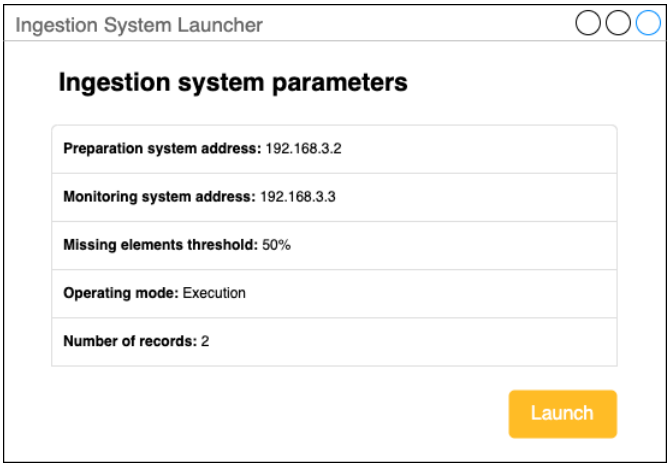


3.1.1 Set ingestion system parameters (Jacopo)



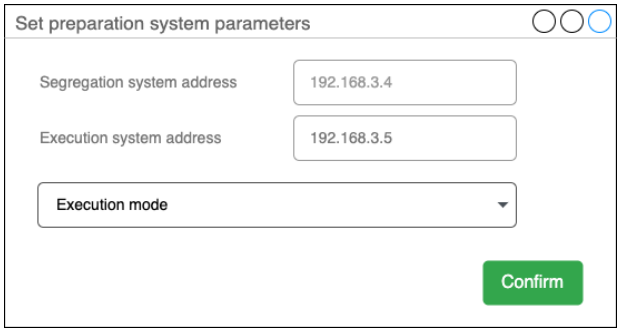
Subtask	Step	Actor	Cognitive effort	Occurrence	Cost
1	Open ingestion system parameters form	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
2	Show ingestion system parameters form	System			
3	Set preparation system address	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
4	Set monitoring system address	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
5	Set missing elements threshold	ML Engineer	Analyze (4)	1	1*4*1.4 = 5.6
6	Set operating mode	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
7 – IF (80%)	IF operating mode is Execution				
7.1	Set number of records in a mission equal to 2	ML Engineer	Remember (1)	0.3	1*0.8*1.4 = 1.12
8 – ELSE (20%)					
8.1	Set number of records in a mission equal to 3	ML Engineer	Remember (1)	0.7	1*0.2*1.4 = 0.28
				TOTAL COST	12.6

3.1.2 Activate ingestion system (Jacopo)



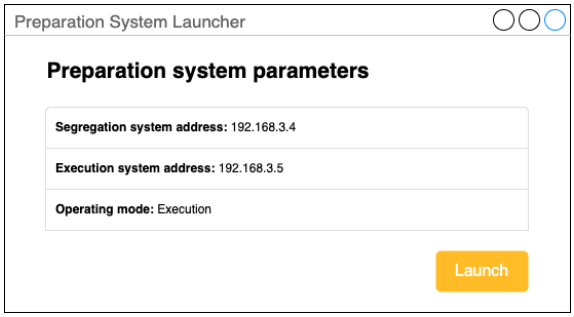
Subtask	Step	Actor	Cognitive effort	Occurrence	Cost
1	Open ingestion system launcher	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
2	Show ingestion system launcher	System			
3	Launch ingestion system	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
				TOTAL COST	2.8

3.1.3 Set preparation system parameters (Jacopo)



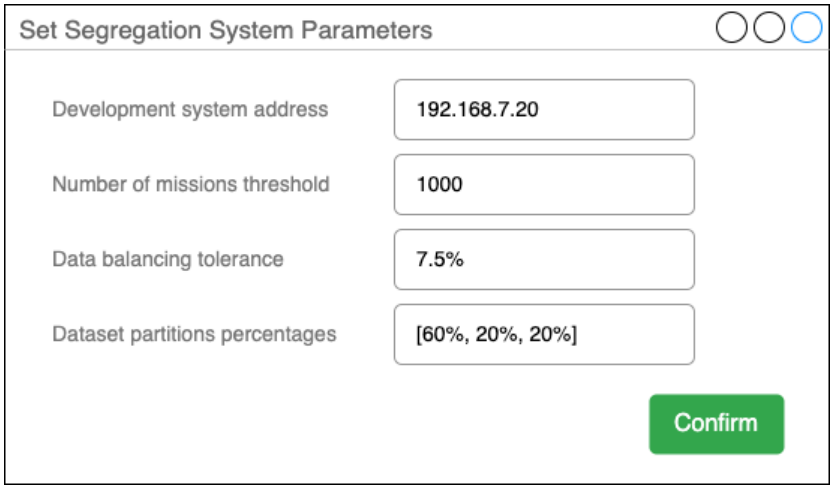
Subtask	Step	Actor	Cognitive effort	Occurrence	Cost
1	Open preparation system parameters form	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
2	Show preparation system parameters form	System			
3	Set segregation system address	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
4	Set execution system address	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
5	Set operating mode	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
				TOTAL COST	5.6

3.1.4 Activate preparation system (Jacopo)



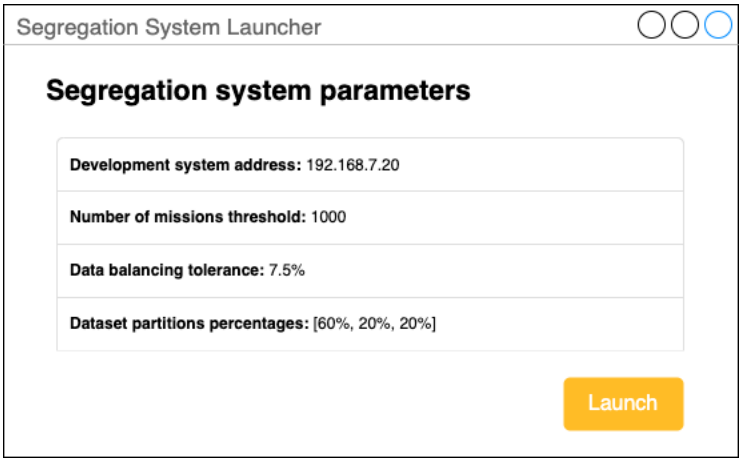
Subtask	Step	Actor	Cognitive effort	Occurrence	Cost
1	Open preparation system launcher	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
2	Show preparation system launcher	System			
3	Launch preparation system	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
				TOTAL COST	2.8

3.1.5 Set segregation system parameters (Federico)



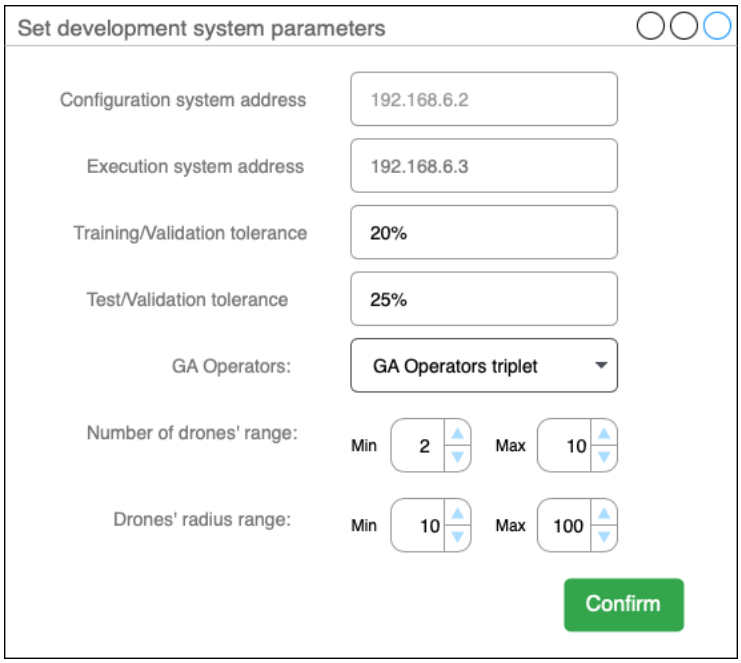
Subtask	Step	Actor	Cognitive effort	Occurrence	Cost
1	Click on 'Configure Segregation System Parameters'	Data Analyst	Remember (1)	1	1*1*1=1
2	Show panel to set Segregation System Parameters	System	--	--	--
3	Set the number of missions threshold by experience	Data Analyst	Analyze (4)	1	4*1*1=4
4	Set the data balancing tolerance by experience	Data Analyst	Analyze (4)	1	4*1*1=4
5	Set the dataset partitions percentages by experience	Data Analyst	Analyze (4)	1	4*1*1=4
6	Set the development system address	Data Analyst	Remember (1)	1	1*1*1=1
7	Click on 'Confirm' button	Data Analyst	Remember (1)	1	1*1*1=1
8	Save the configuration	System	--	--	--
				TOTAL COST	15

3.1.6 Activate segregation system (Federico)



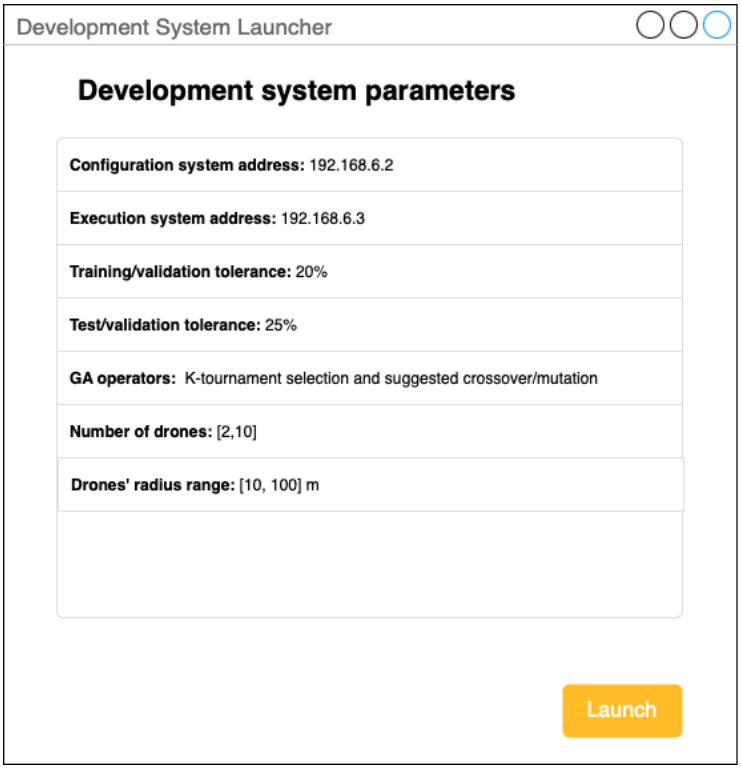
Subtask	Step	Actor	Cognitive effort	Occurrence	Cost
1	Open segregation system launcher	Data Analyst	Remember (1)	1	1*1*1 = 1
2	Show segregation system launcher	System			
3	Launch segregation system	Data Analyst	Remember (1)	1	1*1*1 = 1
				TOTAL COST	2

3.1.7 Set development system parameters (Franco)



Subtask	Step	Actor	Cognitive effort	Occurrence	Cost
1	Set configuration system address	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
2	Set execution system address	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
3	Set Training/Validation tolerance	ML Engineer	Apply (3)	1	3**11.4 = 4.2
4	Set Validation/Test tolerance	ML Engineer	Apply (3)	1	3*1*1.4 = 4.2
5	Set GA operators	ML Engineer	Understand (2)	2	1*2*1.4 = 2.8
6	Set number of drones' range	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
7	Set the drones' radius range	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
8	Click *Confirm*	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
				TOTAL COST	18.2

3.1.8 Activate development system (Franco)

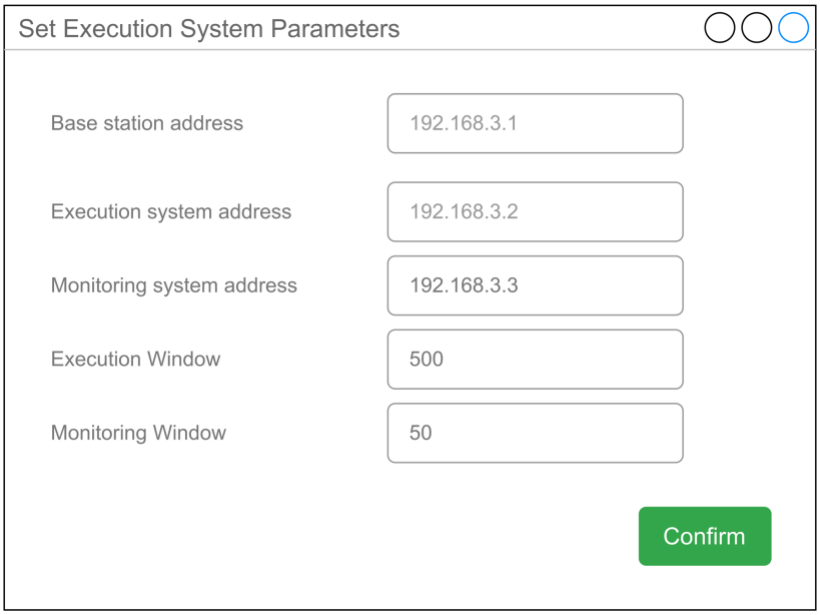


Subtask	Step	Actor	Cognitive effort	Occurrence	Cost
1	Open development system launcher	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
2	Show development system launcher	System			

3	Launch development system	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
				TOTAL COST	2.8

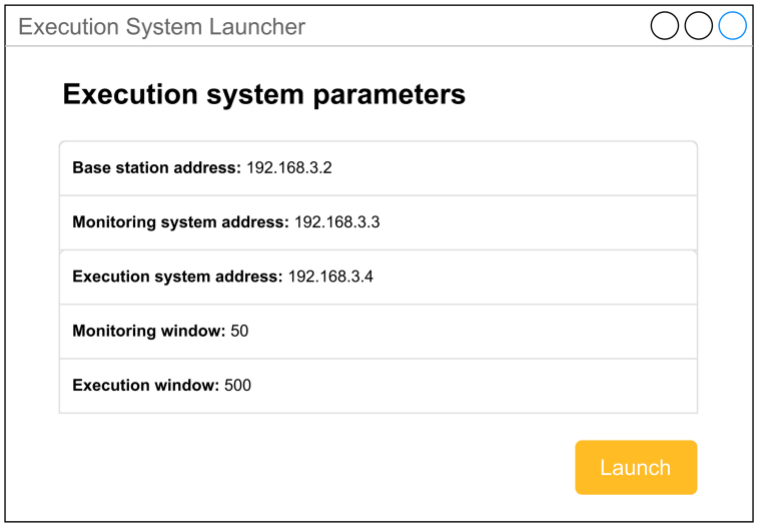
3.1.9 Set execution system parameters (Matteo)

The action of setting the parameters for the monitoring window and execution window are assumed to have a cognitive effort of 4 (analyze) as there is not a clear rule to derive them. The ML Engineer has the responsibility of choosing the appropriate value for these windows; the windows' values may be decreased when there is a need to have a stricter control over the optimizer's behavior or may be relaxed when the optimizer seems to work fine after some cycles of execution/monitoring.



