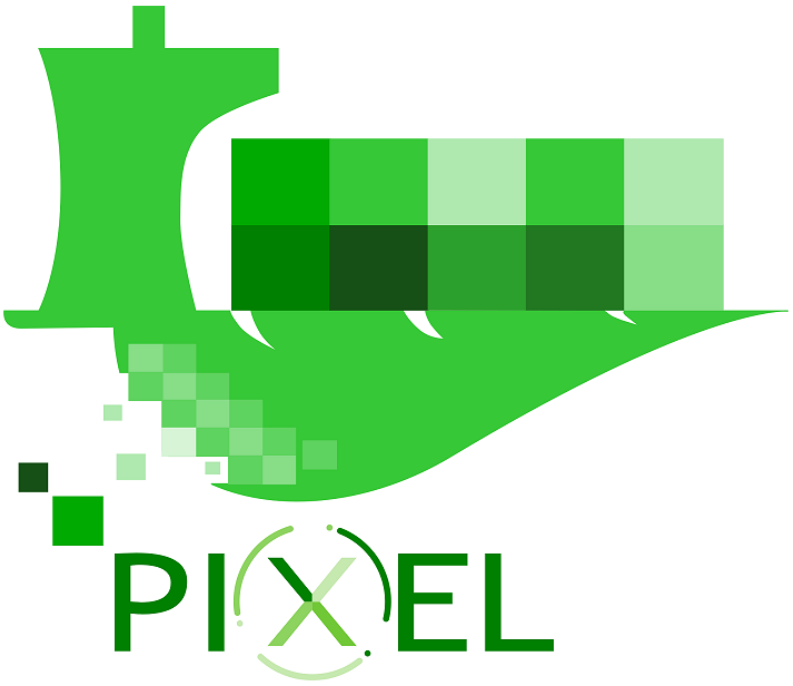
This project has received funding from the European's Union Horizon 2020 research innovation program under grant agreement No. 769355



D8.2 – Technical Evaluation v1

Deliverable No.	D8.2	**Due Date**	31-DEC-2019
Type	Report	**Dissemination Level**	Public
Version	1.0	**Status**	Intermediate
Description	This document presents the results of performance, user acceptance and security evaluation.		
Work Package	WP8		

Authors

Name	Partner	e-mail
Benjamin Molina	P01 – UPV	benmomo@upvnet.upv.es
Jose Antonio Clemente	P02 – PRO	jclemente@prodevelop.es
Flavio Fuart	P03 – XLAB	flavio.fuart@xlab.si
Damjan Murn	P03 – XLAB	damjan.murn@xlab.si
Tomaž Martinčič	P03 – XLAB	tomaz.martincic@xlab.si
Tim Lunar	P03 – XLAB	tim.lunar@xlab.si
Paolo Casoto	P04 – INSIEL	paolo.casoto@gmail.com
Charles Garnier	P05 – CATIE	c.garnier@catie.fr
Romain Quéraud	P05 – CATIE	r.queraud@catie.fr
Marc Despland	P06 – ORANGE	marc.despland@orange.com
Olivier Le Brun	P07 CREOCEAN	lebrun@creocean.fr
Luka Traven	P08 – MEDRI	luka.traven@medri.uniri.hr
Stjepan Piličić	P08 – MEDRI	stjepan.pilicic@medri.uniri.hr
Teodora Milosevic	P08 – MEDRI	teodoramilosevic1@gmail.com
Tamara Corsano	P09 – SDAG	t.corsano@sdag.it
Eleonora Anut	P09 – SDAG	e.anut@sdag.it
Cinzia Ninzatti	P09 – SDAG	c.ninzatti@sdag.it
Christos K. Papadopoulos	P10 – THPA	cpapadopoulos@thpa.gr
Athanasios Chaldeakis	P11 – PPA	ahaldek@gmail.com
Stefano Bevilacqua	P12 – ASPM	s.bevilacqua@monfalconeport.it
Thibault Guillon	P13 – GPMB	T-Guillon@bordeaux-port.fr
Leonidas Pitsikas	P14 – IPEOPLE	lpitsikas@peoplethinkbeyond.com
Annie Kortsari	P15 – CERTH	akorts@certh.gr

History

Date	Version	Change
26-JUN-2019	0.1	Document creation with first version of ToC
07-OCT-2019	0.2	Updated ToC
12-DEC-2019	1.0	Submission for internal review
18-DEC-2019	1.0	Submission to the PIC
31-DEC-2019	1.0	Submission to the EC

Key Data

Keywords	Data collection, data analysis, technical assessment, impact assessment, platform, product quality, quality in use, data quality
Lead Editor	Romain Quéraud, P05 – CATIE
Internal Reviewer(s)	Tamara Corsano (P09 – SDAG), Fabrice Klein (P13 – GPMB)

Abstract

The goal of this deliverable “Technical Evaluation v1.0” is to assess the technical work done in PIXEL up to M20. Most of the work done in the document is the application of what we defined in the previous deliverable D8.1 “Evaluation Plan”. The technical impact assessment is, as such, split into two distinct evaluations:

- The technical impact assessment of the PIXEL platform
- The technical impact assessment of the PIXEL use-cases

For the PIXEL Platform, we evaluate technical characteristics per modules at a laboratory level, such as memory consumption and CPU usage and obtain different KPIs. Those characteristics are derived from the ISO/IEC norm “Product Quality Model”. Evaluated modules are those defined in the table 5 of deliverable D8.1:

- Port and City Environmental Management Model
- Energy Demand Models
- Hinterland multimodal transport Models
- Environmental Pollution Models
- PIXEL Data Acquisition
- PIXEL Information Hub
- PIXEL Operational Tools
- PIXEL Integrated Dashboard and Notification
- PIXEL Security and Privacy Module

We have been able to make measurements for all the modules above. We stated some recommendations to follow in future developments of the modules, but without any critical points. Overall evaluation shows that modules are technically efficient.

The technical impact assessment of the PIXEL Use-cases plans to evaluate the project following the ISO/IEC norm “Quality In Use Model” for the user acceptance and “Data Quality Model”. As this evaluation requires the models to be close to a final integrated version, we were not able right now to evaluate the PIXEL Use-cases in this document. We chose instead, to define the evaluation methodology for the data that we plan to acquire. We show that TAM3-like and AIMQ-like questionnaires have been created and are ready to be disseminated to ports. We also know how to evaluate and compare questionnaires data:

- Comparing users’ behaviour between a pre-release and post-release of the PIXEL platform
- Analysing results with reference results as defined in the AIMQ research paper

The next version of the Technical Evaluation v2.0 (Deliverable D8.3) is expected at the end of the project in M36. Thus, we will ensure that the platform will be fully integrated and that end-users will have enough maturity with the platform to answer the questionnaires.

Statement of originality

Table of contents

List of tables

List of figures

List of acronyms

Acronym	Explanation
AIMQ	Methodology for Information Quality Assessment
CPU	Central Processing Unit
DAL	Data Acquisition Layer
EMP	Environmental Management Plan
GPMB	Grand Port Maritime de Bordeaux - Port of Bordeaux
IH	Information Hub
IT	Information Technology
ISO/IEC	International Organisation for Standardization / International Electrotechnical Commission
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
OS	Operating System
OT	Operational Tools
PAS	Port Activity Scenario; It is the model provided by T4.1.
PMIS	Port Management Information System
PPA	Piraeus Port Authority SA
TAM-3	Technology Acceptance Model 3
RAM	Random-Access Memory
REST API	Representational State Transfer Application Programming Interface
SDAG	Stazioni Doganali Autoportuali Gorizia
SILI	Sistema Informativo Logistico Integrato (Integrated Logistic Informated System), a system provided by Regione Friuli Venezia Giulia and managed by Insiel tomonitor and authorize entries to the Ports of Monfalcone and Trieste; it also monitors dangerous goods flows along the regional motorway network
THPA	Thessaloniki Port Authority

1. Introduction

1.1. Objectives & scope of the document

This document is the second document of WP8 and deals about the technical impact assessment of the PIXEL project. Its goal is to present the evaluation methods, the data collection methods, the data analysis and give out recommendations for the correct technical development of the project.

This document is structured in two main parts:

- The first one addresses the technical evaluation of the PIXEL platform
- The second one addresses the technical evaluation of the PIXEL Use Cases

For both parts, we take inputs from what has been defined in the previous deliverable (D8.1) in which we defined characteristics and sub-characteristics to evaluate. We also defined involved partners for the evaluation, and, as a result, we present a collaborative work in this document.

1.2. Deliverable context and structure

Table 1: Deliverable context

Keywords	Subjects
Objectives	The overall goal of WP8 is to evaluate the project in terms of (i) technical functioning and interoperability of all PIXEL Components, (ii) usability and (iii) results. The scope of D8.2 is to apply the methodology defined in D8.1 in order to gather data and derive different characteristics. Those characteristics will then be used to improve the PIXEL project.
Exploitable results	KPI and reports in order to improve the development of the PIXEL project.
Work plan	The D8.2 is directly related to: • WP4 for the technical evaluation of the models • WP6 which gather all elements to a laboratory working platform • WP7 which integrates the platform in the ports
Milestones	This deliverable contributes to MS10 – Final Evaluation (Means of verification: D8.3, D8.4 and D8.5 released and approved).
Deliverables	Detected inputs from: • D3.2: PIXEL Requirements Analysis • D3.4: Use cases and scenarios manual v2 • D4.2: PIXEL Models v2 • D6.3: PIXEL data acquisition, information hub and data representation v1 • D7.1: Integration Report v1 • D8.1: Evaluation Plan Detected outputs to: • D6.4: PIXEL data acquisition, information hub and data

	representation v2 • D7.2: Integration Report v2 • D8.3: Technical Evaluation v2
Risks	This deliverable deals with a risk identified in D8.1, relative to the delay of the platform trials beginning. As such, evaluation of the PIXEL use-cases cannot be achieved in this deliverable, this will be done in the second version. However, there is enough data to assess the already implemented parts, which means the PIXEL platform at a laboratory level.

1.3. Intended audience

This deliverable aims at providing feedback and guidelines to PIXEL developers. As such, we directly target responsible partners from WP4-5-6-7 in order to give them some KPIs that would influence future developments.

As we also consider the final users' feedback, it can also be read by those final users which would like to check how their evaluation is considered.

2. Applying evaluation and validation framework

2.1. Overall evaluation approach

The Technical Impact Assessment will be conducted for both the PIXEL Platform (for the evaluation of the IT part of the PIXEL project) and the PIXEL use cases (for the evaluation of the user acceptance and data quality). It will focus on:

- Technical performance;
- User acceptance;
- Information security and robustness.

To develop the technical impact assessment framework, we will base our work on three evaluation models. These models are based on the International Standards on System and Software Quality Requirements and Evaluation (Square):

- The first model (ISO/IEC 25010 Product Quality Method) is related to the evaluation of the PIXEL platform in regard to the properties of the software and the dynamic properties of the system.
- The second model (ISO/IEC Quality in Use Model) is directly linked with the assessment of the usage evaluation of the platform by end-users (ports for PIXEL).
- The last model (ISO/IEC 25012 Data Quality Model) is somehow complementary with the two others since it refers to the evaluation of the data provided by PIXEL platform.