Subtask	**Step**	**Actor**	**Cognitive effort**	**Occurrence**	**Cost**
1	Set Base station address	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
2	Set execution system address	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
3	Set Monitoring system address	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
4	Set Execution Window	ML Engineer	Analyze (4)	1	4*1*1.4 = 5.6
5	Set Monitoring window	ML Engineer	Analyze (4)	1	4*1*1.4 = 5.6
6	Click *Confirm*	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
				TOTAL COST	16.8

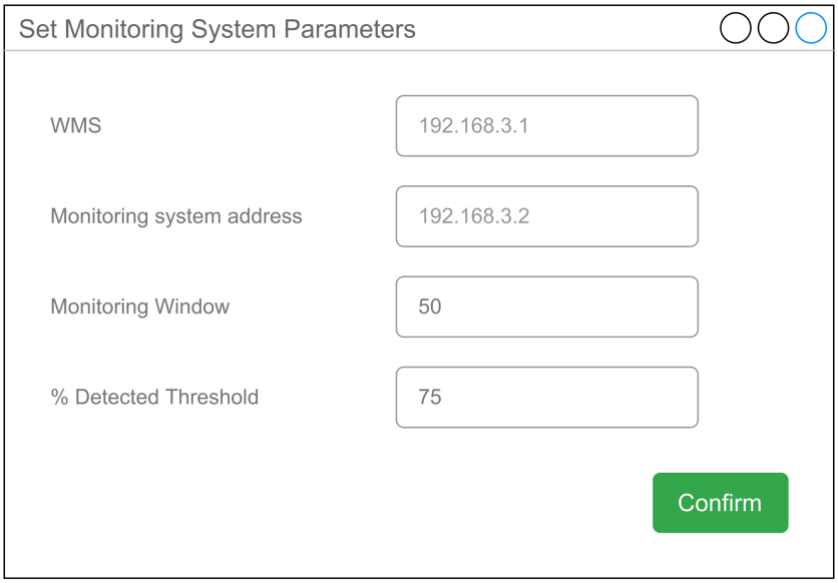
3.1.10 Activate execution system (Matteo)



Subtask	Step	Actor	Cognitive effort	Occurrence	Cost
1	Open execution system launcher	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
2	Show execution system launcher	System			
3	Launch execution system	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
				TOTAL COST	2.8

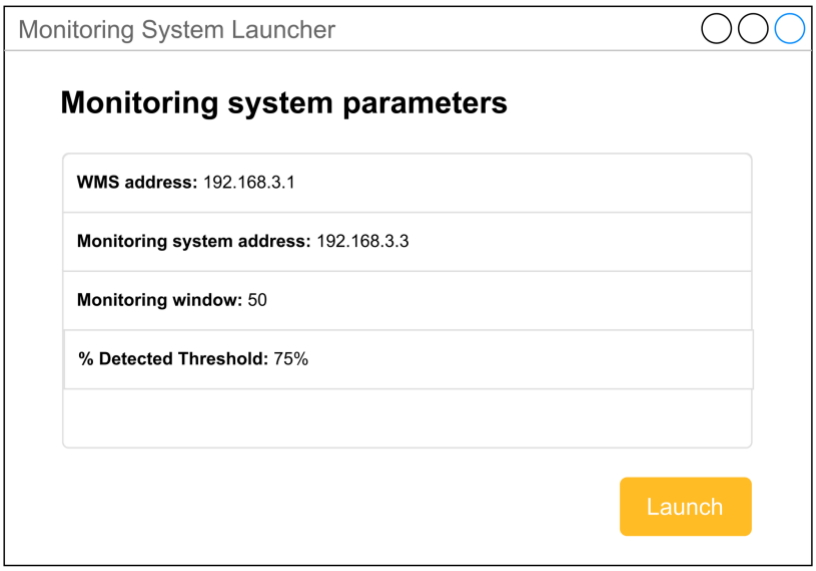
3.1.11 Set monitoring system parameters (Matteo)

The action of setting the monitoring window in the monitoring system is, unlike in the execution system, assumed to have a cognitive effort of (1). In fact, the responsibility of setting this parameter is of the ML Engineer and he/she has already made this decision during the setup of the execution system, so he/she just needs to remember the value to insert into the client.



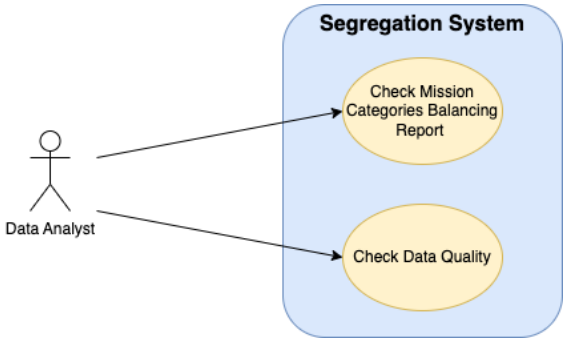
Subtask	Step	Actor	Cognitive effort	Occurrence	Cost
1	Set WMS address	ML Engineer	Remember (1)	1	1*1.4*1 = 1.4
2	Set monitoring system address	ML Engineer	Remember (1)	1	1*1.4*1 = 1.4
3	Set Monitoring window	ML Engineer	Remember (1)	1	1*1.4*1 = 1.4
4	Set % Detected threshold	ML Engineer	Analyze (4)	1	4*1.4*1 = 5.6
5	Click *Confirm*	ML Engineer	Remember (1)	1	1*1.4*1 = 1.4
				TOTAL COST	11.2

3.1.12 Activate monitoring system (Matteo)

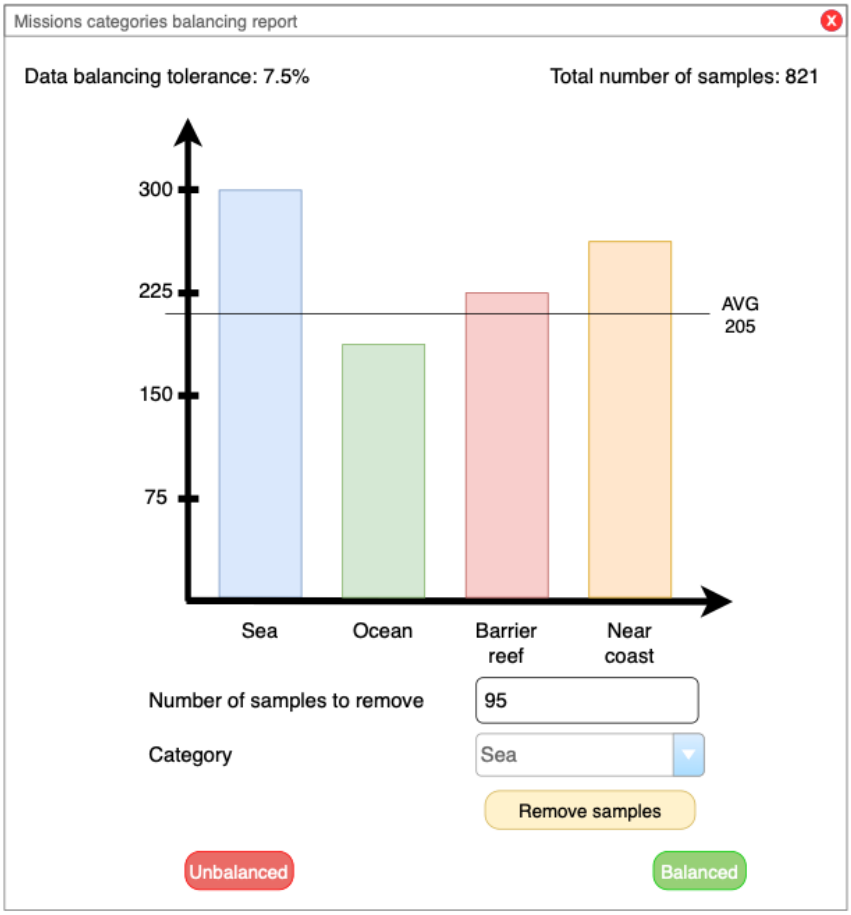


Subtask	Step	Actor	Cognitive effort	Occurrence	Cost
1	Open monitoring system launcher	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
2	Show monitoring system launcher	System			
3	Launch monitoring system	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
				TOTAL COST	2.8

3.2 GENERATE PREPARED MISSIONS SET



3.2.1 Check mission categories balancing report (Federico)



Subtask	Step	Actor	Cognitive Effort	Occurrence	Total cost
1	Open the missions categories balancing report	Data Analyst	Remember (1)	1	1*1*1 = 1

2	Show the missions categories balancing report	System	--	--	--
3	Calculate the tolerance of the average number of samples	Data Analyst	Apply (3)	1	3*1*1 = 3
4	**FOR** each categories	--	--	--	--
4.1	Compare (difference) the actual number of samples with the average number of samples	Data Analyst	Apply (3)		3*1*1 = 3
4.2 IF (60%)	**IF** the difference is over the tolerance of the average number samples	--	--	--	--
4.2.1	Set the number of samples to remove (difference)	Data Analyst	Remember (1)	0.6	1*0.6*1 = 0.6
4.2.2	Find and select the considered category	Data Analyst	Remember (1)	0.6	1*0.6*1 = 0.6
4.2.3	Click the 'Remove samples' button	Data Analyst	Remember (1)	0.6	1*0.6*1 = 0.6
4.2.3	Remove samples of the selected category and update the view	System	--	--	--
4.3 ELSEIF (20%)	**ELSEIF** the difference is under the tolerance of the average number of samples	--	--	--	--
4.3.1	Click the 'Unbalanced' button	Data Analyst	Remember (1)	0.2	1*0.2*1 = 0.2
4.3.2	->6	--	--	--	--
5	Click the 'Balanced' button	Data Analyst	Remember (1)	1	1*1*1=1
6	Close the panel	System	--	--	--
				TOTAL COST	4+4*(3+1.8+0.2)+1 = 25

3.2.2 Check data quality (Federico)

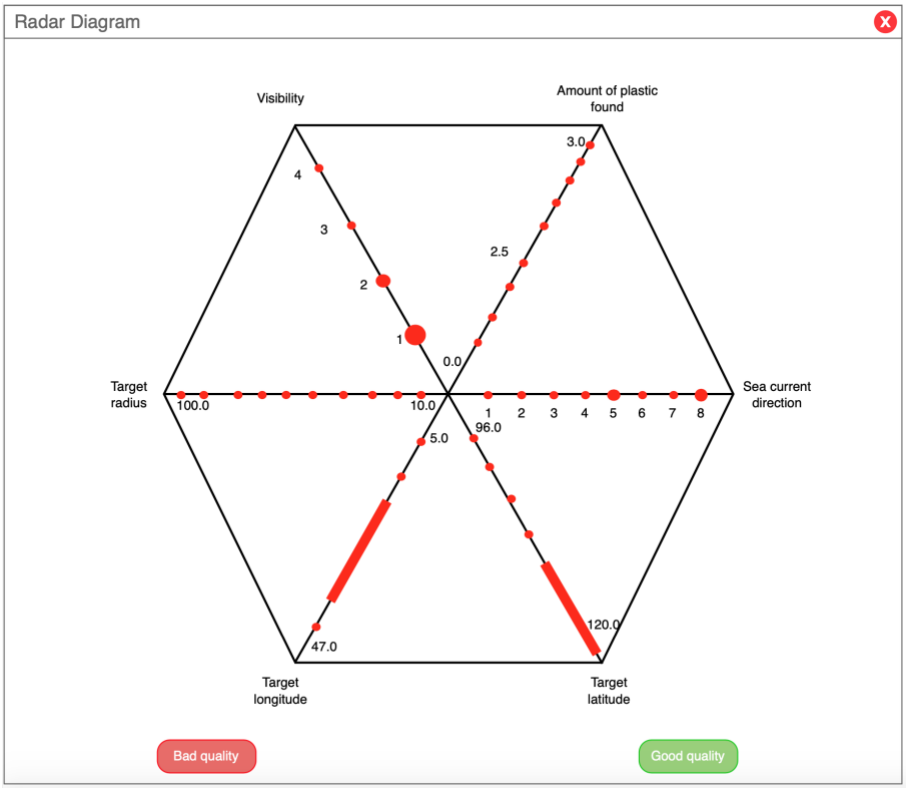
Sea current direction [N(1), S(2), E(3), O(4), N-E(5), N-O(6), S-E(7), S-O(8)]

Category[sea(1), ocean(2), barrier reef(3), near coast(4)]

Visibility[high(1), medium(2), medium-low(3), low(4)]

The other dimensions are normalized with the min-max normalization.

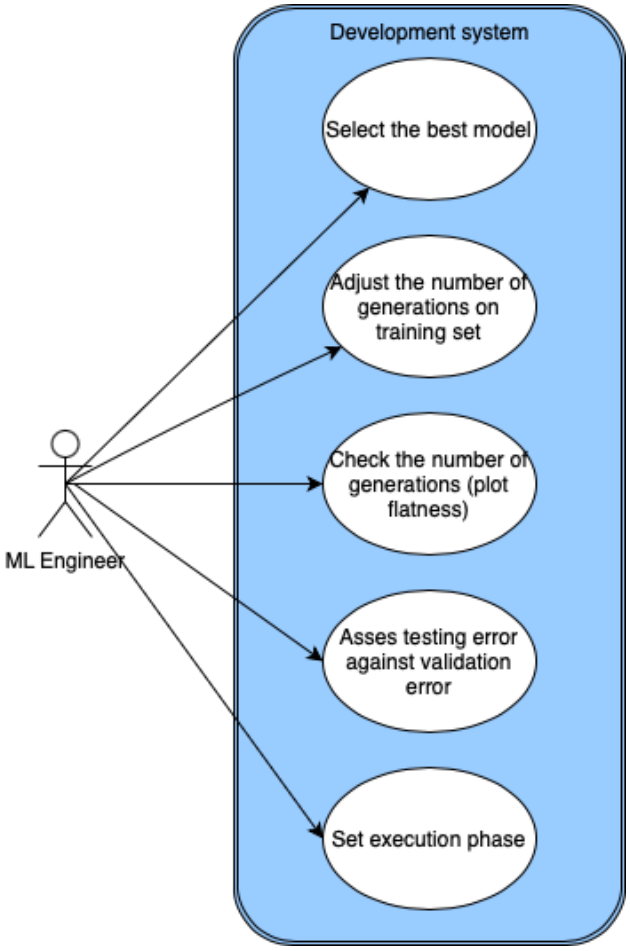
Latitude and longitude are to be understood as global, for the optimizer it could be useful to know them globally in combination with information relating to the sea current.



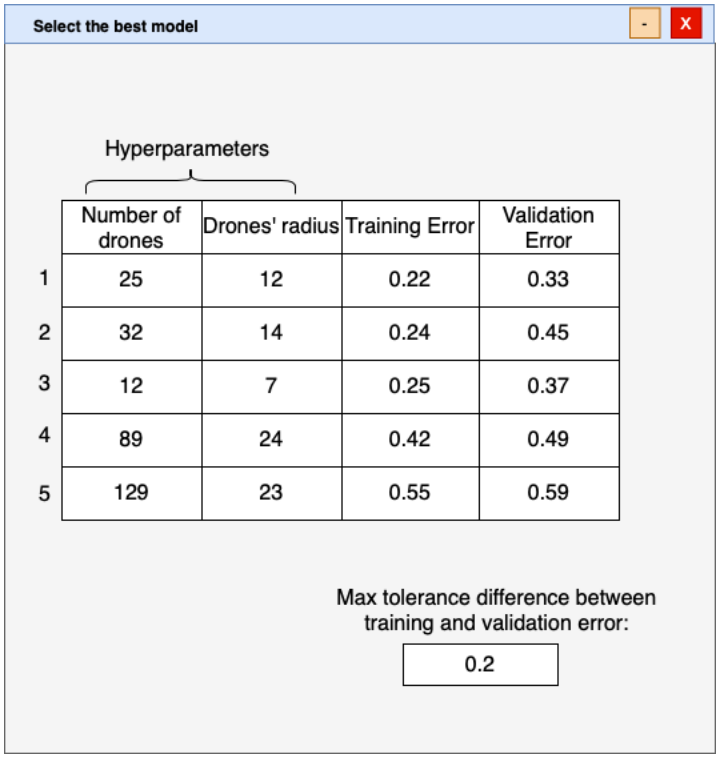
Subtask	Step	Actor	Cognitive Effort	Occurrence	Total cost
1	Open the radar diagram	Data Analyst	Remember (1)	1	1*1*1 = 1
2	Show radar diagram panel	System	--	--	--
3	**FOR** each feature	--	--	--	--
3.1	Take a feature and examine the feature distribution	Data Analyst	Analyze (4)	1	1*4*1 = 4
3.2 – IF (20%)	**IF** the distribution is not uniform as expected	--	--	--	--
3.2.1	Click on 'Bad quality' button	Data Analyst	Remember (1)	0.2	1*0.2*1 = 0.2
3.2.2	Set to false the 'good quality' property	System	--	--	--
3.2.3	-> 7	--	--	--	--
4	Click on 'Good quality button	Data Analyst	Remember (1)	1	1*1*1 = 1

5	Set to true the 'good quality' property	System	--	--	--
7	Close the radar diagram panel	System	--	--	--
				Total cost:	1+6*(4+0.2)+ 1 = 27.2

3.3 DEVELOPMENT SYSTEM



3.3.1 Select the best model (Franco)



Subtask	Actor	Step	Cognitive Effort	Occurrence	Total cost
1	ML Engineer	Open the top 5 models performances' metrics	Remember (1)	1	1*1*1.4 = 1.4
2	System	Shows models performances' metrics			
3	ML Engineer	Select validation error column	Remember (1)	1	1*1*1.4 = 1.4
4	System	Sort by validation error (ASC)			

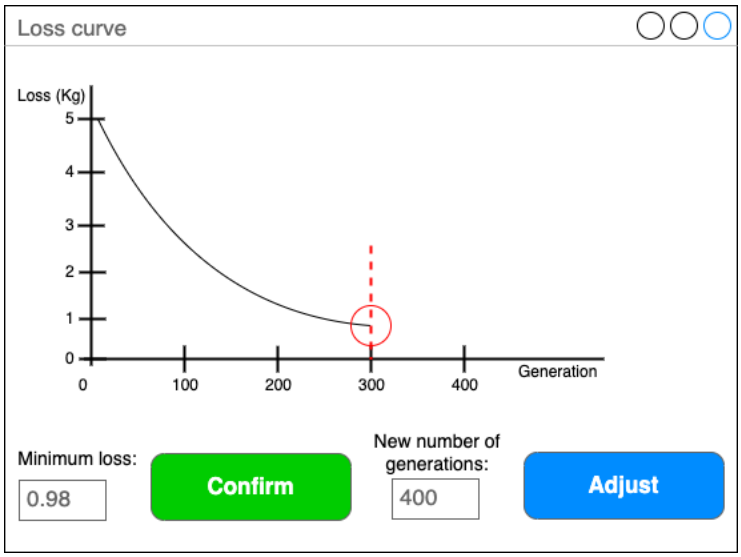
5	ML Engineer	**FOR** each model, find the first model with difference among errors below tolerance	Apply (3)	1	3*1*1.4 = 4.2
6	ML Engineer	**FOR** each next model satisfying the tolerance condition in order, check if the validation error is similar but the number of drones or the drones' radius is lower	Apply (3)	1	3*1*1.4 = 4.2
6.1	System	**IF** the condition is satisfied			
6.1.1 (IF 30%)	ML Engineer	Selects it as best model	Remember (1)	0.4	1*0.3*1.4 = 0.42
6.2	System	**ELSE**			
6.2.1. ELSE 70%)	ML Engineer	Maintain the previous model	Remember (1)	0.6	1*0.7*1.4 = 0.98
				Total cost:	12,6

Percentage choice: 30/70% is an optimistic hypothesis, if we have a lot of mission data, I will frequently have models with similar performances.

3.3.2 Adjust the number of generations on the training set (Jacopo)

Working hypothesis

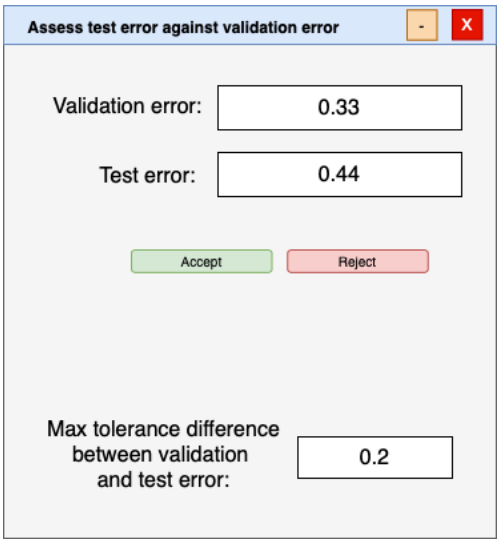
- The loss curve is non-increasing
- The loss is measured in Kilograms of plastic found



Subtask	Step	Actor	Cognitive Effort	Occurrence	Total cost
1	Open the loss curve of the current model	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
2	Show the loss curve of the current model	System			
3	Evaluate for how many generations the loss is flat	ML Engineer	Apply (3)	1	3*1*1.4 = 4.2
4 – IF (40%)	IF the loss is flat for at least half of the generations	ML Engineer			
4.1	Reduce by one third the number of generations to manage overfitting	ML Engineer	Apply (3)	0.4	3*0.4*1.4 = 1.26
5 – IF (40%)	IF the loss is not flat at the end of the generations	ML Engineer			
5.1	Enlarge by one third the number of generations	ML Engineer	Apply (3)	0.4	3*0.4*1.4 = 1.68
7 ELSE (20%)	ELSE	ML Engineer			
7.1	Confirm the number of generations		Remember (1)	0.2	1*0.2*1.4 = 0.28

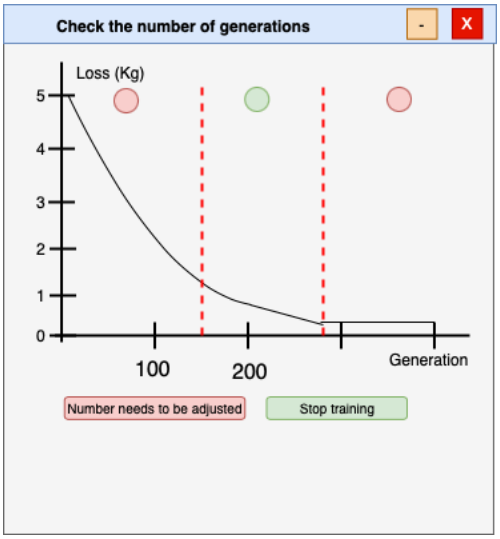
				Total cost:	8,82

3.3.3 Assess testing error against validation error (Franco)



Subtask	Step	Actor	Cognitive Effort	Occurrence	Total cost
1	Open the model performance	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
2	Show the model performance	System			
3	**IF** the difference between test error and validation error is below tolerance	ML Engineer	Apply (3)	1	3*1*1.4 = 4.2
4 (IF 50%)	Accept model	ML Engineer	Remember (1)	1	1*0.5*1.4 = 0.7
5	**ELSE**				
6 (ELSE 50%)	Reject model	ML Engineer	Remember (1)	1	1*0.5*1.4 = 0.7
				Total cost:	7

3.3.4 Check the number of generations (Franco)



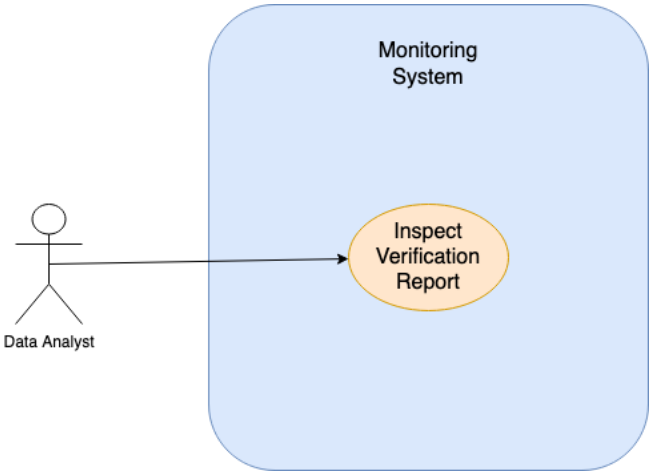
Subtask	Step	Actor	Cognitive Effort	Occurrence	Total cost
1	Open the model training plot	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
2	Shows the model training plot	System			
3	**IF** the number of generations is not flat or not descending anymore	ML Engineer		1	
4 (IF 50%)	Stop training	ML Engineer	Apply (3)	1	3*0.5*1.4 = 2.1
5	**ELSE** (still flat or descending)				
6 (ELSE 50%)	Continue training (the number of generations must be adjusted)	ML Engineer	Apply (3)	1	3*0.5*1.4 = 2.1
				Total cost:	5.6

3.3.5 Set execution phase (Franco)

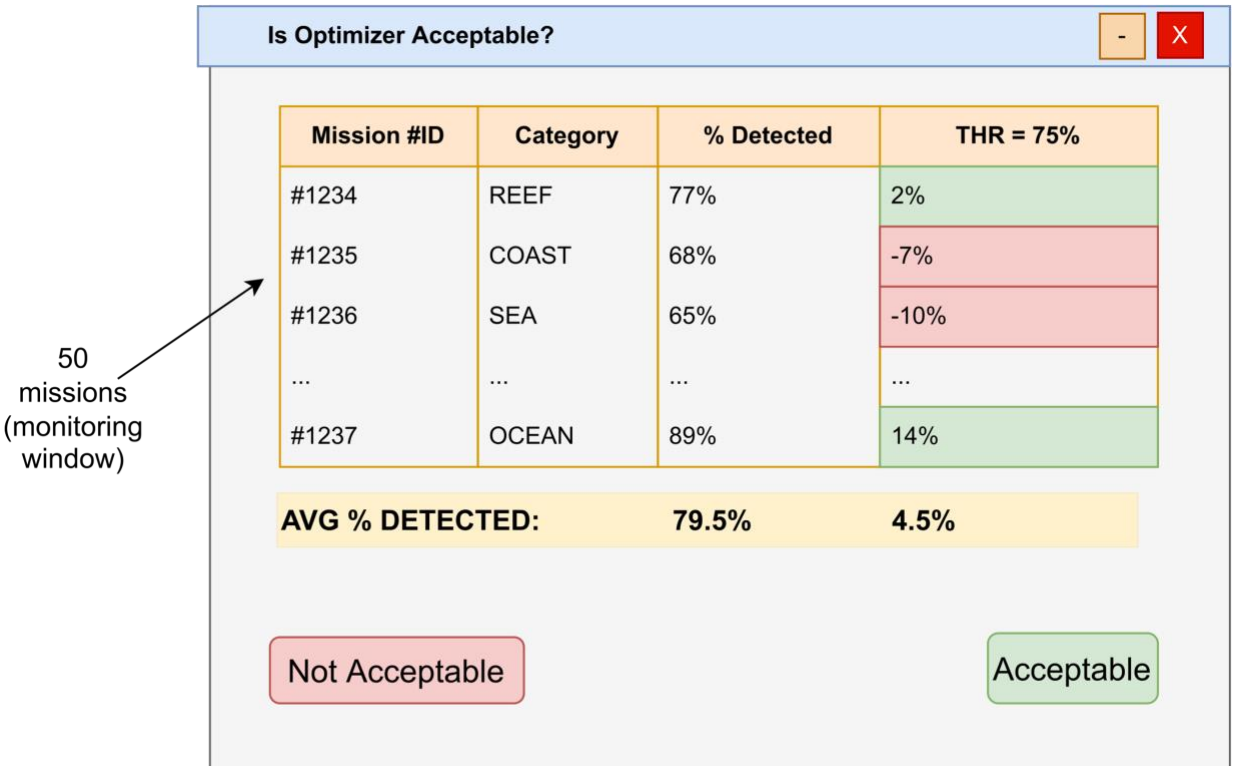
Subtask	Step	Actor	Cognitive Effort	Occurrence	Total cost
1	Select current phase	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
2	Shows current phase	System			
3	Set execution phase	ML Engineer	Remember (1)	1	1*1*1.4 = 1.4
				Total cost:	2.8

3.4 VERIFY OPTIMIZER

3.4.1 Inspect Verification Report (Matteo)

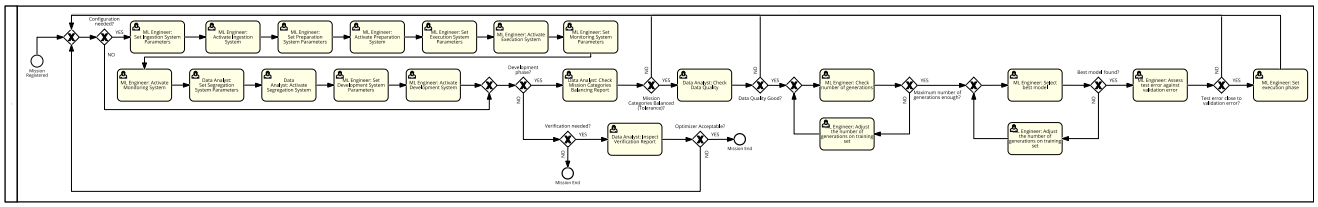


Notice that the probabilities of entering the IF/ELSE conditions depend on various hypotheses. For example, if the company takes the training data from a global consortium of firms specialized in detecting marine plastic then the probability of entering the ELSE condition (hence the classifier needs to be retrained) should be low, around 15-20%. Still, even if we have a lot of good quality data, the optimizer might be retrained due to periodical changes of the environmental conditions (sea currents, marina plastic visual characteristics, …).