For the technical impact assessment of PIXEL these models will be used, adapted or modified to our specific context. The ISO standard defines a list a characteristics and sub-characteristics for each of the three models. In order to clearly identify which ones of these characteristics are applicable to PIXEL, a survey has been shared with the whole consortium. Results of this survey have been described and analysed in D8.1, and reminded here in Appendix A. We will use them as a basis for the technical impact assessment.

For each characteristic or sub-characteristic listed in the ISO standards, the PIXEL consortium has agreed on which ones must be assessed and has established how to measure them. The evaluation criteria were also defined in the previous deliverable, D8.1. We will use those criteria to do the evaluation, in D8.2 and later on in D8.3.

WP8 is heavily dependent on other work packages that focus on the technical development. It has been noticed as a risk in D8.1 that integration (WP7 mainly) would likely not be advanced enough to be able to process to the user acceptance evaluation in this deliverable. Thus, this deliverable will only be able to assess the PIXEL platform, and not the PIXEL Use-Cases. We suggest instead the following plan:

- The PIXEL platform will be evaluated, according to the specifications defined in D8.1. We will be able to provide feedback in order to improve the platform.
- The PIXEL Use-Cases will not be assessed in this deliverable since the integration of PIXEL platform has just started. Despite some first work has been done to integrate PIXEL solution in ports, it is not enough to allow ports to effectively use the platform. That means that if they did not experience it in real condition, they will not be able to give feedback on it. In this deliverable, we present and explain the methodology that will be used for the evaluation on the PIXEL use-case.

We rely heavily on the work done in D8.1 for this. In fact, for the technical impact assessment of the PIXEL Use Cases, we defined the characteristics/sub-characteristics to be evaluated, and the calculation method in order to obtain the different KPIs, but we did not define in depth the questionnaires that are going to be disseminated. This will be the purpose of section 4 of D8.2, in which we will not only explain how we will disseminate the questionnaires and collect the results, but also explain how the data will be analysed.

2.2. Interrelation with other WPs and/or Tasks

Because this is the technical impact assessment deliverable of the project, we take input from all technical work packages, which means, for the technical impact assessment of the PIXEL platform:

- WP4: T4.1, T4.2, T4.3, T4.4
- WP6: T6.2, T6.3, T6.4, T6.5, T6.6

The task T5.3 of WP5 will be dealt in D8.3, as well as the predictive algorithms to be developed in T4.5. Task T6.1, on its side, was a design task were we evaluated different architecture approaches and came to a conclusion, does not contain anything to be evaluated.

For the technical impact assessment of the PIXEL Use-Cases, inputs are mainly coming from WP7 integration. Because it is a work package that recently started, and has not gone far enough yet, we will not be able to collect data from the questionnaires, and we will instead detail how we plan to do the analysis.

3. Technical Impact Assessment of the PIXEL platform

3.1. Port and City Environmental Management Model

The border between the port and the city is a conflictive point for urban, environmental and social tensions. For this, it is essential that port projects should have an environmental management plan (EMP), as was mentioned in deliverable D4.1.

Port and City Environmental model focuses on modelling the supply chain as a necessary step to identify emission sources and predict the impact of load transitions (energy cost, transport system overload, etc). This model should integrate partial models like energy model, pollution and transport demand models as well.

3.1.1. Assessment scenario

The evaluated run of the PAS builder uses data from the GPMB use cases. It is composed of:

- Three stopovers with one or two handlings each.
- The defined supply chain Cereale_SPBL_Export for two types of cereals: “Blé” and “Maïs” as defined in French is their data. This supply chain is composed of 9 operations, with either a fixed duration of a duration dependant on the throughput and amount in the cargo. This supply chain also defines periods of time in which the machines are usable.
- A list of machines available in GPMB. There are 9 machines that are referenced by the supply chain.

The fixed composition link is available as a reference for internal readers for reproducibility. It will also be useful for D8.3 to compare evaluation results and demonstrate improvements.

Tests are performed on a computer with:

- OS: Ubuntu 18.04.3 LTS.
- CPU: Intel Xeon E5-1650 v4; 4000 MHz max clock speed; 6 cores.
- RAM: 4*8192MB DDR4 with a 2400 MT/s clock speed.
- Drive: HDD model WDC WD10EZEX-60WN4A0; 1TB; 7200rpm; SATA 3.1.

3.1.2. KPI Data Collection and Results

We choose to measure the following KPIs for T4.1 and present the calculation process and results below.

1. Straightforward task accomplishment:

Yes. By developing the PAS for T4.1, focus was made around customization, meaning that some steps will be unnecessarily heavy for some users, while quite useful for others. Still, we chose to keep that feature.

2. Portion of completed requirements:

67% (relevant) or 38% (all). We evaluated this KPI on all common functional requirements defined in D3.2. For all requirements, we stated if they were relevant or not for our specific model, then we stated if we completed them. Table below summarizes this assessment. Readers must keep in mind that a lot of KPI are directly linked with integration of the models in the Operational Tools (WP6) and test and validation in real conditions (WP7). This work is still an on-going work in WP6 and WP7 and will be completed.

Table 2: List of completed requirements for the Port and City Environmental Management Model

Common functional Requirements	Priority (as define in PIXEL requirements)	Relevant (defined by task leader)	Completed	Comments
Import historical Data (36)	M	1	0	It is planned to integrate with PIXEL information hub to retrieve historical data.
Interaction with models (41)	M	1	1	We are able to input/output jsons from other modules to interact with their models.
Anomaly and event list (44)	M	1	0	User preference is not yet fully integrated

Anomaly and event detection (45)	C	0	0	
Homogenize Data (61)	M	1	1	We provide DataModels.
Catalogue of models (62)	M	1	1	We provide DataModels.
Detection of anomalies (63)	M	1	0	We will raise a warning to the user in case of configuration/input mismatch.
Feedback (64)	S	1	1	We have "comments" fields in the DataModels.
Centralized user administration system (65)	M	0	0	
Configurable Dashboard (66)	M	0	0	
UI 0tification System (67)	M	0	0	
Port Operational KPI list (70)	M	0	0	
Operation Interface (71)	M	0	0	
Analyse historical data (81)	M	1	1	By providing jsons
Support for manually provided data (86)	M	1	1	By editing jsons
Discovery service for data (104)	M	0	0	
Visualization of data (105)	M	0	0	

3. Mean CPU utilization: 12.91 %.

We use python subprocess to start a new process for the model run only and use the psutil python package to measure CPU utilization over time. Psutil gives a per-core usage in percents, which sums to 1200% with our configuration, which we bring it back to a 0-100% scale. Results are shown below.

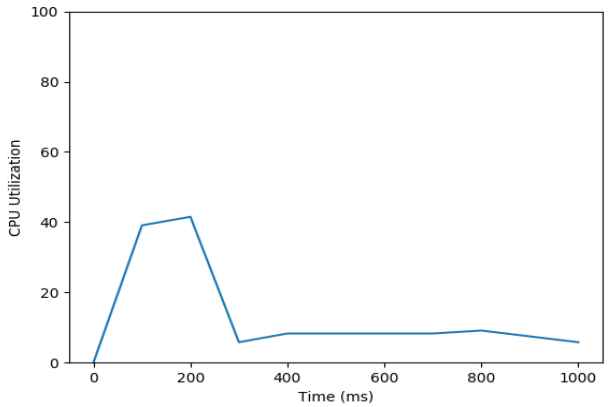


Figure 1: CPU Utilization over time for the Port and City Environmental Management Model

We see from the plot and from the recorded data that over the 4 seconds run, CPU has a mean utilization of 12.91 %.

4. Mean memory usage: 4.78 MB.

We use the massif tool of valgrind in order to monitor the execution of our python script, which gives memory evolution along a single run of the energy module. Result of the run is shown below:

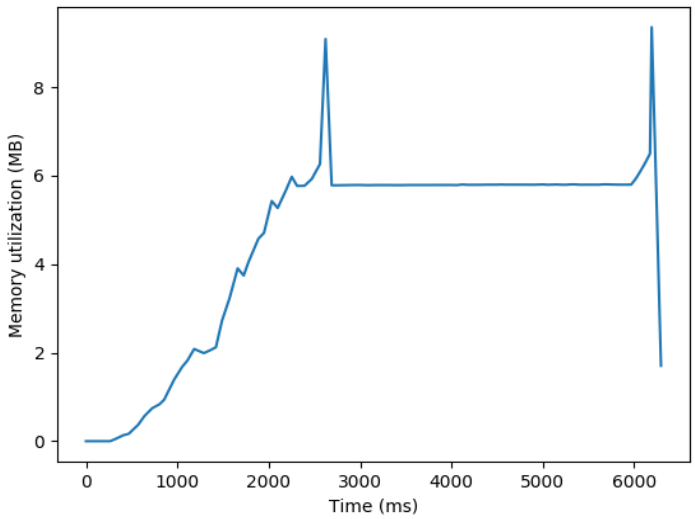


Figure 2: Memory Utilization over time for the Port and City Environmental Management Model

We also have a csv output with those values, one value per 0.1 seconds. It allows us to calculate the mean memory usage: 4.78 MB.

5. Maximum memory usage: 9.35 MB.

The same method as above is applied. We find the maximum memory usage to be: 9.35 MB.

6. Maximum processing power used: 498.3 %.

Using the same method as for the mean CPU Utilisation, we keep the summed data and don't divide it with the number of probes. Results are shown below:

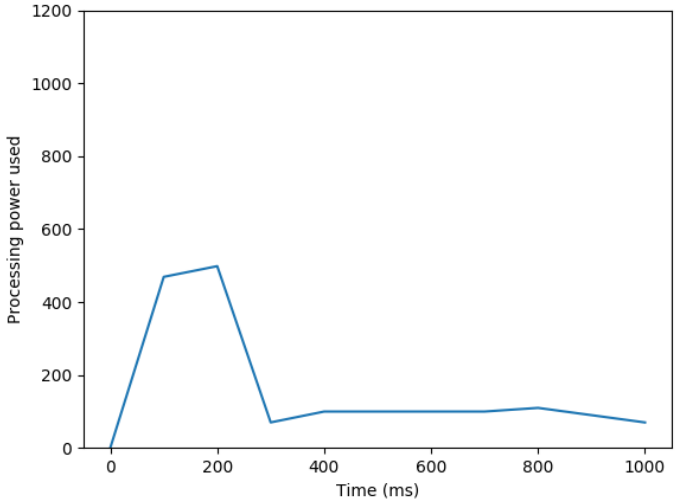


Figure 3: Processing power used over time for the Port and City Environmental Management Model

We see from the plot and from the data collected, that the maximum processing power used is 498.3 %.

7. Simultaneous requests: > 1000.

We used Dask to run parallel computations using our model. We show below results of time per process, augmenting the number of processes that run at the same time. It shows that we can run more than 1000 processes without augmenting processing time, asserting that our model can be used on simultaneous cases.

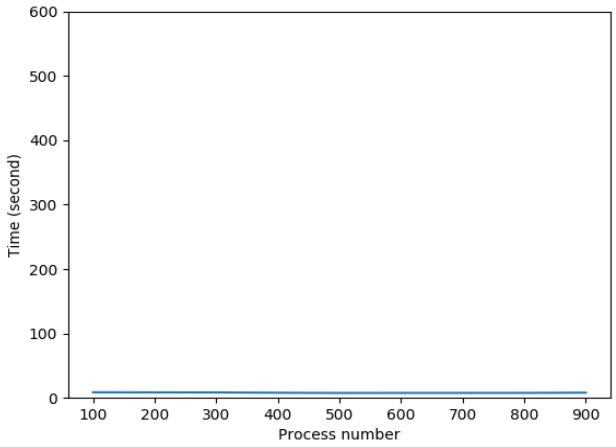


Figure 4: Average execution time over multiple batch size of process for the Port and City Environmental Management Model

8. Percentage of modularity: 100%.

The PAS aims to be executed with a JSON input and produce a JSON as output. It is 100% modular as we can execute every step by itself, making manual-modifications on the inputs and outputs.

9. Percentage of reusable assets: 100%.

The aim of the PAS is to be integrated in the PIXEL platform and adapted in the different ports. It has been built with the aim to be reusable.

Results for all KPIs are summarized in the table below:

Table 3: KPI summary for the Port and City Environmental Management Model

KPI	Result
Straightforward task accomplishment	No
Portion of completed requirements	67% (relevant) / 38% (all)
Mean CPU utilisation	12.91%
Mean memory usage	4.78 MB.
Maximum memory usage	9.35 MB.
Maximum processing power used	498.3%
Simultaneous requests	> 1000
Percentage of modularity	100%
Percentage of reusable assets	100%

3.1.3. Problems Faced and Recommendations

The "*Portion of completed requirements*" KPI allows us to detect that some of the requirements have not been implemented yet, mainly concerning the customization that would allow us to detect inconsistencies. It is a known problem that will be corrected during integration taking into account GPMB advice.

The "*Simultaneous requests*" KPI shows very good results on the above assessment scenario. Only the evaluated CPU consumption seems to be high, but this may be due to the global python environment. So running the model multiple times in the same python environment works, but we have to be careful if multiple run occurs in separate environments (e.g. isolated docker containers).

The "*Mean memory usage*" and "*Maximum memory usage*" KPIs shows small memory consumption, however we have to remember that the test was realised for one supply chain only and a relatively small input charge. Memory consumption shouldn't reach very high level, but this needs to be monitored when we will use the model on larger time periods.

3.2. Energy Demand Model

Energy demand models focus on modelling the port's energy demand and production to provide information about energy availability, reliability and efficiency. By setting parameters summarizing the port activity mechanisms, the Port and City environmental management model and the PAS are able to estimate the according port activity scenarios and identify the main consumption items and the different possible consumption scenarios. The output of the PAS builder is a per machine allocation time list.

Using the time list provided by the PAS builder, we are able to compute metrics and time series of port consumption, and thus model the energy demand of the port. This model will give advices to ports on how to better use and optimize their energy consumption.

This model can be used to know the present power demand of ports operations as well as predict it. The port can then use it to test different configurations and "what-if" scenarios to know how the consumption evolves. The aim is also to couple it with the energy production prediction model.

3.2.1. Assessment scenario

Taking input from the PAS builder with the first supply chain of GPMB, we can run our energy demand model and predict port consumption on the global period and overtime. We can compute a consumption per machine, as well as per area.

Tests are performed on a computer with:

- OS: Ubuntu 18.04.3 LTS.
- CPU: Intel Xeon E5-1650 v4; 4000 MHz max clock speed; 6 cores.
- RAM: 4*8192MB DDR4 with a 2400 MT/s clock speed.
- Drive: HDD model WDC WD10EZEX-60WN4A0; 1TB; 7200rpm; SATA 3.1.

3.2.2. KPI Data Collection and Results

We choose to measure the following KPIs for T4.2 and present the calculation process and results below.

1. Straightforward task accomplishment: Yes.

By developing T4.2 model, focus was made around customization, meaning that some steps will be unnecessarily heavy for some users, while quite useful for others. Still, we chose to keep that feature.

2. Portion of completed requirements: 80% (relevant) or 50% (all).

We evaluated this KPI on all common functional requirements defined in D3.2. For all requirements, we stated if they were relevant or not for our specific model, then we stated if we completed it. Table below summarizes this assessment. Readers must keep in mind that a lot of KPI are directly linked with integration of the models in the Operational Tools (WP6) and test and validation in real conditions (WP7). This work is still an on-going work in WP6 and WP7 and will be completed.

Table 4: List of completed requirements for the Energy Demand Model

Common functional Requirements	Priority (as define in PIXEL requirements)	Relevant (defined by task leader)	Completed	Comments
Import historical Data (36)	M	1	0	It is planned to integrate with PIXEL information hub to retrieve historical data.
Interaction with models (41)	M	1	1	We are able to input/output jsons from other modules to interact with their models.
Anomaly and event list (44)	M	0	0	Anomaly should have been listed in the upper model (PAS Builder)
Anomaly and event detection (45)	C	0	0	-
Homogenize Data (61)	M	1	1	We use the DataModels defined in the PAS builder.
Catalogue of models (62)	M	1	1	Same as the "Homogenize Data" functional requirement.
Detection of anomalies (63)	M	1	0	We may be able to do it in the future while analysing time series and raise it in the future.
Feedback (64)	S	1	1	We have "comments" fields in the DataModels.
Centralized user administration system (65)	M	0	0	This has not to be dealt in the model-level.
Configurable Dashboard (66)	M	1	1	We provide visualisations examples that can be integrated in the UI.
UI 0tification System (67)	M	0	0	This has not to be dealt in the model-level.
Port Operational KPI list (70)	M	0	0	This has not to be dealt in the model-level.
Operation Interface (71)	M	0	0	This has not to be dealt in the model-level.
Analyse historical data (81)	M	1	1	We can take both historical/current/prediction jsons generated by the PAS builder.
Support for manually provided data (86)	M	1	1	The model will work as long as we have json data.
Discovery service for data (104)	M	0	0	This has not to be dealt in the model-level.
Visualization of data (105)	M	1	1	We provide visualisations examples.

3. Mean CPU utilization: 27.14 %.

We use python subprocess to start a new process for the model run only and use the psutil python package to measure CPU utilization over time. Psutil gives a per-core usage in percent, to that sums to 1200% with our configuration, which we bring it back to a 0-100% scale. Results are shown below.

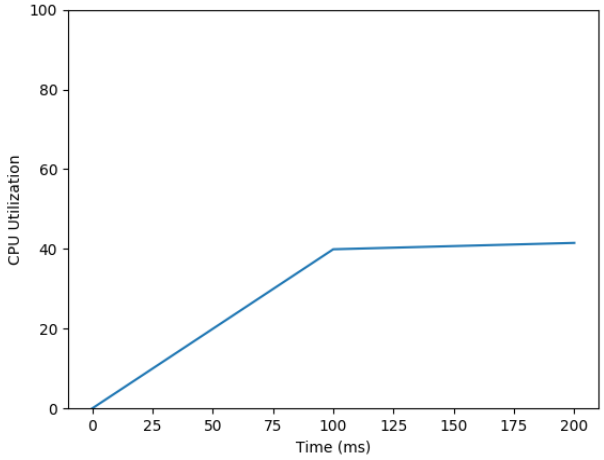


Figure 5: CPU utilization over time for Energy Demand Model

We see from the plot and from the recorded data that over the 4 seconds run, CPU has a mean utilization of 27.14 %.

4. Mean memory usage: 3.23 MB.

We use the massif tool of valgrind in order to monitor the execution of our python script, which gives memory evolution along a single run of the energy module. Result of the run is shown below:

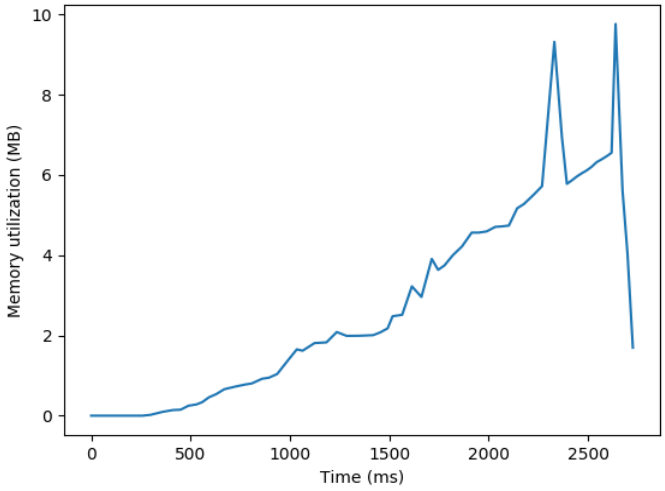


Figure 6: Memory utilization over time for Energy Demand Model

We also have a csv output with those values, one value per 0.1 seconds. It allows us to calculate the mean memory usage: 3.23 MB.

5. Maximum memory usage: 9.76 MB.

The same method as above is applied. We find the maximum memory usage to be: 9.76 MB.

6. Maximum processing power used: 498.2 %.

Using the same method as for the mean CPU Utilisation, we keep the summed data and don't divide it with the number of probes. Results are shown below:

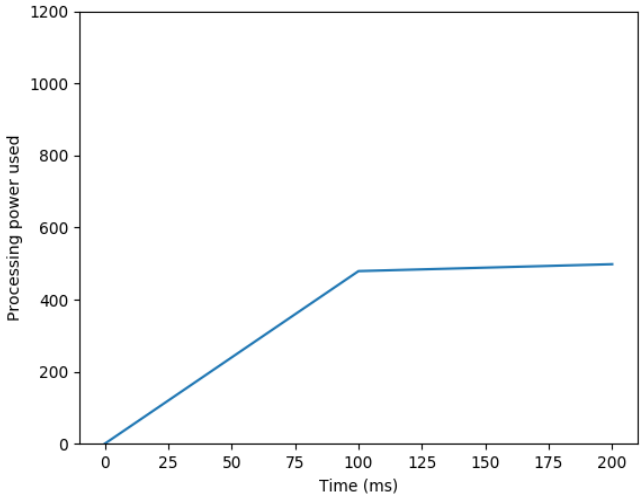


Figure 7: Processing power used over time for Energy Demand Model

We see from the plot and from the data collected, that the maximum processing power used is 498.2 %.

7. Simultaneous requests: > 1000.

We used Dask to run parallel computations using our model. We show below results of time per process, augmenting the number of processes that run at the same time. It shows that we can run more than 1000 processes without augmenting processing time, asserting that our model can be used on simultaneous cases.