Subtask	Step	Actor	Cognitive Effort	Occurrence	Total cost
1.	Shows a table with: Mission ID, Mission Type, % of Detected Targets	System	///	1	///
2.	Compare average % of detected targets with threshold	Data Analyst	Apply (3)	1	3*1*1
3. **IF** (80%)	**IF** average detected targets are above threshold	///	///	0.8	///
3.1	Clicks on *Acceptable*	Data Analyst	Remember (1)	0.8	1*0.8*1
4. **ELSE** (20%)	///	///	///	0.2	///
4.1	Clicks on *Not Acceptable*	Data Analyst	Remember (1)	0.2	1*0.2*1
				Total cost:	4

4 AS-IS



4.1 DOMAIN APPLICATION ASSUMPTIONS

- To simulate the factory in the long term, we assume to have to train 5 optimizers, and each one is going to be trained 3 times
- Development phase?
  Once the 5 optimizers have been trained, each one will be online for at least 500 executions. Since the probability of rejecting an optimizer during the monitoring phase is 20%, we can assume that after 3 cycles of execution/monitoring we must retrain the optimizer, leading to a probability under 1% of being in development. Considering that we have many loops inside the development phase and that each optimizer will be trained 3 times, we set the probability of the gateway as around 3*4 loopbacks * 5 optimizers / 1500 = around 5%:
    - YES – 5%
    - NO – 95%
- Configuration needed?
  We have 5 optimizers and each one will be trained and deployed 3 times, so a total of 5*3*2 phase switching between development and execution. We assume that the retraining happens after 3 execution windows, so we have 5*6 / 1500 = around 2%
    - YES – 2%
    - NO – 98%
- Mission categories balanced?
  We assume that once in 5 times there is a class that has few samples (according to the tolerance percentage chosen by configuration); so, in this case we need to request new data before proceeding to split the dataset. Obviously, this percentage is affected by the number of samples we collect/are provided with (the higher this number, the lower the probability that the classes will be balanced with respect to the total number of samples, but we will have higher costs) and by the tolerance percentage chosen (which can be larger or smaller depending on the requirements)
    - YES – 80%
    - NO – 20%
- Data quality good?
  We assume that once in 5 times at least one of the features is not balanced as we expected. This percentage is greatly influenced by the company expectation with respect to the specific domain in which the company operates (e.g., sea and ocean) and by the number of samples we have.
    - YES – 80%
    - NO – 20%
- Maximum number of generations enough?
  We assume that we require in average 4 adjustments to the number of generations before concluding training. This percentage is though greatly affected by considerations such as the architecture of the model and the amount of data we have available.

    - YES – 25%
    - NO – 75%
- Best model found?
  We assume that 10% of the time, none of the top 5 models can respect the tolerance between validation and training error. This tolerance may though decrease drastically the more the data available.
    - YES – 70%
    - NO – 30%
- Test error close to validation error?
  We assume that 20% of the time, the best model cannot respect the tolerance between validation and test error.
    - YES – 80%
    - NO – 20%
- Verification needed?
  We assume to have an execution window of 500 and a monitoring window of 50. Since the "Verify Report" human task is executed only one time every 550 executions, the probability would be 1/550. Since we assume to have 5 classifiers, we can round to 1%.
    - YES – 1%
    - NO – 99%
- Optimizer acceptable?
  We take our data from a global consortium so it's safe to assume that the 80% of the times there will be no need to retrain the optimizer as we have plenty of available data of good quality.
    - YES – 80%
    - NO – 20%

4.1.1 Number of initial tokens

We assume to have 100 prepared missions to set to train an optimizer, and since we have 4 types of missions, we are going to have 25 missions per type. We are going to have re-training of these optimizer, and we assume to re-train on average after 500 missions. During monitoring we consider 50 cases to monitor. To simulate the fabric in the long term, we train 5 optimizers, and so we are going to have 5 training, each that is going to be trained 3 times

Number of tokens = 5 parameter optimizers*(500 missions+50 monitoring)*3 training = 8250 ~= 10000

In this way, we can verify the fabric in the long term.

4.2 AS-IS SIMULATION

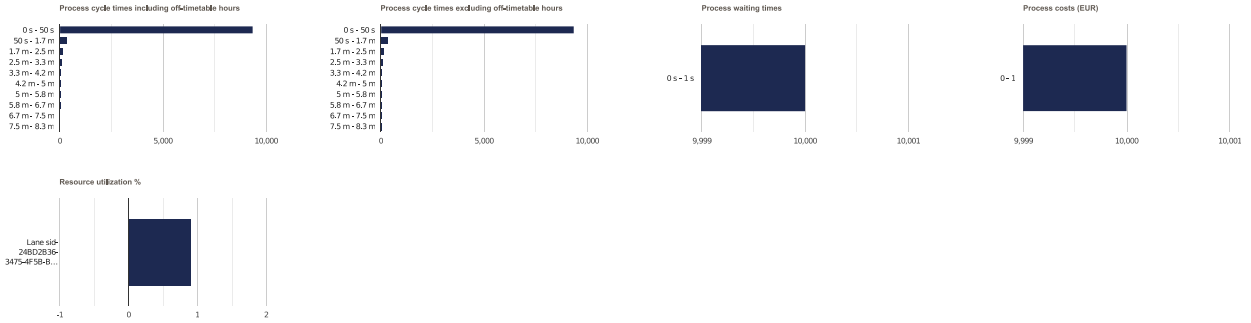
General information

Completed process instances 10000

Total cost 0 EUR

Total simulation time 14.9 weeks

Charts

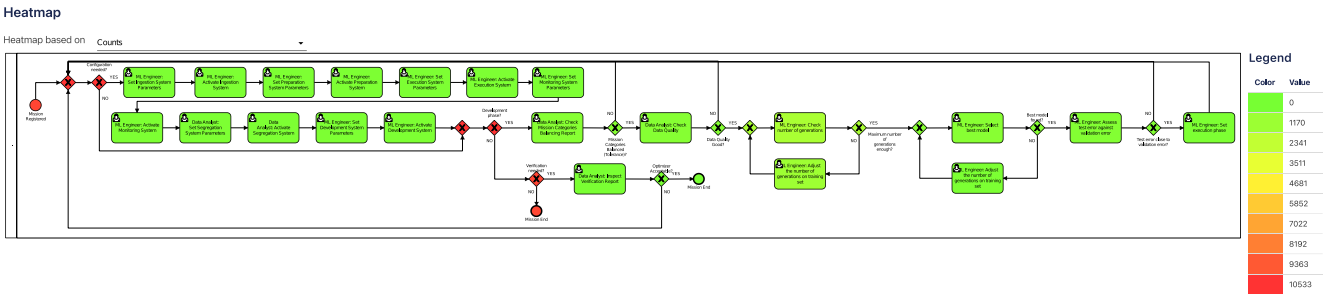


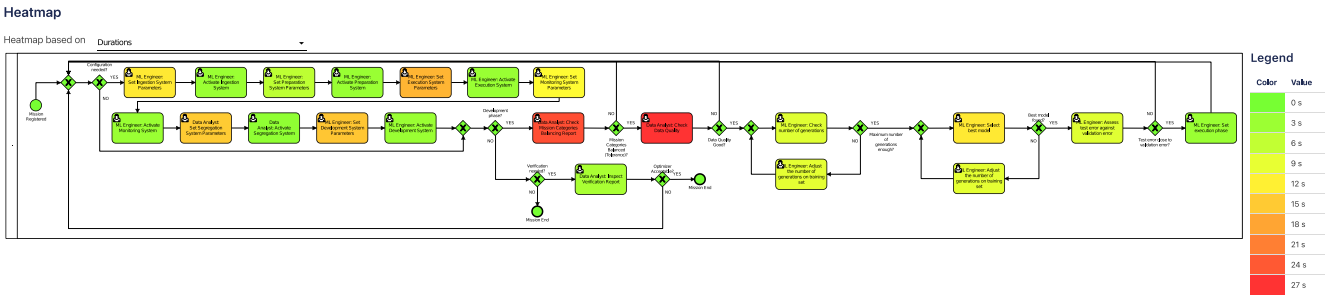
Scenario Statistics

	Minimum	Maximum	Average
Process instance cycle times including off-timetable hours	0 seconds	8.3 minutes	8.1 seconds
Process instance cycle times excluding off-timetable hours	0 seconds	8.3 minutes	8.1 seconds
Process instance costs	0 EUR	0 EUR	0 EUR

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
Data
Analyst: Activate Segregation System	197	0 s	0 s	0 s	1.8 s	2 s	2.2 s	0 s	0 s	0 s	0	0	0	0	0	0
Data Analyst:
Set Segregation System Parameters	197	0 s	0 s	0 s	13.5 s	15 s	16.5 s	0 s	0 s	0 s	0	0	0	0	0	0
Data Analyst: Check Data Quality	479	0 s	0 s	0 s	24.5 s	27.1 s	29.9 s	0 s	0 s	0 s	0	0	0	0	0	0
Data Analyst: Check Mission Categories Balancing Report	508	0 s	0 s	0 s	22.5 s	25 s	27.5 s	0 s	0 s	0 s	0	0	0	0	0	0
Data Analyst: Inspect Verification Report	103	0 s	0 s	0 s	3.6 s	4 s	4.4 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer:
Activate Ingestion System	197	0 s	0 s	0 s	2.5 s	2.8 s	3.1 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer:
Activate Preparation System	197	0 s	0 s	0 s	2.5 s	2.8 s	3.1 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer:
Set Ingestion System
Parameters	197	0 s	0 s	0 s	11.3 s	12.7 s	13.9 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer:
Set Preparation System Parameters	197	0 s	0 s	0 s	5 s	5.6 s	6.2 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Activate Development System	197	0 s	0 s	0 s	2.5 s	2.8 s	3.1 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Activate Execution System	197	0 s	0 s	0 s	2.5 s	2.8 s	3.1 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Activate Monitoring System	197	0 s	0 s	0 s	2.5 s	2.8 s	3.1 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Adjust the number of generations on training set	172	0 s	0 s	0 s	7.9 s	8.8 s	9.7 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Adjust the number of generations on training set	1213	0 s	0 s	0 s	7.9 s	8.8 s	9.7 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Assess test error against validation error	377	0 s	0 s	0 s	7.6 s	8.4 s	9.2 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Check number of generations	1590	0 s	0 s	0 s	7.6 s	8.4 s	9.2 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Select best model	549	0 s	0 s	0 s	11.3 s	12.6 s	13.9 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Set Development System Parameters	197	0 s	0 s	0 s	13.9 s	15.5 s	16.9 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Set Execution System Parameters	197	0 s	0 s	0 s	15.2 s	16.8 s	18.4 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Set Monitoring System Parameters	197	0 s	0 s	0 s	10.1 s	11.2 s	12.3 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Set execution phase	203	0 s	0 s	0 s	2.5 s	2.8 s	3.1 s	0 s	0 s	0 s	0	0	0	0	0	0





5 TO-BE

5.1 MODIFICATION AT HANDOFF LEVEL (Federico)

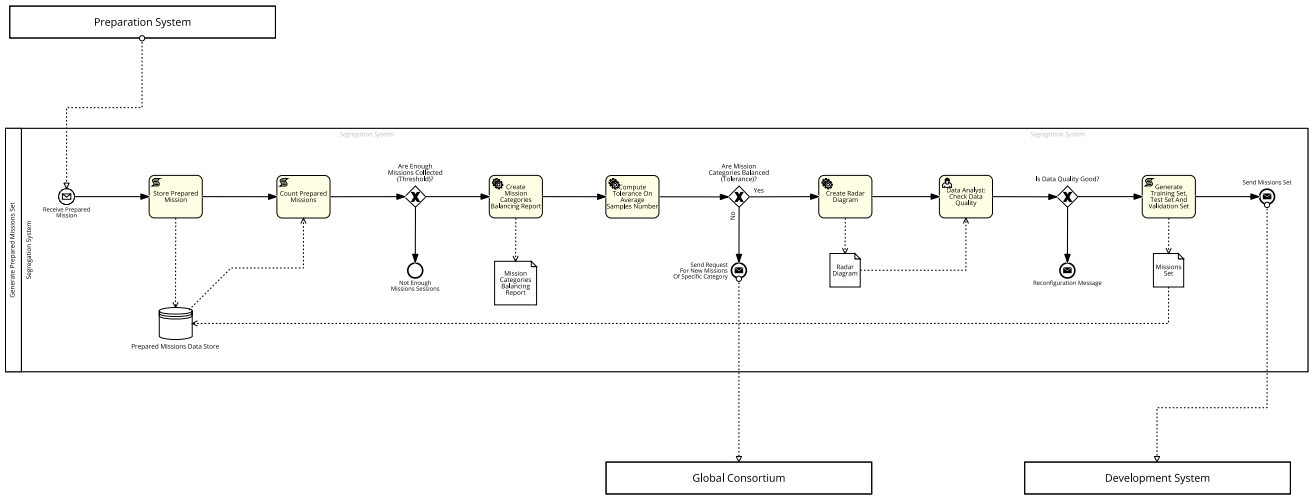
We perform a modification at "handoff level" by going to remove the "user task" of "check mission categories balancing report ". When we need to balance the data, instead of going to collect new data, we get the data from a global consortium that provides us a number of missions in the category we need to balance.

With that update we can consider the probability associated to the gateway “Mission categories balanced?”:

- YES – 95%
- NO – 5%

Because only the first time we can have unbalanced data.

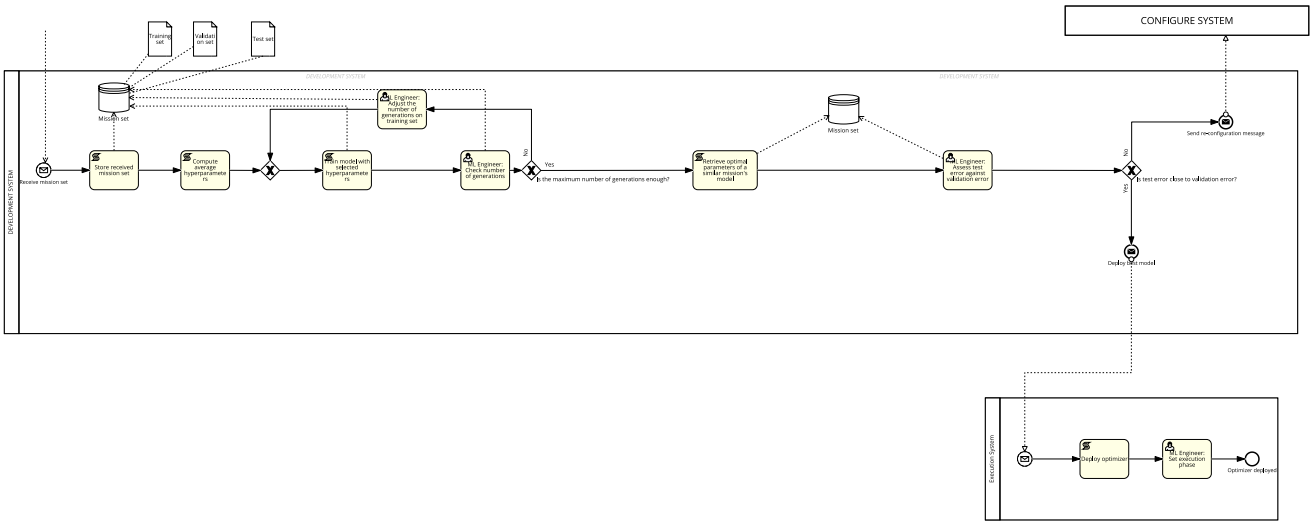
Simplified BPMN model:



5.2 MODIFICATION AT SERVICE LEVEL (Franco)

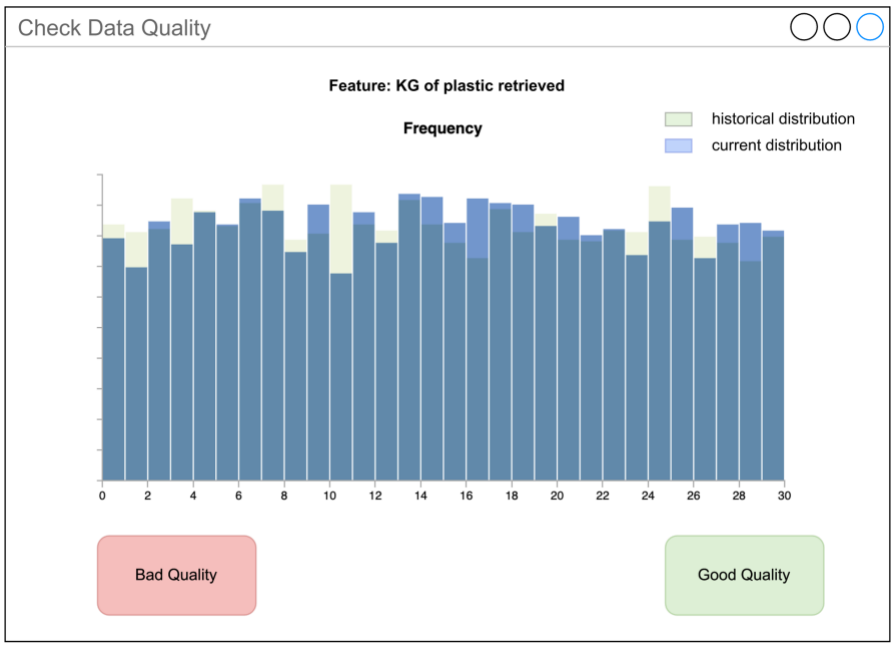
We applied a modification at the service level to the development system, avoiding the grid search and the inner steps involved, but using the optimal parameters of a similar mission. We take a similar case and avoid the training loop and the cost needed to find the best model among the top 5 parameterizers.

Simplified BPMN model:



5.3 MODIFICATION AT TASK LEVEL (Matteo) (OUTDATED)

Since the task *Check data quality* is by far the most expensive in our AS-IS model simulation, we decided to lower its cognitive effort from a maximum of 4 (analyze) to a maximum of 2 (understand). The system now provides for each feature of the radar diagram an additional graph to help the Data Analyst: the current distribution of the feature is superimposed with the historical “good quality” distributions of the same feature. The Data Analyst just has to compare the current distribution with the superimposed one to see if they are similar.



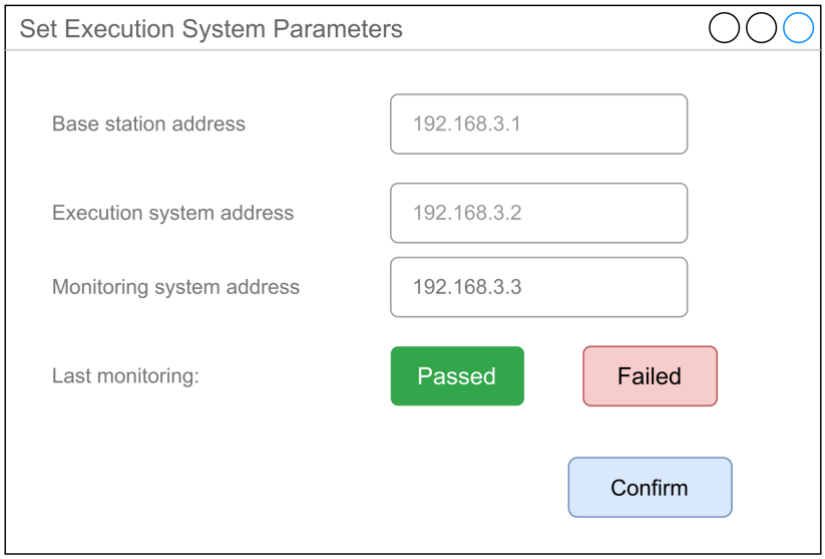
Subtask	Step	Actor	Cognitive Effort	Occurrence	Total cost
1	Launch the *Check Data Quality* helper	Data Analyst	Remember (1)	1	1*1*1 = 1
2	**FOR** each feature	--	--	--	--
2.1	Show a superimposed of the historical distribution of this feature (when balanced as expected) on the current distribution of the feature	System	--	--	--
2.2	Check if the historical distribution is similar to the current distribution	Data Analyst	Understand (2)	1	1*2*1 = 2
2.3 – **IF** (20%)	**IF** the distribution is not similar as expected	--	--	--	--
2.3.1	Click on 'Bad quality' button	Data Analyst	Remember (1)	0.2	1*0.2*1 = 0.2
2.3.2	Set to false the 'good quality' property	System	--	--	--
2.3.3	Go to 6	System	--	--	--
3	Click on 'Good quality button	Data Analyst	Remember (1)	1	1*1*1 = 1
4	Set to true the 'good quality' property	System	--	--	--

5	Add feature distributions to historical data	System	--	--	--
6	Close the panel	System	--	--	--
				Total cost:	1+6*(2+0.2)+ 1 = 15.2

5.4 MODIFICATION AT TASK LEVEL (Matteo) (UPDATED)

5.4.1 Set Execution System Parameters (Matteo)

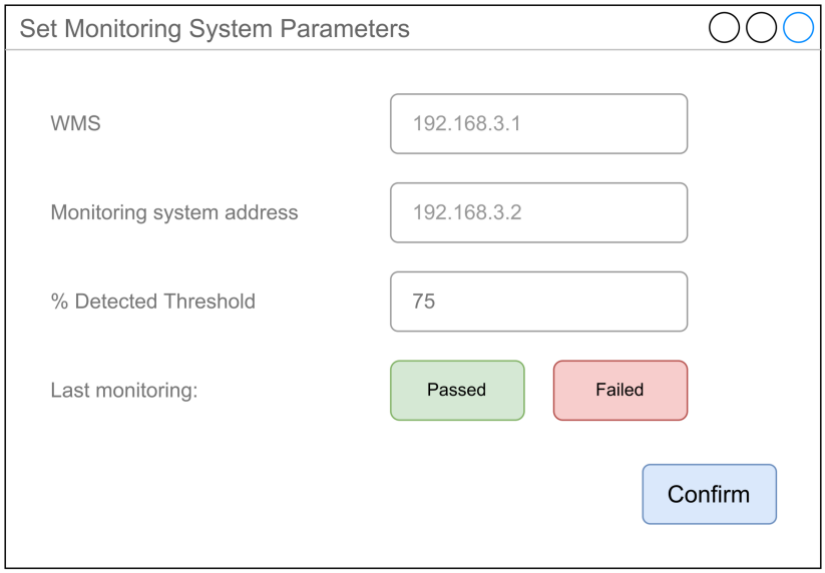
In the original task the ML Engineer has always to use his/her experience when setting the windows, increasing the cost of the task by a lot. We can lower the cost by letting the ML Engineer decide three parameters, only once and at the beginning of the lifecycle of the optimizer: the standard parameters for the windows and an additional parameter to dynamically increase/decrease the windows based on the status of the last monitoring of the optimizer (here assumed to be equal to 10%). The task now can be performed by a Data Analyst with a lower cognitive effort and a lower overall monetary cost.



Subtask	**Step**	**Actor**	**Cognitive effort**	**Occurrence**	**Cost**
1	Set Base station address	Data Analyst	Remember (1)	1	1*1*1 = 1
2	Set execution system address	Data Analyst	Remember (1)	1	1*1*1 = 1
3	Set Monitoring system address	Data Analyst	Remember (1)	1	1*1*1 = 1
4	Check last monitoring status	Data Analyst	Apply (3)	1	3*1*1 = 3
5 **IF (80%)**	If the optimizer passed the last verification	---	---	---	---
5.1	Click *Passed*	Data Analyst	Remember (1)	0.8	1*0.8*1 = 0.8

5.2	Increase the last monitoring and execution window by 10%	System	---	---	---
6. **ELSE (20%)**	---	---	---	---	---
6.1	Click *Failed*	Data Analyst	Remember (1)	0.2	1*0.2*1 = 0.2
6.2	Decrease the last monitoring and execution window by 10%	System	---	---	---
7	Click *Confirm*	Data Analyst	Remember (1)	1	1*1*1 = 1
				TOTAL COST	8

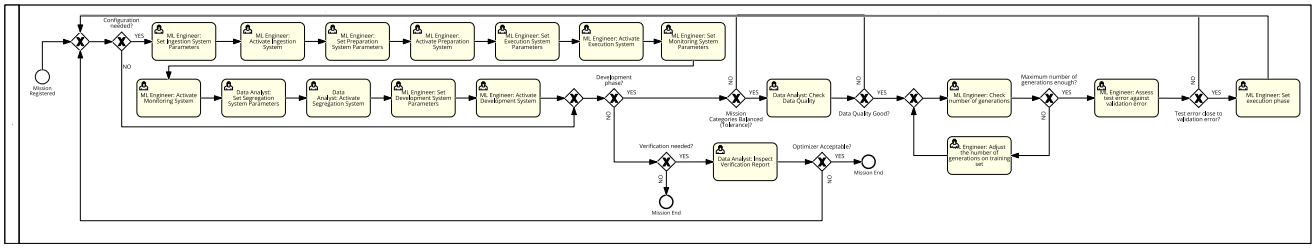
5.4.2 Set Monitoring System Parameters (Matteo)

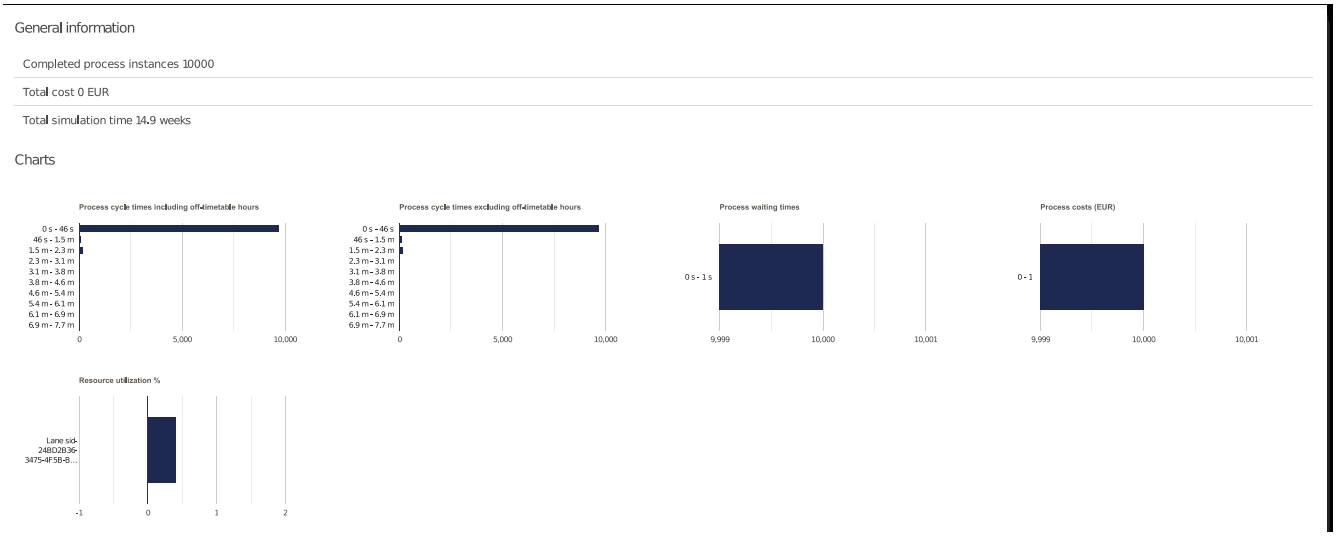


Subtask	**Step**	**Actor**	**Cognitive effort**	**Occurrence**	**Cost**
1	Set WMS address	ML Engineer	Remember (1)	1	1.4*1*1 = 1.4
2	Set monitoring system address	ML Engineer	Remember (1)	1	1.4*1*1 = 1.4
3	Check last monitoring status	ML Engineer	Apply (3)	1	3*1.4*1 = 4.2
4 **IF (80%)**	If the optimizer passed the last verification	---	---	---	---
4.1	Click *Passed*	ML Engineer	Remember (1)	0.8	1.4*0.8*1 = 1.12

4.2	Increase the last monitoring window by 10%	System	---	---	---
5. **ELSE (20%)**	---	---	---	---	---
5.1	Click *Failed*	ML Engineer	Remember (1)	0.2	1.4*0.2*1 = 0.28
5.2	Decrease the last monitoring window by 10%	System	---	---	---
6	Set threshold for % of detected targets	ML Engineer	Analyze (4)	1	1.4*1*1 = 1.2
7	Click *Confirm*	ML Engineer	Remember (1)	1	1.4*1*1 = 1.4
				TOTAL COST	11.2