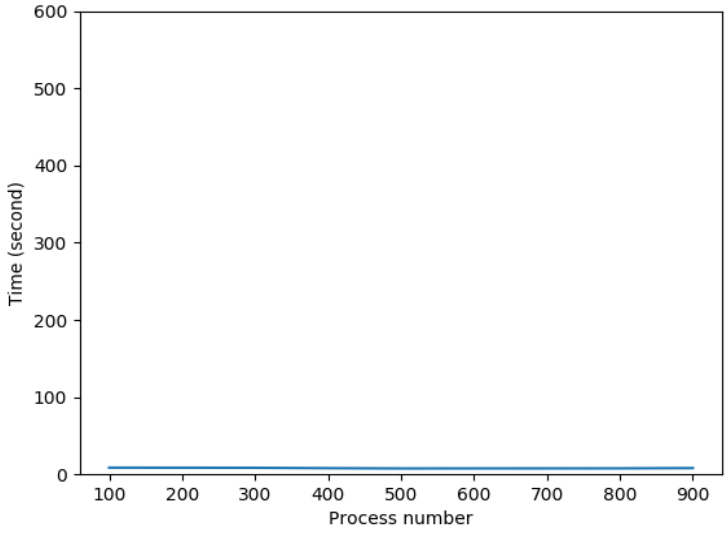


Figure 8: Average execution time over multiple batch size of process for Energy Demand Model

8. Percentage of modularity: 100%.

The energy demand model, as part of the PAS, aim to be executed with a JSON input and produce a JSON as output. As the PAS is 100% modular, we can define the energy demand model as 100% as well.

9. Percentage of reusable assets: 100%.

The aim of the energy demand model is to be integrated in the PIXEL platform and adapted in the different ports. It has been built with the aim to be reusable.

Results for all KPIs are summarized in the table below:

Table 5: KPI summary for the Energy Demand Model

KPI	Result
Straightforward task accomplishment	No
Portion of completed requirements	80% (relevant) / 50% (all)
Mean CPU utilisation	27.14 %
Mean memory usage	3.23 MB.
Maximum memory usage	9.76 MB.
Maximum processing power used	498.2 %.
Simultaneous requests	> 1000
Percentage of modularity	100%
Percentage of reusable assets	100%

3.2.3. Problems Faced and Recommendations

The Energy Demand Model is heavily relying on the PAS builder as a previous step, thus many of its characteristics are the same as the PAS. It is then recommended to improve first the PAS model in order to have a better input to feed to the Energy demand model before trying to directly improve the energy demand model.

One complicated thing during the technical impact assessment of the Energy demand model was to find a way to measure the simultaneous requests KPI, as it is completely dependent on the hardware. We specified the configuration in the assessment scenario in order to have reproducibility.

3.3. Hinterland multimodal transport Models

Hinterland multimodal transport models are aimed at describing the impact of port activities (in terms of vessels reaching the Port of Monfalcone for loading and unloading activities) on the regional traffic, by comparing different transportation modes (e.g.: trucks, trains, etc.).

Models, more specifically, have been developed in order to achieve the following goals:

- To provide a better understanding of all the different effects that goods transportation may produce: environmental effects (e.g.: air pollution, noise pollution, etc.), economic effects (e.g.: road maintenance costs due to transportation of heavy goods by using trucks) and social effects (e.g.: impact of the trucks' flow moving in and out the port on the local population living near the port facilities);
- To move towards a better prediction of potential truck congestion (and consequent traffic jams) at the port entrance, by considering both marine and terrestrial flows reaching and leaving the port for loading and unloading operations. In particular models are aimed at supporting operators during congestion

management, by suggesting alternative solutions (such as reaching the SDAG trucks parking area) in order to overcome congestion and release some of the pressure on port's entrances.

This model can be used to predict congestion on a daily basis according to incoming data or, at the same time, to perform what-if scenario when vessel arrivals need to be planned.

3.3.1. Assessment scenario

In order to perform an effective assessment of proposed models, a web-based modelling tool has been developed. The tool, available as open source on project repository, is based on the ASP MVC Core technology and adopts a SQL Server 2019 Express database as a temporary replacement of the IH module to store and retrieve the following information:

- Calls for ship, including weight and classification of loaded and unloaded goods;
- Data about incoming trucks, defined as sum of planned trucks and expected truck (as estimated by the predictive algorithm for truck traffic);
- Data about parking lot availability at both SDAG and Port of Monfalcone;
- Multimodal routed defined for each kind of good and related values of energy consumption and pollution (e.g.: So2 per Ton/Km).

Testing involves scenarios which are the most critical according with the experience of operators of Monfalcone's Port (e.g.: car loading activities at Monday morning).

Given the set of input calls for ship for a specific day, the model will try to evaluate:

- The impact required to transport specific unloaded goods (e.g.: slabs) for each multimodal transportation route;
- The probability of truck congestion having place at port entrance;
- The probability of contingency plans (e.g.: move trucks to SDAG) to prevent or solve congestion according with real time data about parking lot availability (at both Port of Monfalcone and SDAG).

The model does not present a significant complexity in terms of calculation; a standard laptop has been used for validation purposes.

3.3.2. KPI Data Collection and Results

Following KPIs have been selected, according to previously described models, in order to represent assess the effectiveness of the model proposed in T4.3:

1. Straightforward task accomplishment: Partly.

Proposed model emphasizes some aspects and issues which are particularly relevant for Port of Monfalcone (e.g.: unloading of bulk goods which need to be carried directly to the destination, without usage of warehouses) and may not completely fit in different ports.

2. Portion of completed requirements: 87,5% (relevant) or 53% (all).

KPI is evaluated by considering all common functional requirements defined in D3.2. In particular, for each requirement, relevance and completeness have been considered. Readers must keep in mind that some KPI are directly linked with integration of the models in the Operational Tools (WP6) and test and validation in real conditions (WP7). This work is still an on-going work in WP6 and WP7 and will be completed.

Table 6: List of completed requirements for the Transport Model

Common functional Requirements	Priority (as define in PIXEL requirements)	Relevant (defined by task leader)	Completed	Comments
Import historical Data (36)	M	1	0	It is planned to integrate with PIXEL information hub to retrieve historical data, in particular about truck and vessel arrivals.
Interaction with models (41)	M	0	1	Data provided by the model, in terms of both raw and aggregated data, can be shared with other models if required.
Anomaly and event list (44)	M	0	1	The model is aimed at identifying anomalies in terms of traffic congestion at the port's entrance
Anomaly and event detection (45)	C	0	0	-
Homogenize Data (61)	M	1	1	Usage of FIWARE data models
Catalogue of models (62)	M	1	1	Usage of FIWARE data models
Detection of anomalies (63)	M	1	1	The model is aimed at identifying anomalies in terms of traffic congestion at the port's entrance
Feedback (64)	S	0	0	
Centralized user administration system (65)	M	0	0	This has not to be dealt in the model-level.
Configurable Dashboard (66)	M	1	1	Several dashboards have been included.
UI Notification System (67)	M	0	0	This has not to be dealt in the model-level
Port Operational KPI list (70)	M	0	0	This has not to be dealt in the model-level
Operation Interface (71)	M	0	0	This has not to be dealt in the model-level
Analyse historical data (81)	M	1	1	Historical data can be analysed and used in what-if scenario
Support for manually provided data (86)	M	1	1	The UI provides such functionality
Discovery service for data (104)	M	0	0	This hasn't to be dealt in the model-level.
Visualization of data (105)	M	1	1	Several dashboards have been included.

3. Computational requirements: the model does not present a significant complexity in terms of calculation. Model execution based on a weekly timeframe has no significant impact, in terms of both CPU and RAM, on a standard laptop. The model automatically performs re-evaluation every time a change in terms of data takes

place (e.g.: a new data is provided by the IH or, on the other hand, the operator is performing a What-If scenario by introducing forecasted data).

4. Simultaneous requests: > 100

Model has been strictly tested by performing several simultaneous requests. No degradation of performances has been identified for 100 or less concurrent requests.

5. Percentage of modularity: 100%

The model is 100% modular as we can execute the model, reset it to current status (according to data hosted by the IH), manually change each parameter (e.g.: parking lot availability, incoming vessels and freight type, etc.)

6. Percentage of reusable assets: 100%

The model is based on data commonly available to each port (e.g.: list of incoming vessels, traffic of truck required for loading and unloading operation, parking lot availability, pollution produced by different transportation modes).

3.3.3. Problems Faced and Recommendations

During execution of T4.3 several problems have been faced in order to improve the effectiveness of proposed models and, more specifically, to make them cope with pilot's expectations. In particular, the following issues could impact on the effectiveness of proposed models:

- Weather forecasting: weather (in particular rain) can impact on the results of the proposed models in many different ways:
    - Several bulk goods cannot be unloaded while raining (e.g.: salt, kaolin, urea, cellulose);
    - Rain impacts on emission of pollutants, by altering the results expected by the model in terms of different intermodal transportation routes;
- Traffic jam or accident involving regional highways leading to the entrance of the port: proposed model considers such event, but it requires operator to explicitly declare when such events take place.

From the point of view of technical issues, no significant issues have been identified, in particularly in terms of computational complexity.

3.4. Environmental Pollution Models

One of the main goals of the Environmental Pollution models was the development and application of a dispersion model for the use cases of Port of Piraeus and Port of Thessaloniki. By using weather data and source information the model simulates dispersion of a pollutant in the ambient air. Such simulations can assist the port manager/operator in the decision-making process in order to optimize various activities within the port and minimize their impact on the environment.

Noise dispersion models (noise maps) are used for different purposes, but the main goal is to assess the noise levels in some area. Those noise levels can reflect the current situation or some potential future scenario. In the second case, the model can be used for planning, in this case, port development or noise reduction methods (such as building of noise barriers or optimizing the activities in order to minimize the noise levels).

Another use of the model is that it can help to choose the places where to put noise sensors and/or where to do noise measurements. The model can show where are the noise levels closest (or even above) the regulated values and the measurements should be done in those areas.

3.4.1. Assessment scenario – Air Pollution Model

The air dispersion model for the use cases was done with the American Meteorological Society (AMS) and the United States Environmental Protection Agency (EPA) Regulatory Model – AERMOD. It is an open source steady-state Gaussian plume air dispersion model which predicts downwind pollutant contestations based on source emissions, site parameters (terrain features, land use, etc.), meteorological fields, building locations and more. The AERMOD modelling systems includes five pre-processors, one dispersion model and one post processors. The pre-processors are:

1. AERMAP which is a terrain data pre-processor that processes custom on-site or commercially available digital elevation data
2. AERMET, which is a meteorological data pre-processor that that processes commercially available meteorological data
3. AERSURFACE
4. AERMINUTE
5. BPIPPRM

The main dispersion model is AERMOD and the post processors is AERPLOT.

The chosen approach for incorporating the AERMOD Modelling System in the PIXEL solution is to write python scripts which prepare the input files for the models. To achieve this the scripts, fill in predefined input files with values and information directly from the end user or in an automated fashion

The resulting scripts generate input files for the AEMET, AERMAP and AERMOD models. The other components and models from the AERMOD Modelling System were discarded for several reasons. Namely to make the it more user friendly and because of limited data availability. Also, the post processing will most likely be done without the help of the AERPLOT post processor.

3.4.2. KPI Data Collection and Results – Air Pollution Model

Following KPIs have been selected, according to previously described models, in order to represent assess the effectiveness of the air pollution model proposed in T4.4:

1. Straightforward task accomplishment: Partly.

Proposed model by using weather and source data can simulated the dispersion of a certain pollutant in a certain time period. These models and their scripts can be used in any ports without any adjustments be used in any port.