5.5 TO-BE SIMULATION (All)





Scenario Statistics

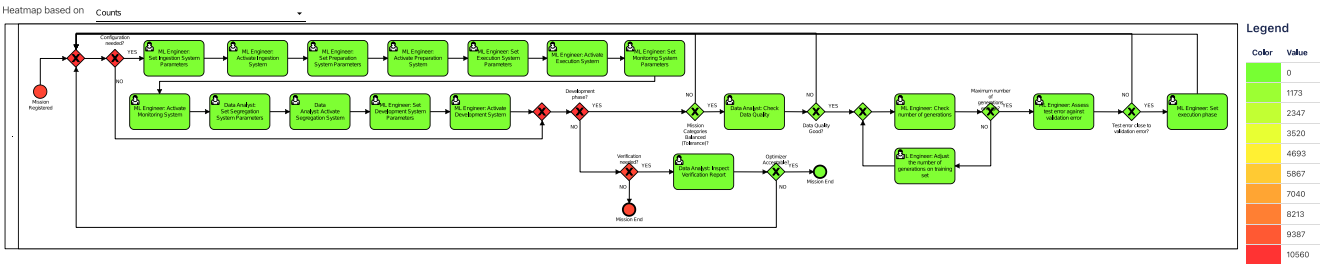
	Minimum	Maximum	Average
Process instance cycle times including off-timetable hours	0 seconds	7.6 minutes	3.7 seconds
Process instance cycle times excluding off-timetable hours	0 seconds	7.6 minutes	3.7 seconds
Process instance costs	0 EUR	0 EUR	0 EUR

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
Data
Analyst: Activate Segregation System	237	0 s	0 s	0 s	1.8 s	2 s	2.2 s	0 s	0 s	0 s	0	0	0	0	0	0
Data Analyst:
Set Segregation System Parameters	237	0 s	0 s	0 s	13.5 s	15 s	16.5 s	0 s	0 s	0 s	0	0	0	0	0	0
Data Analyst: Check Data Quality	499	0 s	0 s	0 s	13.7 s	15.2 s	16.7 s	0 s	0 s	0 s	0	0	0	0	0	0
Data Analyst: Inspect Verification Report	103	0 s	0 s	0 s	3.6 s	4.1 s	4.4 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer:
Activate Ingestion System	237	0 s	0 s	0 s	2.5 s	2.8 s	3.1 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer:
Activate Preparation System	237	0 s	0 s	0 s	2.5 s	2.8 s	3.1 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer:
Set Ingestion System
Parameters	237	0 s	0 s	0 s	11.3 s	12.6 s	13.9 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer:
Set Preparation System Parameters	237	0 s	0 s	0 s	5.1 s	5.6 s	6.2 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Activate Development System	237	0 s	0 s	0 s	2.5 s	2.8 s	3.1 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Activate Execution System	237	0 s	0 s	0 s	2.5 s	2.8 s	3.1 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Activate Monitoring System	237	0 s	0 s	0 s	2.5 s	2.8 s	3.1 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Adjust the number of generations on training set	318	0 s	0 s	0 s	7.9 s	8.8 s	9.7 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Assess test error against validation error	92	0 s	0 s	0 s	7.6 s	8.3 s	9.2 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Check number of generations	410	0 s	0 s	0 s	7.6 s	8.4 s	9.2 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Set Development System Parameters	237	0 s	0 s	0 s	13.9 s	15.5 s	16.9 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Set Execution System Parameters	237	0 s	0 s	0 s	15.1 s	16.8 s	18.5 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Set Monitoring System Parameters	237	0 s	0 s	0 s	10.1 s	11.2 s	12.3 s	0 s	0 s	0 s	0	0	0	0	0	0
ML Engineer: Set execution phase	47	0 s	0 s	0 s	2.5 s	2.8 s	3.1 s	0 s	0 s	0 s	0	0	0	0	0	0

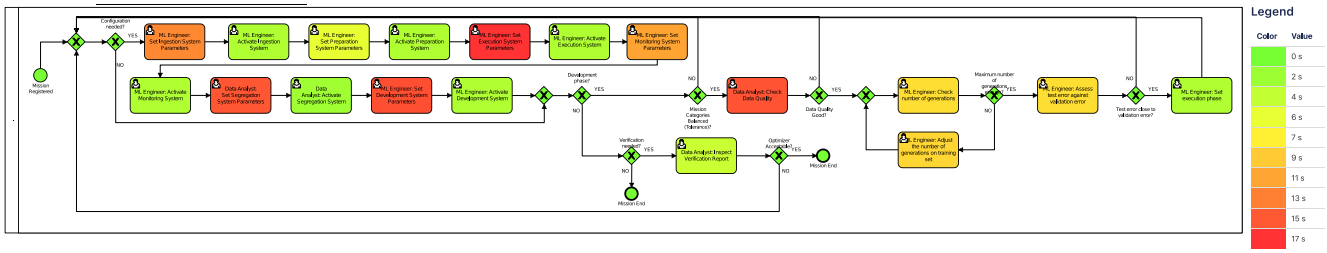
Heatmap

Heatmap based on Counts



Heatmap

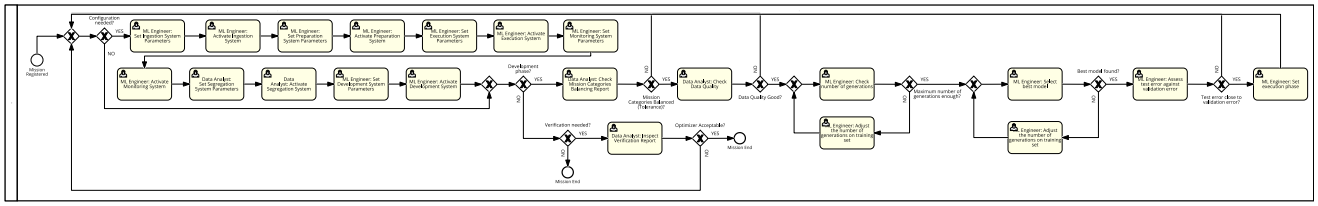
Heatmap based on Durations



6 PROCESS MINING (ALL)

6.1 NORMATIVE MODEL

We've used the AS-IS model as a normative model during the overall process mining.

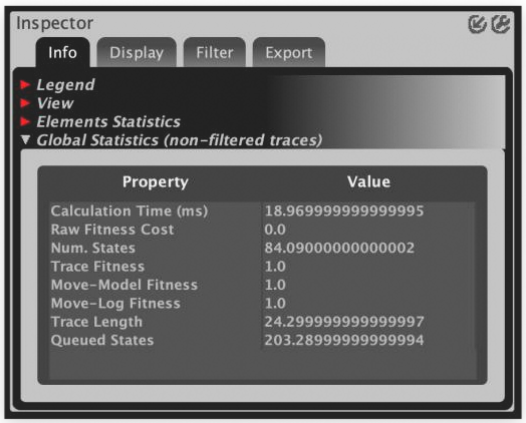


6.2 PROCESS LOG

In order to obtain the process log, the normative model has been simulated using BIMP as tool considering 100 tokens, 50% as branching proportion for each gateway, 1€ as task cost, and 1s as task duration. The process log was imported in Disco and exported as csv after a filtering operation (considering case ID, activity e complete timestamp). The log was then converted to XES and a BPMN model was mined using the ProM and Apromore tools.

6.3 FOUR QUALITY DIMENSIONS USING THE PROCESS LOG AND THE NORMATIVE MODEL

The BPMN was converted into a Petri net using the tool "Convert BPMN Diagram to Petri net (control-flow)". Conformance checking has been performed then using the "Replay Log on Petri Net for Conformance Analysis" tool, using the process log and the Petri net obtained from the normative model.



The **fitness** is 1 because the original log perfectly fits the mined model and noise was set to zero.

Precision : 0,80818

Generalization : 0,99916

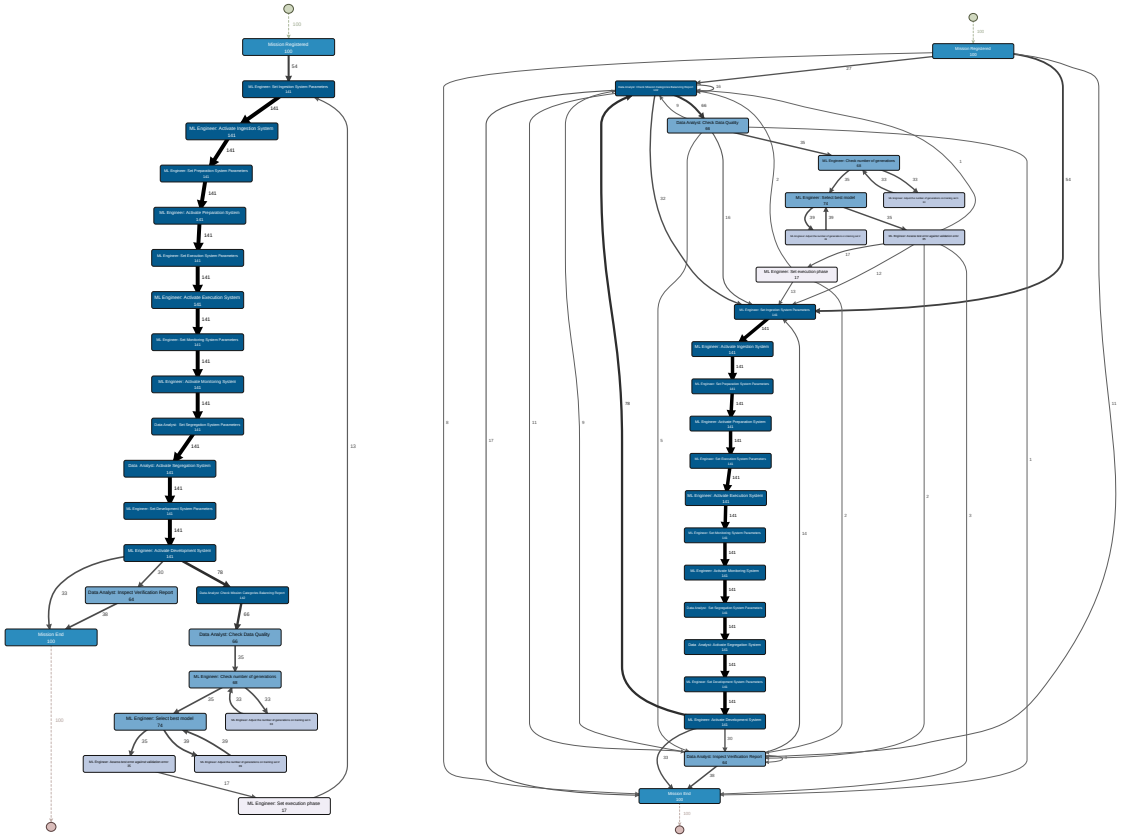
Precision is not equal to 1 because extra behavior from the normative model with respect to the original log is permitted, mainly because of the low number of event traces.

Generalization is close to 1, permitting us to consider additional behavior with respect to the original log.

Simplicity has been computed considering the overall number of gateways, sequence flows and activities, obtaining a value of **78** (13 gateways + 44 sequence flows + 21 activities).

6.4 TRANSITION MAP FROM THE ORIGINAL LOG (DISCO)

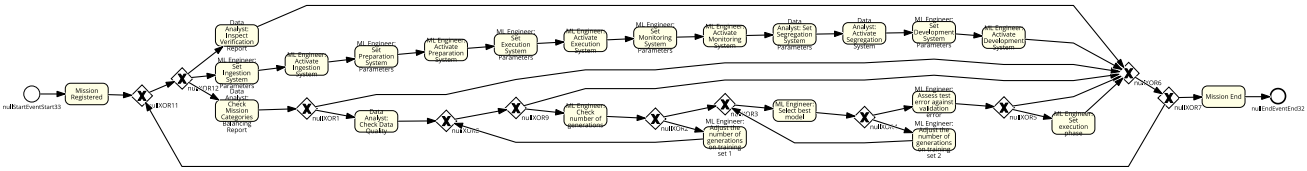
The process log has been used in order to extract the transition map using the Disco software tool.



All possible paths needed are present in the transition map.

6.5 PROM'S BPMN MODEL GENERATED FROM THE PROCESS LOG

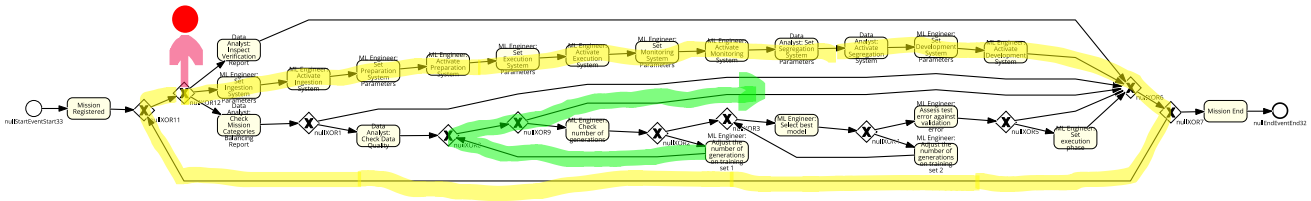
The model was mined using the BPMN Miner tool in ProM (Inductive Miner-Infrequent with noise threshold set to 0 in order to ensure maximized fitness). The mined model is clearly different from the original one.



As we can see from the BPMN mined, compared to the original model we have that:

- the path "Configuration needed? NO - Development phase? NO - Verification needed? NO", allowing the process to terminate directly, is not present. (red)

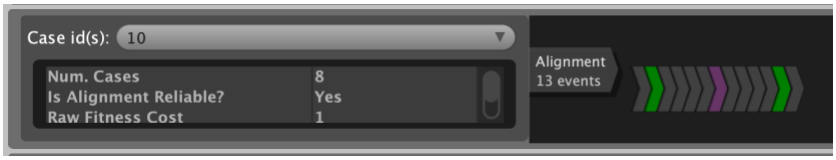
- the gateway equivalent to the question "Data quality good?" (nullXOR9) involved now the incoming arc from the adjustment of the number of generations, also allowing this adjustment to bypass the verification of the number of generations. (green)
- The path involving all settings and activations can now come back to the beginning without traversing any activity or end event. (yellow)



6.6 FOUR QUALITY DIMENSIONS USING THE PROCESS LOG AND THE PROM'S MODEL

The petri net has been generated converting the BPMN using the "Convert BPMN diagram to Petri Net (control-flow)" tool. Conformance checking has been performed using the "Replay Log on Petri Net for Conformance Analysis" tool, using the process log and the Petri net obtained from the ProM's model.

Considering the value of **fitness**, the trace fitness is not equal to 1 due to the miner being unable to detect a direct path between the start event and the end event. We also tried to add a dummy activity in the path but it was still unable to discover the path. The following screenshot shows an example of trace event that needs that path.



We measured **Precision** and **Generalization** using the "Measure Precision/Generalization" plugin.

Precision : 0,80360

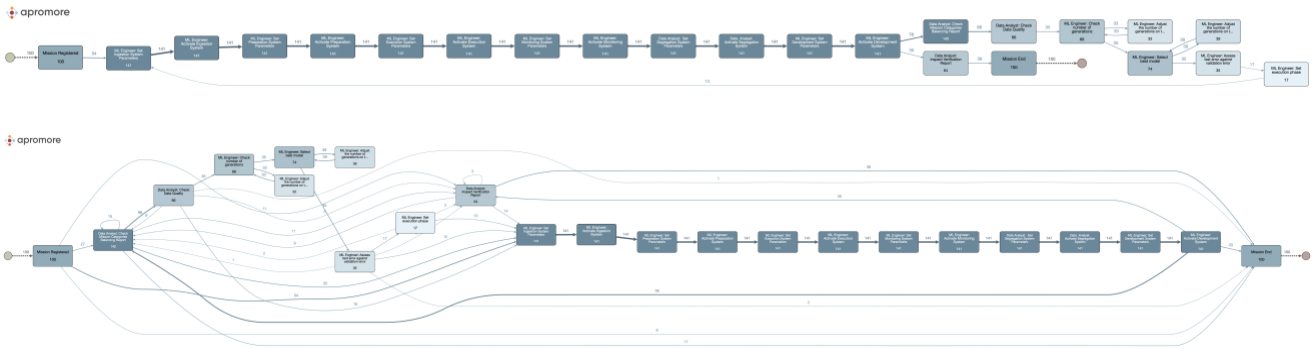
Generalization : 0,99915

We can see from the highlighted values that they are quite comparable to the ones obtained on the normative model. Precision is not extremely high, due to the non-perfect match with the behavior of the process log. Generalization is instead very close to 1, meaning that it does not restrict behavior just to the log, avoiding overfitting. **Simplicity** has been computed considering the overall number of gateways, sequence flows and activities, and assessed equal to **77** (11 gateways + 23 activities + 43 sequence flows). The model is as complex as the normative one.

6.7 TRANSITION MAP FROM THE ORIGINAL LOG (APROMORE)

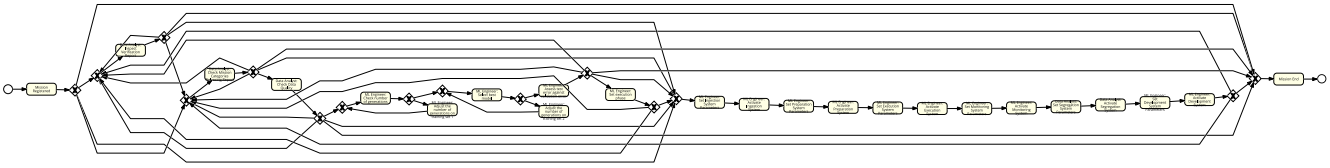
Using the **Apromore** tool, the transition map has been generated.

-



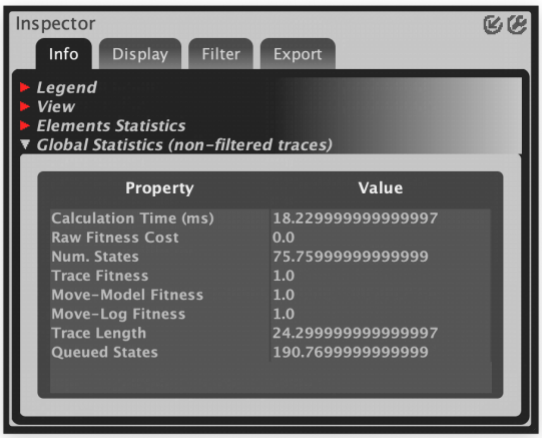
The latter is the spaghetti version. All paths are present in the transition map. We can see, from a comparison with Disco's transition map, that the two are identical.

6.8 APROMORE'S BPMN MODEL GENERATED FROM THE PROCESS LOG



We can notice, with respect to the ProM's model, the presence of a path from the beginning to the end. The mined model is equal to the normative one.

6.9 FOUR QUALITY DIMENSIONS USING THE PROCESS LOG AND THE APROMORE'S MODEL



As expected, **fitness** is equal to 1, since the model permits exactly all paths present in the normative model, allowing also the reachability of the end event from the start event.

Precision : 0,80818

Generalization : 0,99916

For the same reason, **precision** and **generalization** are exactly identical to the ones obtained on the normative model. Again, due to the other behaviors permitted from the mined model, precision is not equal to 1.

Simplicity has been computed considering the overall number of gateways, sequence flows and activities, and assessed equal to **103** (15 gateways + 23 activities + 65 sequence flows). This model is definitely more complex than the ProM one.

6.10 SUMMARY

Model	Fitness	Precision	Generalization	Simplicity
Normative	1.000	0.808	0.999	78
ProM	0.984	0.803	0.999	77
Apromore	1.000	0.808	0.999	103

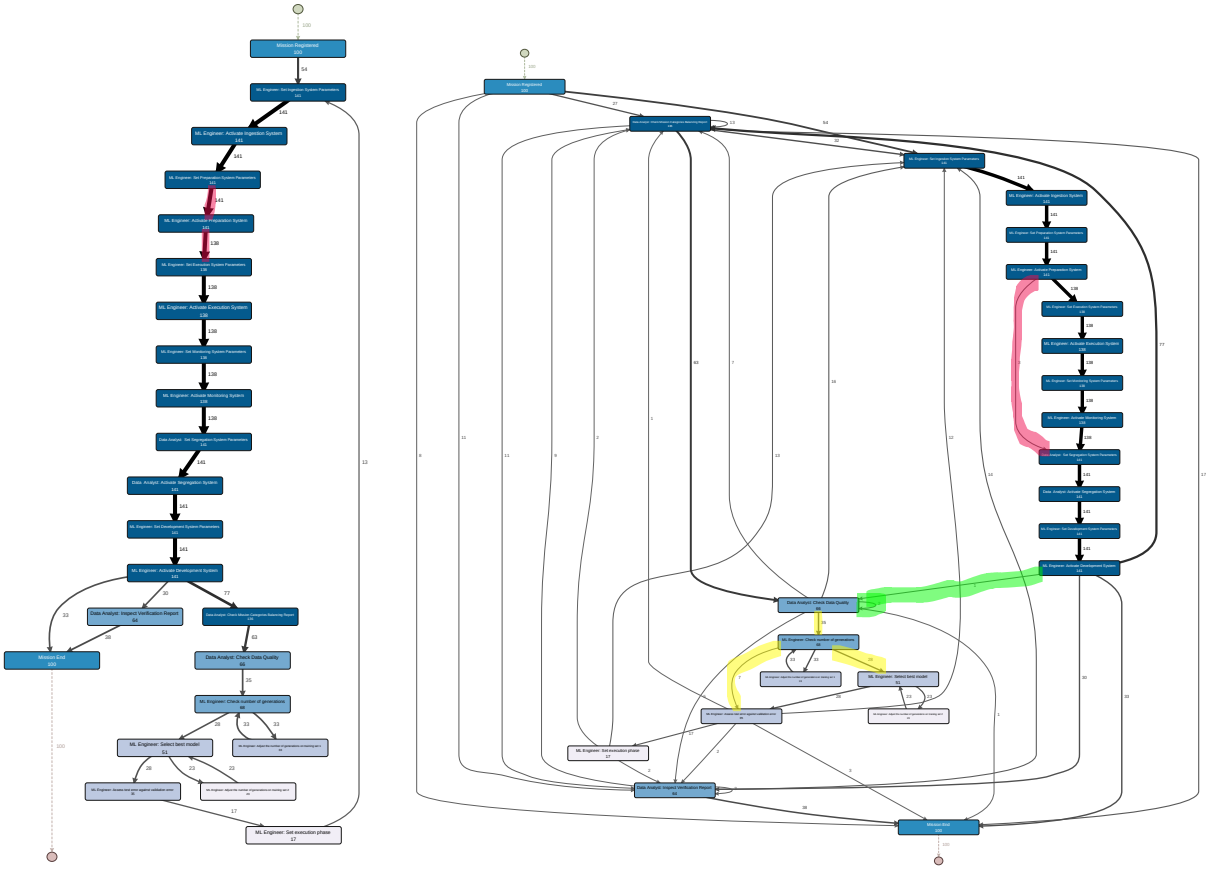
7 MODEL VIOLATION

By manually editing the csv file with Excel, we've modified some tasks to 3 cases in order to obtain 3 realistic violations to the model.

7.1 VIOLATION CASES

1. "Select best model" and "Adjust number of generations on training set" skipped: under the assumption of being able to skip hyper-parameterization by taking the optimal set of hyperparameters from the ones of a customer with a similar profile.
2. "Check mission categories balancing report" can be skipped after the first check because we work under the assumption of informing the user with the subsequent reconfiguration messages that the dataset was quite balanced (test passed the first time), so it should try to keep approximately the current distribution.
3. "Set" and "Activate" of execution and monitoring can be skipped after the first time they are set since the content of the parameters set does not change and the modules are already active.

7.2 TRANSITION MAP USING THE MODIFIED LOG (DISCO)



The transition map generated from the modified log highlight us the modified causes.

Case 1 (Grid search avoided) is highlighted by the yellow marker, allowing the flow to go directly from “Check number of generations to “Assess test error against validation error”.

Case 2 (Balancing avoided if already done) is highlighted by the green marker, allowing the flow to bypass the “Check mission categories balancing report” if already checked.

Case 3 (Set and activate avoided) is highlighted by the red marker, allowing the flow to bypass the set and activate tasks if the monitoring and execution systems are already set and activated.

7.3 VIOLATION CONFORMANCE CHECKING

Conformance checking has been applied using the violated log and the normative model. As expected, fitness is not equal to 1 anymore due to the violations introduced in the logs.

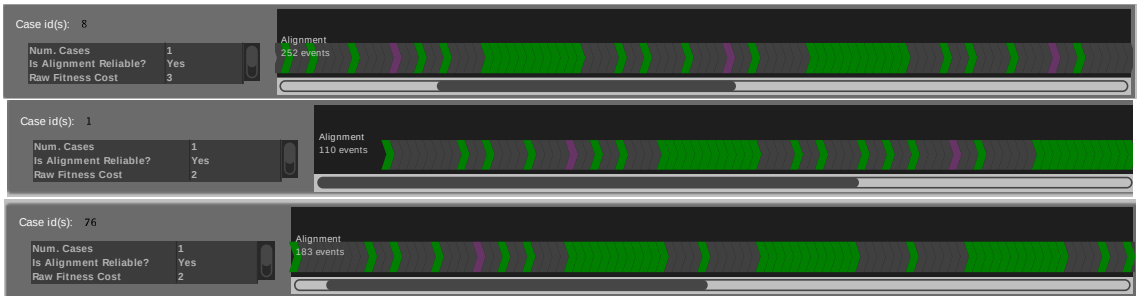
Precision : 0,81460

Generalization : 0,99901

Precision and generalization are comparable to the previous ones obtained using the normal logs.

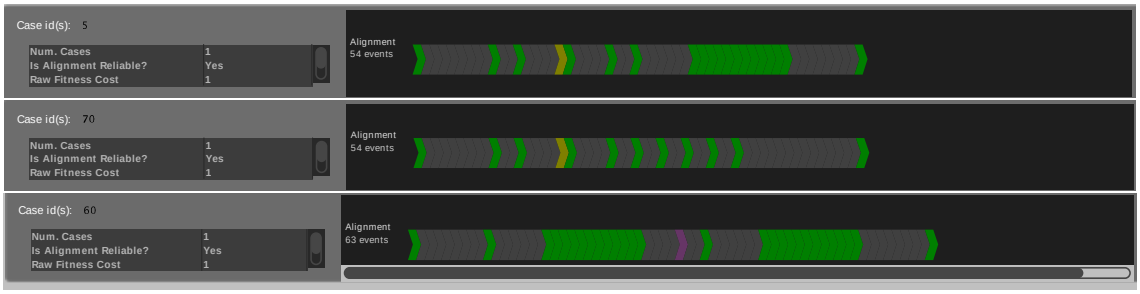
7.3.1 GRID SEARCH SKIP

The log traces 8, 1, and 76 highlights the skipping of tasks in the log which are instead present in the model.



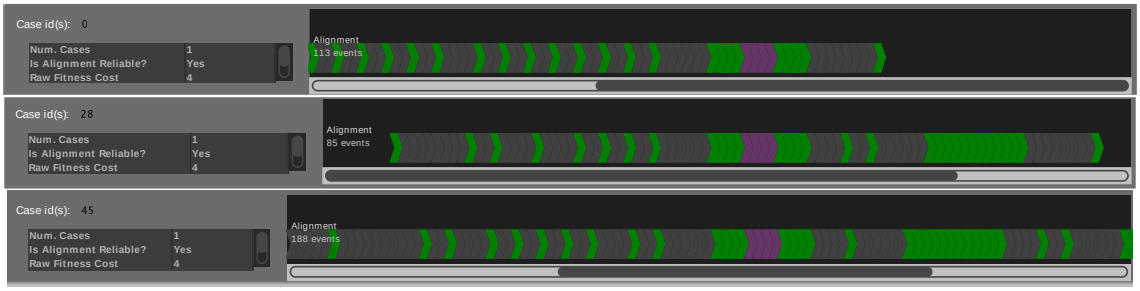
Hyper-parameterization is in fact skipped considering the optimal hyper-parameters involved with a similar customer.

7.3.2 CHECK BALANCING AVOIDED



The log traces 1, 70, and 60 highlights the additional presence of the task in the log which are instead not present in the model in that order, which corresponds to the "Data quality check" task. Due to the fact that we have 2 consecutive "check data quality" we have the presence of the inserted tasks in yellow. The log trace 60 shows only a skipped task due to the false condition in the gateway.

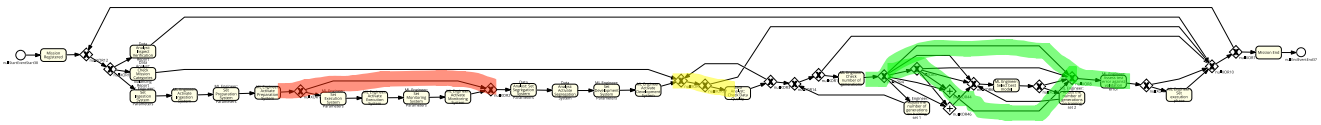
7.3.3 SETTING AND ACTIVATE ON MONITORING AND EXECUTION PHASES



Case ids 0, 28, and 45, modified using the process logs are highlighted in the images. The setting and activations are correctly skipped.

7.4 PROM'S BPMN MODEL GENERATED FROM VIOLATED LOG

The BPMN model is obtained using ProM by applying Inductive Miner (Infrequent version with noise=0) on the violated log.



We can clearly see the addition of a new gateway allowing us to skip the setting and activation of execution and monitoring phases (red marker).

The model can now permit to avoid data balancing check, going directly toward the data quality check, but only in the phase of configuration. We can instead notice how this behavior is not permitted on the non-configuration phase by this mined model (yellow marker).

The mined model can now permit to skip grid search going directly to assess test error and validation error. (green marker)

Fitness is not exactly equal to 1 because of the absence of a path from the start event and end event. This value can also be due to the absence of the data quality check without balancing check in the non-configuration phase.