2. Portion of completed requirements: Relevant: 75%, All: 42%

KPI is evaluated by considering all common functional requirements defined in D3.2. Readers must keep in mind that some KPI are directly linked with integration of the models in the Operational Tools (WP6) and test

and validation in real conditions (WP7). This work is still an on-going work in WP6 and WP7 and will be completed.

Table 7: List of completed requirements for the Environmental Pollution Model

Common functional Requirements	Priority (as define in PIXEL requirements)	Relevant (defined by task leader)	Completed	Comments
Import historical Data (36)	M	1	0	It is planned to integrate with PIXEL information hub to retrieve historical data
Interaction with models (41)	M	1	0	Data provided by the port activity scenario and energy models can be used for emissions calculations which are need input for the air pollution
Anomaly and event list (44)	M	0	0	
Anomaly and event detection (45)	C	0	0	User preference is not yet fully integrated
Homogenize Data (61)	M	1	1	Weather data was transformed by ORANGE
Catalogue of models (62)	M	1	1	AERMOD models were used and a description of each one is provided
Detection of anomalies (63)	M	0	1	Possible in the future after more model testing and comments from ports
Feedback (64)	S	0	0	
Centralized user administration system (65)	M	0	0	This hasn't to be dealt in the model-level.
Configurable Dashboard (66)	M	1	1	We provide visualizations examples (AERPLOT) that can be integrated in the UI.
UI Notification System (67)	M	0	0	This has not to be dealt in the model-level.
Port Operational KPI list (70)	M	0	0	This has not to be dealt in the model-level.
Operation Interface (71)	M	0	0	This has not to be dealt in the model-level.
Analyse historical data (81)	M	1	1	Historical data can be analysed and used in what-if scenario

Support for manually provided data (86)	M	1	1	The scripts allow the user to manually input data
Discovery service for data (104)	M	0	0	This has not to be dealt in the model-level.
Visualization of data (105)	M	1	1	We provide visualizations examples (AERPLOT) that can be integrated in the UI.

The current version of the AERMOD model was developed within the Microsoft Windows operating system (Windows) and has been designed to run on Windows PCs within a Command-prompt using command-line arguments to initiate a model run. The amount of storage space required on the hard disk for a application will depend greatly on the output options selected. Some of the optional output files of concentration data can be rather large.

3. Mean CPU utilization: 18.03 %.

The mean CPU utilisation was monitored and measured with the psutil python package. The model was tested on a 6 core i5-8500 with 8,00 GB RAM installed.

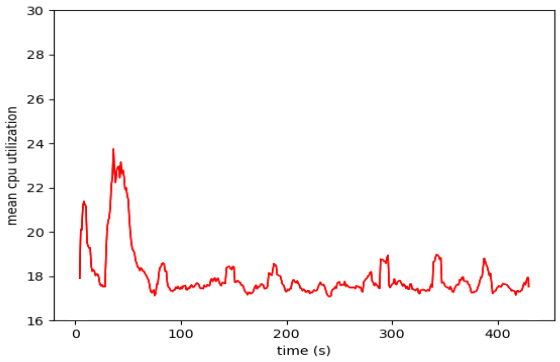


Figure 9: CPU utilization over time for Environmental Pollution Model

4. Mean memory usage: 2.1 MB

The mean memory usage was measured with the Process Explorer tool.

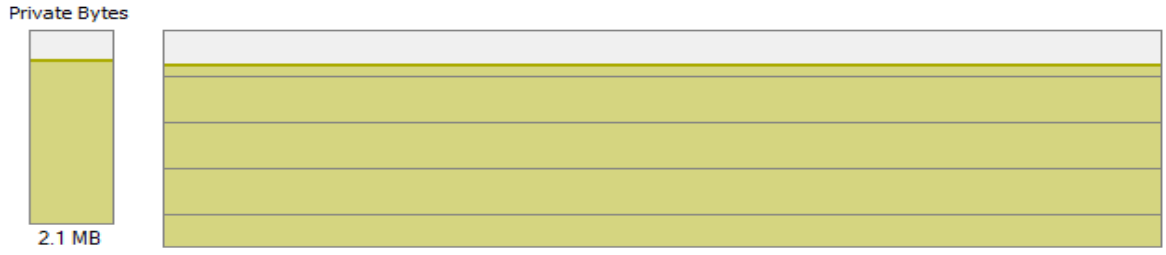


Figure 10: Memory utilization for Environmental Pollution Model

5. Maximum memory usage and maximum processing power, simultaneous requests:

As mentioned before the memory usage and CPU utilisation heavily depends on the model inputs and output functions. Basic model runs can be done with an average laptop. The same can be said for simultaneous calculations.

8. Percentage of modularity:

The AERMAP and AERMET models can be used on their own but the AERMOD model needs the input which are generated with the previous two models

9. Percentage of reusable assets:

The scripts and models are not location or port dependent and can be used u any port desired.

Table 8: Summary of KPI for Air Pollution Model

KPI	Result
Straightforward task accomplishment	Partly
Portion of completed requirements	Relevant: 75%, All: 42%
Mean CPU utilisation	18,03%
Mean memory usage	2,1 MB
Maximum memory usage	Depending on each case
Maximum processing power used	Depending on each case
Simultaneous requests	Depending on how the case calculations are demanding
Percentage of modularity	66%
Percentage of reusable assets	100%

3.4.3. Problems Faced and Recommendations – Air Pollution Model

As mentioned before the amount of required storage space depends on the input parameters and output options. As for this model no real data could be provided at this point, it was tested with a made-up just to test if the model works.

3.4.4. Assessment scenario – Noise Pollution Model

The model assessment was done by running the model for the Port of Thessaloniki. All the relevant noise sources (traffic and port activities) and noise barriers, such as buildings and tanks, were introduced. Noise map was created, and the results were compared to previous similar models done for the port.

For the successfully completion of the simulation, laptop with the following characteristics was used:

- Operating system: Microsoft Windows 10 Pro, 64-bit
- Processor: Intel Core i5-8250U CPU @1.60 GHz 1.80GHz, x64
- 8 GB RAM

3.4.5. KPI Data Collection and Results – Noise Pollution Model

1. Straightforward task accomplishment: Yes. The model was created for the Port of Thessaloniki and can be used to assess the noise levels and influence of different meteorological conditions.

2. Mean CPU utilisation: 24-25%

This KPI was assessed using the Process Explorer software. Although it doesn't have provide the user with an exact number, it is possible to assess the value by checking the graphical representation, such as the one of figure 11, where it can be seen that CPU usage is almost constant during the whole process (the value on the left represents the CPU usage in the moment when the screenshot was taken).

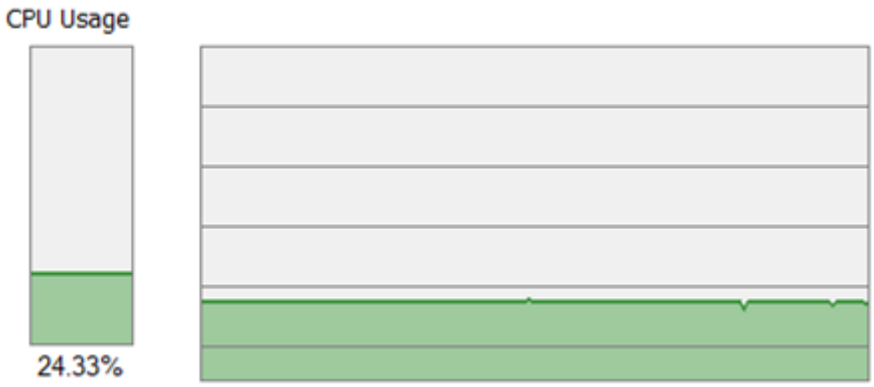


Figure 11: CPU usage for the noise model

3. Mean memory usage: 60MB

Same as with the mean CPU utilisation, the Process Explorer doesn't provide the user with the exact number of mean memory usage and it has to be assessed by looking at graphical representation. Despite the values are much more varied than for the mean CPU utilisation, it was observed that the values are varying between 50MB and 70MB, such as in the figure 12.

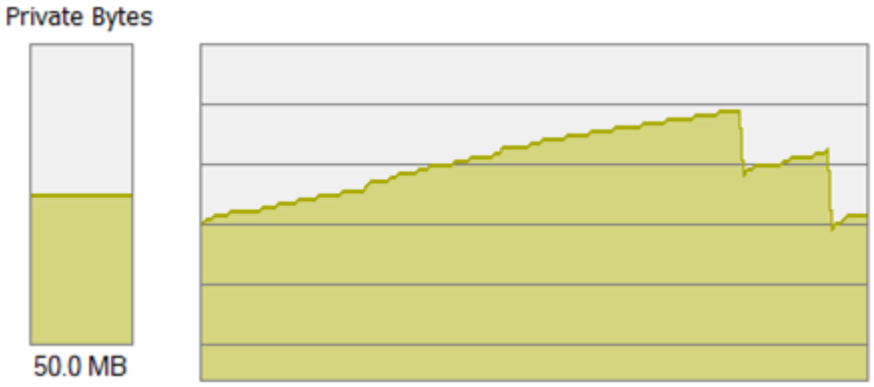


Figure 12: Memory usage for the noise model

4. Maximum memory usage: 85.9MB

Unlike for the previous two KPIs, this one was directly provided by the Process Explorer.

5. Maximum processing power used: 25%

As stated before, and shown on figure 11, the processor usage is pretty consistent and never went over 25%.

Table 9: KPI for Noise model

KPI	Result
Straightforward task accomplishment	Yes
Mean CPU utilisation	24-25%
Mean memory usage	60MB
Maximum memory usage	85.9MB
Maximum processing power used	25%

3.4.6. Problems Faced and Recommendations - Noise Pollution Model

There were no significant problems faced from the technical side of the noise model. The calculation times are reasonable (shown in table 10) and the model can be easily run on an average laptop.

Table 10: Time calculation of Noise Model

Calculation points	Height points (yes/no)	Time
10 (receivers only)	No	Few seconds
2220 (receivers and grid – 50m)	No	21 minutes
8918 (receivers and grid – 25m)	No	85 minutes
10 (receivers only)	Yes	1.5 – 2 minutes
2220 (receivers and grid – 50m)	Yes	3 hours

3.5. PIXEL Data Acquisition

3.5.1. Assessment scenario

KPIs assigned to the assessment (PIXEL Data Acquisition assessment) have been described in D8.1, while the tools and methods for their collection has been defined in D6.3. In this section this methodology is further elaborated, and the assessment scenario is described in detail.

KPIs are estimated either by *expert judgement* or by the development of tools for *automated measurements*.

Expert judgement is performed using desk research, where an expert evaluates the KPI using the approach defined in D8.1/D6.3. **Functional suitability** and **Maintainability** will be estimated using this approach.

Automated measurements are performed either by usage of existing evaluation software or by development of custom tools for this purpose.

Part of the KPIs will this be collected using **JMeter measurements.** The Apache JMeter™ application is an open-source software designed to load test functional behaviour and measure performance. **Performance efficiency** and **Reliability** have been measured using this approach.

In order to assess the performance in the port area, measurements will be performed with a predefined set of realistic input data relevant to port operations. In the beginning, all measurements will be performed under laboratory conditions, and on the infrastructure, which will be defined in WP7 (cloud environment v.s. on-premises installation and other parameters).

Custom modules: reliability and **portability** are going to be measured used custom modules.

Reliability, portability and few other KPIs depend on the deployment of the modules in an operational scenario in order to measure them, as they are mostly statics related to an operational environment.

Table 11: KPI for data acquisition layer KPI Data Collection and Results

KPI	Measurement method	Reporting
Functional suitability		
Straightforward task accomplishment	Expert judgement	D8.2, D8.3
The portion of completed requirements	Expert judgement	D8.2, D8.3
Performance efficiency		
Maximum number of connected data sources	JMeter	D8.3
Maximum database size	(JMeter)	D8.3
Average latency	JMeter	D8.3
Throughput	JMeter	D8.3
Mean CPU Utilisation	JMeter	D8.3
Mean memory usage	JMeter	D8.3
Maximum memory usage	JMeter	D8.3
Maximum processing power used	JMeter	D8.3
Compatibility		

% of APIs coverage	Expert judgement	D8.3
Ability to acquire data from different data formats	Expert judgement	D8.3
Ability to support different IoT platforms	Expert judgement	D8.3
Ability to export different data formats	Expert judgement	D8.3
Reliability		
Simultaneous requests	JMeter	D8.3
% Monthly availability	Custom module, Phase 2 based on Orion API	D8.3
Success rate	Custom module, Phase 2 based on Orion API	D8.3
Maintainability		
% of modularity	Expert judgement	D8.2
% of reusable assets	Expert judgement	D8.2
% of update	Expert judgement, Phase 2	D8.3
Level of analysability	Expert judgement	D8.2
Portability		
Mean number of errors per hardware or OS change/ upgrade	Custom module, Phase 2	D8.3
Mean number of errors per software change/ update	Custom module, Phase 2	D8.3
Mean number of errors per software install	Custom module, Phase 2	D8.3
Mean number of errors per software uninstall	Custom module, Phase 2	D8.3

3.5.2. KPI Data Collection and Results

3.5.2.1. Expert judgement method

Expert judgement has been used for those KPIs that are either too complicated to automate and an expert approach is more efficient, or where a more qualitative evaluation approach is needed. In the following section we report the assessment procedure and the result of the expert judgement.

Functional suitability

Straightforward task accomplishment: "*A process to add a new data sources will be analysed to verify that the process does not include unnecessary steps. "*

Connecting a new Data Source to the Data Acquisition Layer is a manual operation. Its complexity depends of the data source itself and its exposition mechanism. An NGSI agent has to be developed in order to connect the Data Source to the Data Acquisition Layer in order to access the data and convert them to the right format through the chosen security protocol. A generic framework is provided in order to develop quickly those agents. Once developed, the agent is packaged using Docker and deployed in the PIXEL infrastructure. When the agent is deployed, it starts to collect the data and there are immediately available for PIXEL Information Hub. **Overall, for the listed functionalities the value is YES but the process to deploy the agent could be improved.**

The portion of completed requirements: *"Should have" and "Must have" requirements from deliverable D3.2 will be taken as input in order to extract all requirements specifically targeting T6.2.*

Table 12 lists all PIXEL requirements related to the Data Acquisition Layer that have the priority set to *"Should have"* or *"Must have"*. It also lists other PIXEL software modules related to the requirements and the status of development in the DAL. The status does not assess the fulfilment of the requirement in other modules.

Table 12: PIXEL DAL Requirements ("Should have" and "Must have") and implementation status

Requirement	Addressed in additional modules	Implemented in DAL
Common functional requirements		
Homogenize Data [61] *Status: This is the purpose of the NGSI agents. They import the data and transform them using common Data Model before pushing them to IH.*	IH	yes
Support for manually provided data [86] *Status: NGSI Agent provides several ways to import data through the DAL, one of them is pushing CSV or Json files. But it is also possible to handle HTML forms requests.*	IH	yes
Port of Bordeaux – Energy Management Use Case		
Support electricity consumption sensors [9] *Status: The definition of the NGSI agent is in progress with GPMB*	IH	In progress
Monitor expected port calls [11] *Status: The NGSI agents is developed and deployed*	IH	yes
Collect sensor data through Port Community System (VIGIEsip) [12] *Status: DAL provides solution to access VIGIEsip. The data to import have to be identified*	IH	yes*
Support Air Quality Sensors [14] *Status: The definition of the NGSI agent is in progress with GPMB*	IH	In progress
Support wind speed sensors [16] *Status: The definition of the NGSI agent is in progress with GPMB*	IH	In progress
Support weather sensor/service [17] *Status: The definition of the NGSI agent is in progress with GPMB*	IH	In progress

Support old sensors (gauge stations network) [18] *Status: GPMB must provide an API to access those data*	Not defined yet	no
Monitoring l'Ostrea dredge environmental impact [20] *Status: No data API identified yet*	IH, PEI	no
Monitor energy consumption of the port authority [22] *Status: DAL provide Python Framework to develop NGSI Agent*	PA	yes*
Port of Monfalcone – SDAG – Intermodal Transport Use Case		
Integration with the SILI Information System [23] *Status: DAL provides Python Framework to develop NGSI Agent*	IH	yes*
Integration with the PMIS Information System [24] *Status: DAL provides Python Framework to develop NGSI Agent.*	IH	yes*
Integration with ASPM video monitoring system [25] *Status: DAL provides Python Framework to develop NGSI Agent.*	IH	yes*
Integration with the SDAG Access Control System [27] *Status: DAL provides Python Framework to develop NGSI Agent.*	IH	yes*
Integration with data provided by sensors, cameras and feeds by third parties [28] *Status: DAL provides Python Framework to develop NGSI Agent.*	IH	yes*
Port of Thessaloniki – Port City Integration Use Case		
Support wind and weather sensors [47] *Status: DAL provides Python Framework to develop NGSI Agent*	IH	yes*
Support air quality sensors [48] *Status: DAL provides Python Framework to develop NGSI Agent*	IH	yes*
Support water quality sensors and data [49] *Status: DAL provides Python Framework to develop NGSI Agent*	IH	yes*
Support noise sensors and data [50] *Status: DAL provides Python Framework to develop NGSI Agent*	IH	yes*
Support real-time fuel consumption sensors [51] *Status: DAL provides Python Framework to develop NGSI Agent*	IH	yes*
Support real-time gate surveillance sensors [52] *Status: DAL provides Python Framework to develop NGSI Agent*	IH	yes*
Support wind and weather data provided by third party [53] *Status: DAL provides Python Framework to develop NGSI Agent*	IH	yes*
Support air quality data provided by third party [54] *Status: DAL provides Python Framework to develop NGSI Agent*	IH	yes*
Support traffic data provided by third party [55] *Status: DAL provides Python Framework to develop NGSI Agent*	IH	yes*
Port of Pireaus – Port City Integration Use Case		

Support air quality sensors [73] *Status: DAL provides Python Framework to develop NGSI Agent*	Not defined yet	yes*
Support water quality data [75] *Status: DAL provides Python Framework to develop NGSI Agent*	Not defined yet	yes*
Integration with the PMIS SPARC N4 [76] *Status: DAL provides Python Framework to develop NGSI Agent*	IH	In progress
Support noise sensors and data [87] *Status: DAL provides Python Framework to develop NGSI Agent*	Not defined yet	yes*
Support pollution and traffic data provided by third party [88] *Status: DAL provides Python Framework to develop NGSI Agent*	IH	yes*

Legend:

- yes: common functional requirements that are implemented in the Data Acquisition Layer
- yes*: Data Acquisition Layer provide the mechanism needed to implement the requirement. But the development of the required NGSI Agent as to be done together with WP7.
- In progress: The development of the NGSI agent is in progress with the port.
- no: The functionality is not yet available.

Result: A total of 30 requirements are related to functionality provided by the Data Acquisition Layer. Out of those, 3 are fully available, 5 in progress and for 20 of them DAL provide the mechanism to implements them

- Total requirements: 30
- Fulfilled requirements (functionality available): 28
- Portion of completed requirements: 93%.

This KPI has to be re-checked in D8.3 in order to verify that the provided functionality fulfils the acceptance criteria in specific pilot executions.

Maintainability

% of modularity: *Will be measured by reporting all the independent components that are part of the data acquisition module and comparing them to the number of all components in the data acquisition module. Individual operation means that a component can offer a complete function with meaningful information in the context of PIXEL.*

As defined in WP6 deliverables, PIXEL Data Acquisition Layer is composed of several components that provide access to each data sources and a central component that allow Information Hub to discover and access those data.

PIXEL DAL modules are independent of each other, each NGSI Agent communicate with the Data Source with appropriate protocol and push the data through Orion using NGSI v2 REST API. The data is provided to IH through the NGSIv2 REST API.

Result: modularity is 100%.

% of reusable assets: *Will be measured by reporting all the reusable components that are part of the data acquisition layer module and comparing them to the number of all components in the DAL. A reusable*

component is considered any that can be applied in a different context of PIXEL with no modifications of the source code.

All modules in the PIXEL Data Acquisition Layer are FIWARE Generic Enablers or NGSI Agent that could be reuse on any FIWARE compatible project in order to acquire the corresponding Data Source.

Result: 100%.

Level of analysability: *Will be measured by reporting the ratio between the numbers of items inside the Data Acquisition Layer for which logging is implemented compared to the number of items for which the specifications require logging.*

All PIXEL DAL components provide logging capabilities.

Result: 100%.

Summary of results are provided in table 13.

Table 13: Summary of expert judgement results

KPI	Measurement approach
Functional suitability	
Straightforward task accomplishment	yes
The portion of completed requirements	93%, conditional on the implementation of pilots in WP7.
Maintainability	
% of modularity	100%
% of reusable assets	100%
Level of analysability	100%

3.5.2.2. Automated data collection and results

PIXEL Data Acquisition Layer relies on to main parts:

- FIWARE Generic Enabler: ORION
- NGSI Agents

The FIWARE foundation provide Performance testing result and script for ORION https://github.com/telefonicaid/fiware-orion/tree/master/test/loadTest
For NGSI Agents, most of them are still in development and performance efficiently is not yet implementable. All those tests result will be available after deployment of the pilot's platforms in WP7.

3.6. PIXEL Information Hub

3.6.1. Assessment scenario

KPIs assigned to the assessment (PIXEL Information Hub assessment) have been described in D8.1, while the tools and methods for their collection has been defined in D6.3. In this section this methodology is further elaborated, and the assessment scenario is described in detail.

KPIs are estimated either by *expert judgement* or by the development of tools for *automated measurements*.

Expert judgement is performed using desk research, where an expert evaluates the KPI using the approach defined in D8.1/D6.3. **Functional suitability** and **Maintainability** will be estimated using this approach.