Precision : 0,73239

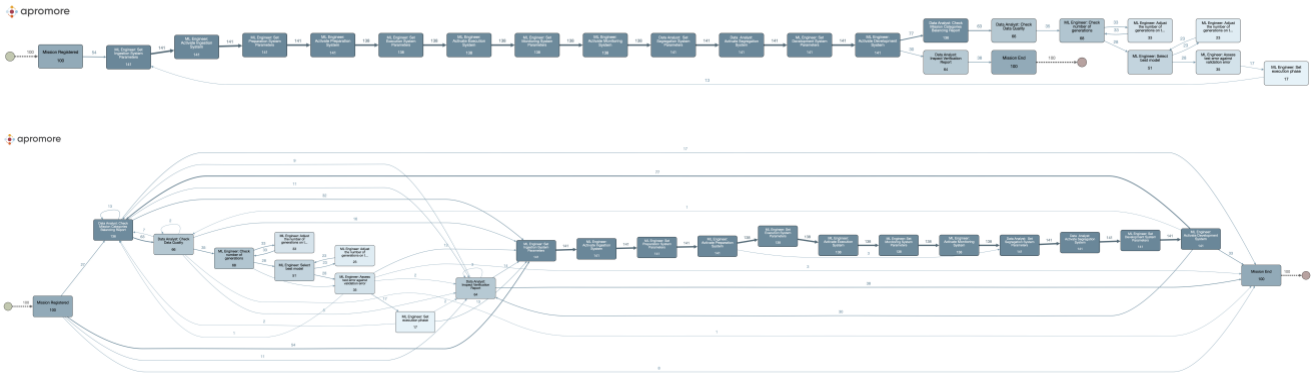
Generalization : 0,99887

Because of the extra behavior permitted by the model **precision** is not that high, e.g. adjusting the number of generations on the training set without involving the check number of generations task.

Generalization is very high, highlighting that the model is not overfitting.

Simplicity is calculated and obtained equal to 106 (23 tasks + 19 gateways + 64 sequence flows).

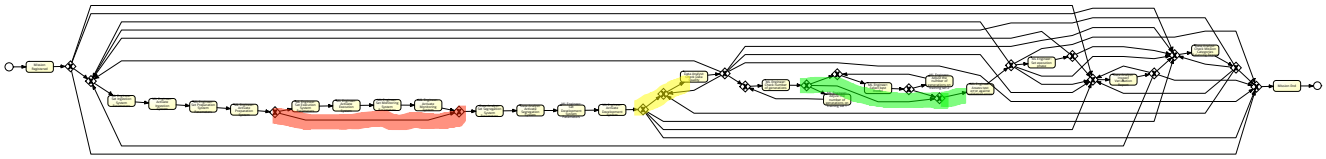
7.5 TRANSITION MAP GENERATED FROM VIOLATED LOG (APROMORE)



The transition map generated from the violated log is identical to the one generated from Disco.

7.6 BPMN MODEL GENERATED FROM VIOLATED LOG (APROMORE)

The BPMN model is obtained using the Apromore tool.



We can clearly see the addition of a new gateway allowing us to skip the setting and activation of execution and monitoring phases (red marker).

The model can now permit to avoid data balancing check, going directly toward the data quality check, but only in the phase of configuration. We can instead notice how this behavior is not permitted on the non-configuration phase by this mined model (yellow marker).

The mined model can now permit to skip grid search going directly to assess test error and validation error. (green marker)

The **fitness** computed is exactly equal to 1 in the case of Apromore, similarly to the non-violated case, because of the additional path from the start event and end event.

Precision : 0,73764

Generalization : 0,99888

Because of the extra behavior permitted by the model **precision** is not that high, e.g. checking the balance and then terminating.

Generalization is very high, highlighting that the model is not overfitting.

Simplicity is calculated and obtained equal to 112 (23 tasks + 19 gateways + 70 sequence flows).

7.7 SUMMARY

Model (violated)	Fitness	Precision	Generalization	Simplicity
Normative	0.995	0.814	0.999	
ProM	0.984	0.732	0.999	106
Apromore	1.0	0.738	0.999	112

8 FINAL CONSIDERATIONS

Model	Fitness	Precision	Generalization	Complexity
Normative	1.000	0.808	0.999	78
ProM	0.984	0.803	0.999	77
Apromore	1.000	0.808	0.999	103

Comparing the different models on the **original log**, the *Apromore* model achieved the best performances. Fitness, generalization and precision values are in fact the same as the normative ones. As a drawback, the number of elements involved are much higher. There is clearly a trade-off between the complexity of the Apromore model and its high values of fitness, precision and generalization.

Model (violated)	Fitness	Precision	Generalization	Complexity
ProM	0.984	0.732	0.999	106
Apromore	1.0	0.738	0.999	112

Again, Apromore is able to give us the best model obtained on the **modified log**, showing again the same trade-off among complexity, precision, generalization, and fitness.

Model	Fitness	Precision	Generalization
Normative normal	0.995	0.814	0.999
Normative violated	1.000	0.808	0.999

We can clearly see the violations effect on the fitness value of the normative model, highlighting the 9 violations we've performed.

From an **internal perspective**, the process obtained from the modified log provides some useful optimizations:

- The violations can help organizations reduce costs by identifying areas for process optimization, such as avoiding to reset parameters which remain the same.
- The violations can decrease the delay and as a consequence may reduce costs of the company. For example we avoid to perform grid search if a similar case has been found, considering a trade-off between effectiveness and speed.

Talking about **customer perspective**, the model obtained considering the violations, involved the following strengths:

- The model can improve the customer experience by identifying and addressing issues that may be causing delays, being more responsive and fast. Some tasks are in fact skipped under certain realistic assumption.

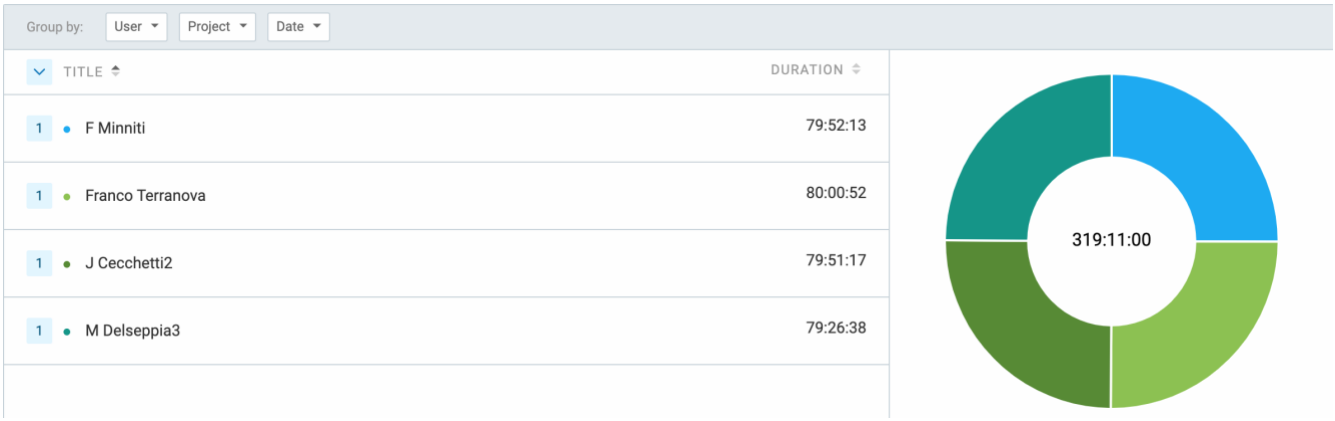
General weaknesses:

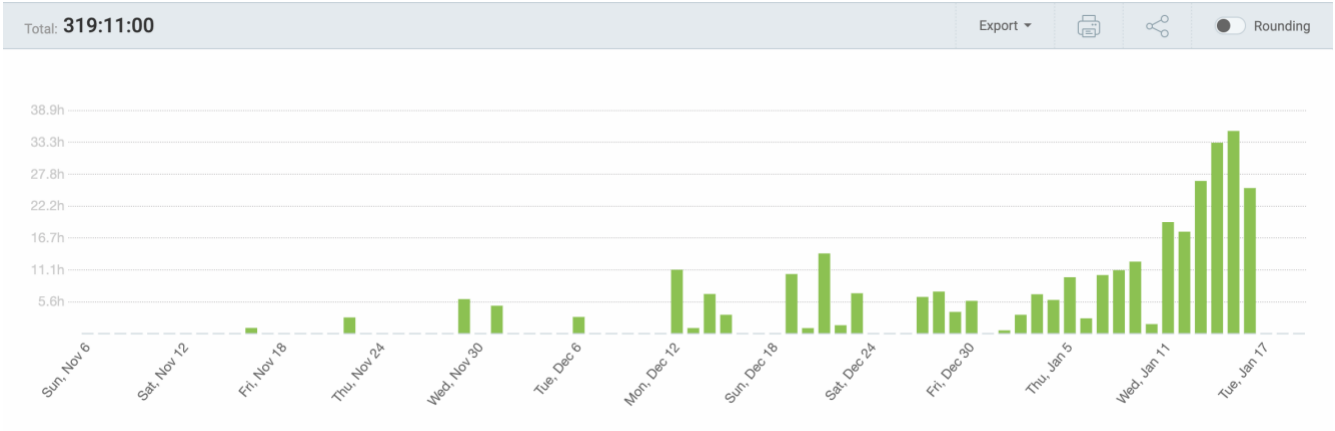
- The model may be complex to implement and require a significant investment in terms of time and resources.
- The model may require a large amount of data to be accurate, which may not be available or accessible.

- The model may require a high level of expertise to interpret the results and make recommendations for improvement.

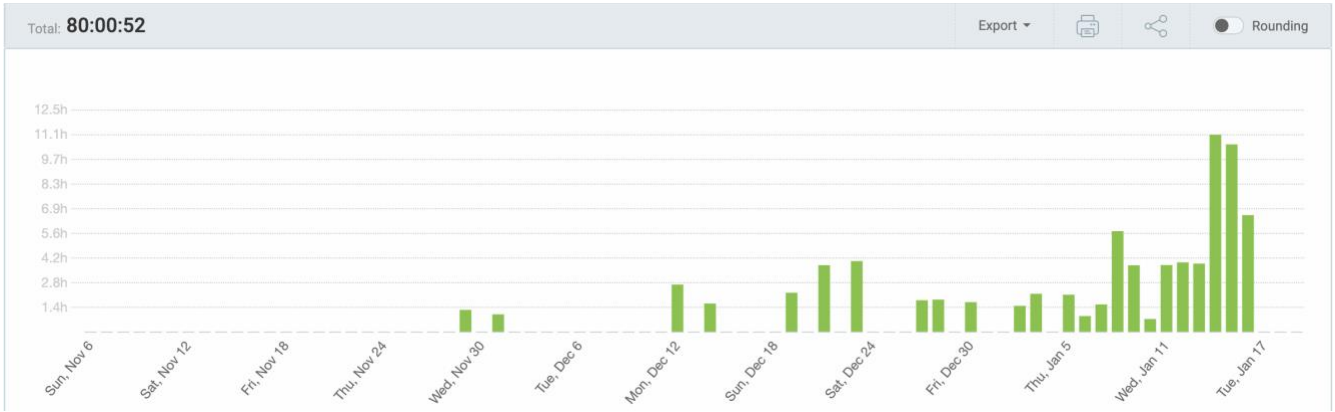
It's worth noting that this report is based on the general understanding of process mining and the new model should be thoroughly evaluated for its own strengths and weaknesses. Additionally, it's important to weight the benefits of the model against the costs and resources required to implement and maintain it.

CLOCKIFY

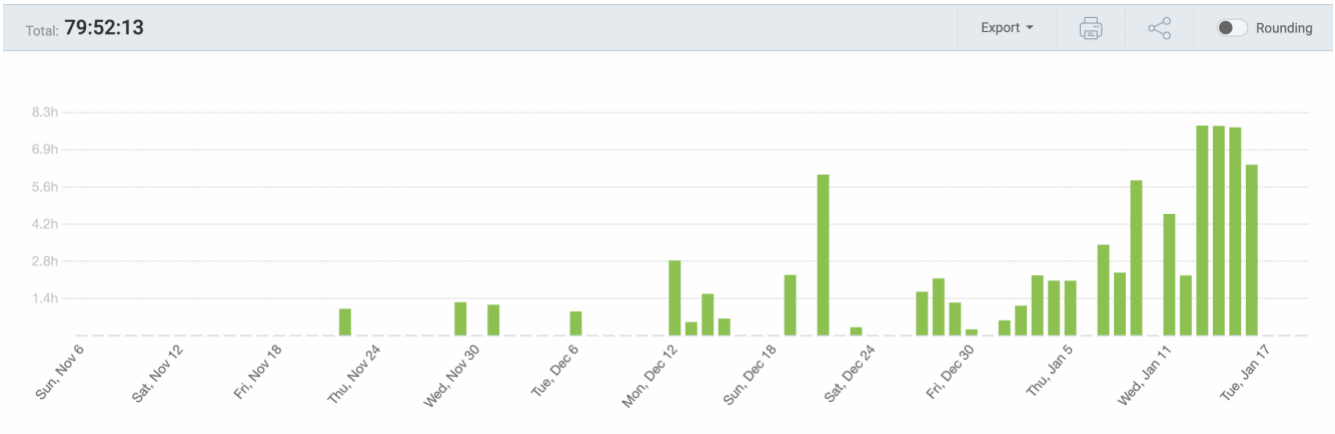




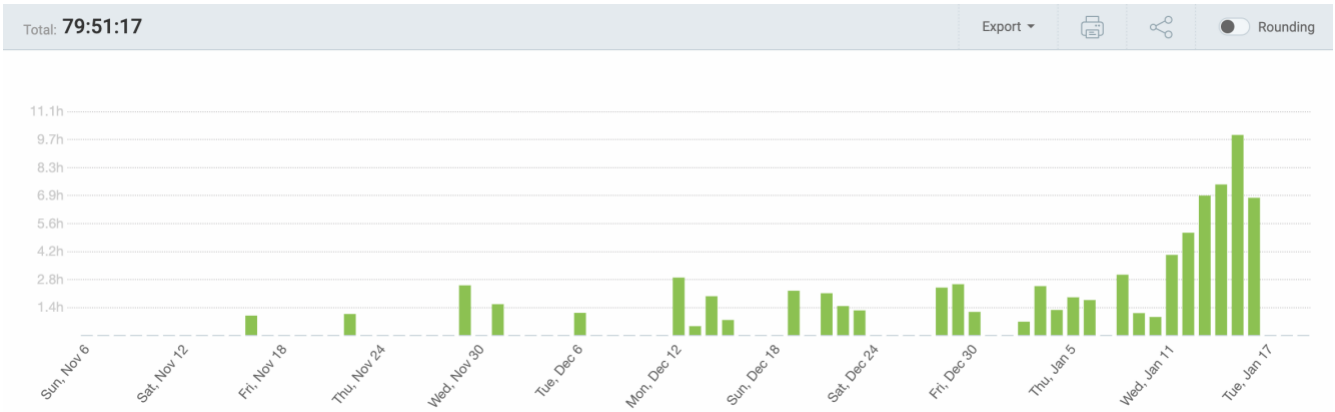
Franco



Federico



Jacopo



Matteo