Automated measurements are performed either by usage of existing evaluation software or by development of custom tools for this purpose.

Part of the KPIs will this be collected using **JMeter measurements.** The Apache JMeter™ application is an open-source software designed to load test functional behaviour and measure performance. **Performance efficiency** and **Reliability** have been measured using this approach.

In order to assess the performance in the port area, measurements will be performed with a predefined set of realistic input data relevant to port operations. In the beginning, all measurements will be performed under laboratory conditions, and on the infrastructure, which will be defined in WP7 (cloud environment v.s. on-premises installation and other parameters).

Custom modules: reliability and **portability** are going to be measured used custom modules.

Reliability, portability and few other KPIs depend on the deployment of the modules in an operational scenario in order to measure them, as they are mostly statics related to an operational environment. These will be reported in ***D8.3 Technical Evaluation v2*** based on deployments resulting from WP7. For the same reason Functional suitability will be reported in both deliverables, as it relates to both test and operational environments as well as on-gong WP6 developments will result in better coverage in D8.3.

Table 14 provides a list of Information Hub KPIs with measurement methods and deliverables where they will be reported (D8.2, D8.3). It also refers to the evaluation scenario for JMeter measurements, how will the KPI be measured:

- during the data collection phase
- during data extraction
- (A,B) in both scenarios.

Table 14: PIXEL IH KPI list with measurement methods and reporting deliverables

KPI	Measurement method	Reporting
Functional suitability		
Straightforward task accomplishment	Expert judgement	D8.2, D8.3
The portion of completed requirements	Expert judgement	D8.2, D8.3
Performance efficiency		
Maximum number of connected data sources	Pseudo-random data generator	A/D8.2
Maximum database size	Measurement	A/D8.2

Average latency	JMeter	B/D8.2
Throughput	JMeter	B/D8.2
Mean CPU Utilisation	JMeter	A, B/D8.2
Mean memory usage	JMeter	A, B/D8.2
Maximum memory usage	JMeter	A, B/D8.2
Maximum processing power used	JMeter	A, B/D8.2
Reliability		
Simultaneous requests	JMeter	B/D8.2
% Monthly availability	Custom module	D8.3
Success rate	Custom module	D8.3
Maintainability		
% of modularity	Expert judgement	D8.2
% of reusable assets	Expert judgement	D8.2
% of update	Expert judgement	D8.3
Level of analysability	Expert judgement	D8.2
Portability		
Mean number of errors per hardware or OS change/ upgrade	Custom module	D8.3
Mean number of errors per software change/ update	Custom module	D8.3
Mean number of errors per software install	Custom module	D8.3
Mean number of errors per software uninstall	Custom module	D8.3

3.6.2. KPI Data Collection and Results

3.6.2.1 Expert judgement method

Expert judgement has been used for those KPIs that are either too complicated to automate and an expert approach is more efficient, or where a more qualitative evaluation approach is needed. In the following section we report the assessment procedure and the result of the expert judgement.

Functional suitability

Straightforward task accomplishment: *A process to add a new data sources and the process to provide data (data extractor) will be analysed to verify that the process does not include unnecessary steps.*

Adding new sources to the PIXEL Information Hub is automatic through a synchronization mechanism developed as part of the integration with DAL. The IH admin user interface provides a sync functionality, where an updated list of DAL data sources can be automatically retrieved. A user can activate or deactivate data collection from those DAL sources. We are concluding that there is full user control over list of avaialble sources and over their activation. For this first part the value of the KPI is **YES**.

The Data Extractor component provides a REST API endpoint for data provision. Several API calls are available to (1) provide a list of data sources, (2) provide a list of available time intervals, (3) provide data by provision of filter query parameters. Users can develop specific REST API clients to extract data available in the IH, thus also for this second part of the KPI the value is **YES**.

Overall, for the listed functionalities the value is YES.

The portion of completed requirements: *"Should have" and "Must have" requirements from deliverable D3.2 will be taken as input in order to extract all requirements specifically targeting T6.3.*

Table 15 lists all PIXEL requirements related to the IH that have the priority set to *"Should have"* or *"Must have"*. It also lists other PIXEL software modules related to the requirements and the status of development in the IH. The status does not assess the fulfilment of the requirement in other modules.

Table 15: PIXEL IH Requirements ("Should have" and "Must have") and implementation status

Requirement	Addressed in additional modules	Implemented in IH
Common functional requirements		
Analyse historical data [81] *Status: data collected through DAL can be stored in the IH and extracted through IH or Elasticsearch REST API.*	OT	yes
Support for manually provided data [86] *Status: data collected through DAL can be stored in the IH and extracted through REST APIs. IH is agnostic in relation to the collection method.*	DAL	yes
Port of Bordeaux – Energy Management Use Case		
Access to traffic data [10] *Status: DAL-IH connection has been implemented. All data collected through DAL is available in the IH.*	DAL	yes*
Collect sensor data through Port Community System (VIGIEsip) [12] *Status: DAL-IH connection has been implemented. All data collected through DAL is available in the IH.*	DAL	yes*
Support Air Quality Sensors [14] *Status: DAL-IH connection has been implemented. All data collected through DAL is available in the IH.*	DAL	yes*
Support wind speed sensors [16] *Status: DAL-IH connection has been implemented. All data collected through DAL is available in the IH.*	DAL	yes*
Support weather sensor/service [17]	DAL	yes*

Status: DAL-IH connection has been implemented. All data collected through DAL is available in the IH.		
Monitoring l'Ostrea dredge environmental impact [20] *Status: Details of the integration of the PEI module in the overall information architecture has not yet been specified.*	DAL, PEI	no
Expose data to VIGIEsip system [82] *Status: all data in the IH is available either through IH or Elasticsearch REST API.*		yes*
Port of Monfalcone – SDAG – Intermodal Transport Use Case		
Integration with the SILI Information System [23] *Status: DAL-IH connection has been implemented. All data collected through DAL is available in the IH.*	DAL	yes*
Integration with the PMIS Information System [24] *Status: DAL-IH connection has been implemented. All data collected through DAL is available in the IH.*	DAL	yes*
Integration with ASPM video monitoring system [25] *Status: DAL-IH connection has been implemented. All data collected through DAL is available in the IH.*	DAL	yes*
Integration with the SDAG Access Control System [27] *Status: DAL-IH connection has been implemented. All data collected through DAL is available in the IH.*	DAL	yes*
Port of Thessaloniki – Port City Integration Use Case		
Support noise sensors and data [50] *Status: DAL-IH connection has been implemented. All data collected through DAL is available in the IH.*	DAL	yes*
Support real-time fuel consumption sensors [51] *Status: DAL-IH connection has been implemented. All data collected through DAL is available in the IH.*	DAL	yes*
Support real-time gate surveillance sensors [52] *Status: DAL-IH connection has been implemented. All data collected through DAL is available in the IH.*	DAL	yes*
Support wind and weather data provided by third party [53] *Status: DAL-IH connection has been implemented. All data collected through DAL is available in the IH.*	DAL	yes*
Support air quality data provided by third party [54] *Status: DAL-IH connection has been implemented. All data collected through DAL is available in the IH.*	DAL	yes*
Support traffic data provided by third party [55] *Status: DAL-IH connection has been implemented. All data collected through DAL is available in the IH.*	DAL	yes*
Port of Piraeus – Port City Integration Use Case		
Integration with the PMIS SPARC N4 [76]	DAL	yes*

Status: DAL-IH connection has been implemented. All data collected through DAL is available in the IH.		
Support pollution and traffic data provided by third party [88] *Status: DAL-IH connection has been implemented. All data collected through DAL is available in the IH.*	DAL	yes*

Legend:

- yes: common functional requirements that are implemented in the Information Hub.
- yes*: requirements that are related to data acquisition for specific data sources in different uses cases. The functionality is, in principle, available in the IH as it relates to a common functional requirement for data availability. However, the functionality needs a final evaluation after the execution of pilots in WP7.
- no: The functionality is not yet available.

Result: At total of 21 requirements are related to functionality provided by the IH. Out of those, 20 are available in the IH, but the integration has not been evaluated for most of them as they are use-case specific.

- Total requirements: 21
- Fulfilled requirements (functionality available): 20
- Portion of completed requirements: 95%.

This KPI has to be re-checked in D8.3 in order to verify that the provided functionality fulfils the acceptance criteria in specific pilot executions.

Maintainability

% of modularity: *Will be measured by reporting all the independent components that are part of the information hub module and comparing them to the number of all components in the information hub module. Individual operation means that a component can offer a complete function with meaningful information in the context of PIXEL.*

As defined in WP6 deliverables, PIXEL Information Hub is composed of several components that provide high-performance processing, storage and provision of data collected through PIXEL. It acts as a central storage for all PIXEL operations.

PIXEL IH modules are independent of each other, as there are designed to communicate through specific communication standards: REST API, Kafka message protocol and Zookeeper communication protocol. In principle, each module could be replaced, as it provides a well-defined set of isolated functions and is decoupled from other parts of the system.

Result: modularity is 100%.

% of reusable assets: *Will be measured by reporting all the reusable components that are part of the information hub module and comparing them to the number of all components in the IH. A reusable component is considered any that can be applied in a different context of PIXEL with no modifications of the source code.*

All modules in the PIXEL Information Hub are context-agnostic in the sense that the IH can be applied in any PIXEL pilot, use-case or, more general, to any port scenario. This is achieved by a generic approach to data sources and structures definition. Data sources, their type and structure are obtained automatically form DAL. Furthermore, all data is available through a generic REST API to client modules (Dashboard, OT).

Result: 100% of assets can be reused in different port scenarios.

Level of analysability: *Will be measured by reporting the ratio between the numbers of items inside the information hub for which logging is implemented compared to the number of items for which the specifications require logging.*

All PIXEL IH components provide logging capabilities.

Result: 100%. Summary of results are provided in table 16.

Table 16: Summary of expert judgement results

KPI	Measurement approach
Functional suitability	
Straightforward task accomplishment	yes
The portion of completed requirements	95%, conditional on the implementation of pilots in WP7.
Maintainability	
% of modularity	100%
% of reusable assets	100%
Level of analysability	100%

3.6.2.2. Automated data collection and results (JMeter and custom tools)

For the purpose of data collection, a setup with two workstations has been used:

- PIXEL DAL and IH deployment is a workstation with:
  - Docker
  - PIXEL Information Hub
  - DAL (FIWARE Orion): used for the Data Collection KPI eval scenario
  - JMeter PerfMon
- Testing workstation: a workstation with installed JMeter probing and reporting tools.

The deployment workstation has the following specifications:

- CPU: Intel(R) Core (TM) i7-6700 CPU @ 3.40GHz
- RAM: 32 GB RAM (2x 16 GB DIMM DDR4 Synchronous 2400 MHz (0.4 ns))
- GPU: GeForce GTX 1050 Ti
- SDD: 500 GB
- OS: Ubuntu 18.10 (64-bit)

As explained in section 3.6.1, two distinct evaluation scenarios have been deployed and executed:

(A) Data collection KPI evaluation

For testing this scenario, a vessel calls pseudo-random data generator has been developed. The generator has the following input parameters:

- number of concurrent generation threads (which equals to number of data sources)
- Frequency of updates in messages/s
- Duration of testing interval and pause between tests
- The message size has been set to 1.5 Kb.

For all tests the duration has been set to three minutes. We execute a series of tests with combination of number of concurrent clients and generation frequency as follows:

- the number of sources is set to: 10 and 100.
- the generation frequency for one source is set to 1 and 4 req/s, amounting to the most demanding workload of 400 req/s.

In total we perform four tests. In this scenario the PerfMon tool is used on the server, while the workload is generated by the custom-developed tool, rather than JMeter probes.

It is important to note, that this scenario data is inserted through DAL, meaning that the reported performance is for the combination of both PIXEL components: DAL and IH.

Table 17 shows the executed test with the following information: number of clients and number of requests performed by each client per second, total expected frequency (clients * req/client) in req/se, total expected number of requests (frq * three minutes) and the actual achieved number of requests during the test execution.

Table 17: PIXEL IH KPI data generation evaluation setup (scenario A)

Test ID	Test setup				Test execution - achieved performance
	clients	req/s/client	Total req/s	Requests (3 min)	Requests (3 min)
1	10	1	10	1,800	1790
2	10	4	40	7,200	7160
3	100	1	100	18,000	18000
4	100	4	400	72,000	71600

Memory CPU and memory utilisation is provided in the table below. All values are provided in %.

Table 18: PIXEL IH KPI data generation evaluation setup (scenario A)

Test ID	CPU mean	CPU max	Memory mean	Memory max
1	12.26	18.65	50.55	50.62
2	20.68	32.20	50.68	50.79
3	36.97	65.21	50.93	51.13
4	66.35	94.04	51.46	51.72

We can observe that the memory usage is constant, around 50%, while CPU usage increases with the increase of messaging frequency. The same figures can be observed in the graph below:

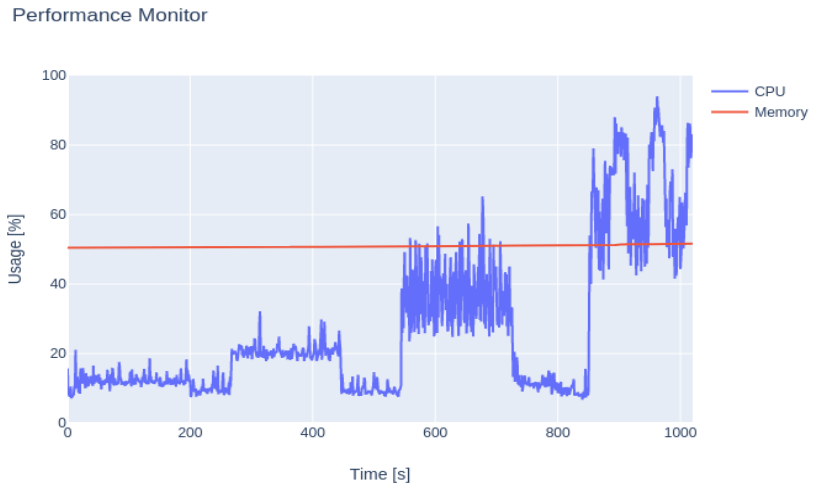


Figure 13: CPU and Memory usage of the Information Hub (scenario A)

We tested the system for up to 100 connected data sources with increasing update frequency. It seems that rather than the number of sources, the limiting factor is the update frequency that is reaching the limit, for the evaluation setup, at around 400 req/s.

However, it has to be noted that for this test we deployed all DAL and IH components on the same workstation, which is an unlikely situation in a production environment, where the deployment would be distributed, thus allowing the execution of processes over multiple servers.

As for the maximum database size KPI, it would be very difficult to measure that value as it depends on a particular deployment, most importantly on the distribution of Elastic search over multiple nodes (servers). As per Elastic search specification, the maximum database size depends on following good practices of index, shards and nodes design.

(B) Data extraction KPI evaluation

For testing this scenario, a set of JMeter HTTP probes has been setup to simulate HTTP REST calls to the PIXEL IH Data Extractor. The JMeter probes have the following input parameters:

- number of concurrent generation threads (which equals to number of connected clients)
- Frequency of requests in messages/s
- Duration of testing interval and pause between tests

For all tests the duration has been set to three minutes with 30 seconds pauses between tests and 30 seconds ramp-up and ramp-down periods. We execute a series of tests with combination of number of concurrent clients and generation frequency as follows:

- the number of clients is set to: 1, 10, 50 and 70.
- the generation frequency for one client is set to 1, 5, 10 req/s.

In total we perform 4x3 = 12 tests. The size of the response was around 40 kb.

In this scenario the PerfMon tool is used on the server, while the workload is generated by JMeter HTTP probes.

Table 19 shows the executed test with the following information: number of clients and number of requests performed by each client per second, total expected frequency (clients * req/client) in req/se, total expected number of requests (frq * three minutes) and the actual achieved number of requests during the test execution.

Table 19: PIXEL IH KPI data generation evaluation setup

	Test setup				Test execution - achieved performance
Test ID	clients	req/s/client	Total req/s	Requests (3 min)	Requests (3 min)
1	1	1	1	180	180
2	10	1	10	1,800	1,806
3	50	1	50	9,000	9,005
4	70	1	70	12,600	12,603
5	1	5	5	900	901
6	10	5	50	9,000	9,000
7	**50**	**5**	**250**	**45,000**	**44,263**
8	**70**	**5**	**350**	**63,000**	**57,962**
9	1	10	10	1,800	1,801
10	10	10	100	18,000	17,903
11	**50**	**10**	**500**	**90,000**	**58,406**
12	**70**	**10**	**700**	**126,000**	**58,136**

This test shows that with the evaluation setup the system can handle the workload of around 300 msg/sec. The workload is more related to the total frequency of requests, rather than on the number of connected clients. Tests 7, 8, 11 and 12 (bold/underline) show this limitation, as the number of processed requests starts lagging behind the test setup.

The table below shows the measured latency, which is, for the acceptance test cases, **in the range of around 20 ms.** For the critical test cases, where the limits of the test setup has been reached, the latency is higher.

Table 20: PIXEL IH KPI latency

Test ID	mean (ms)	standard deviation (ms)	Min (ms)	Max (ms)
1	20	56	12	771
2	27	9	13	70
3	23	8	11	130
4	23	10	10	247
5	10	1	9	32
6	17	9	8	228
7	**22**	**32**	**8**	**853**
8	**203**	**67**	**10**	**1,592**
9	10	1	8	17
10	17	9	8	359
11	**142**	**54**	**10**	**1,891**
12	**205**	**79**	**11**	**1,923**

The same behaviour is seen from the latency graph, below. Critical tests (7, 8, 11, 12) are marked in orange.

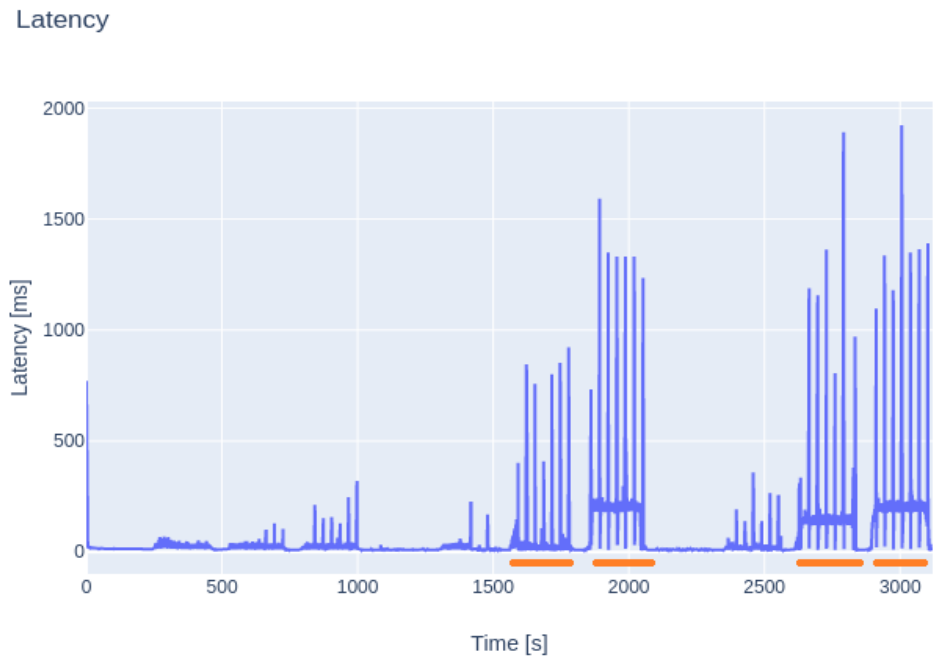


Figure 14: Latency of the Information Hub

Throughput similarly to other KPIs, gets affected once the test setup reaches the limits of operation. In the image below, we can observe that the maximum achievable throughput is around 340 msg/s. In test 11 and 12 we generate more than 350 requests/s the limit is lower.

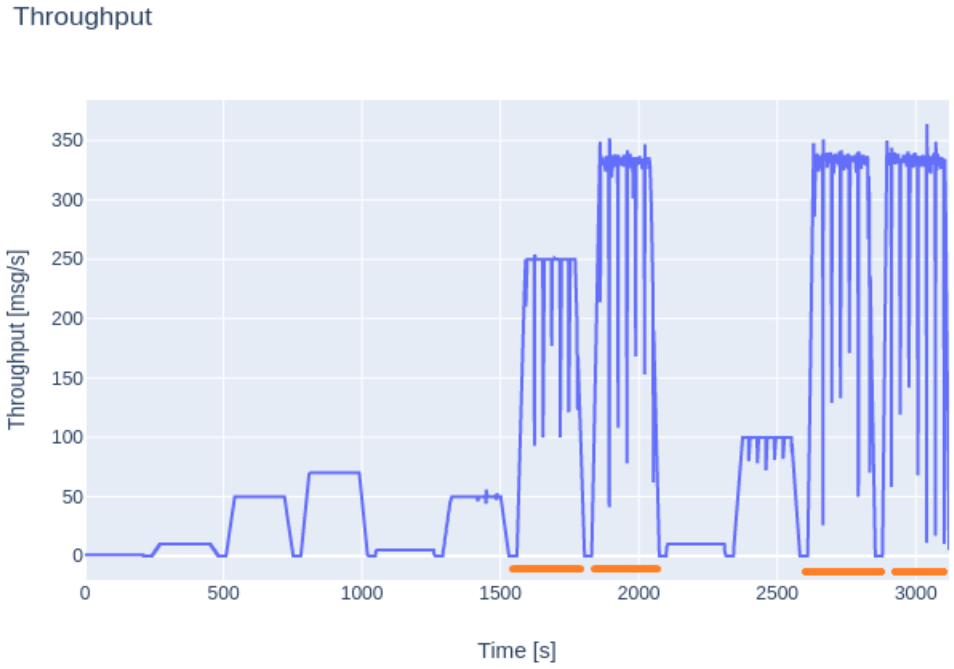


Figure 15: Throughput of the Information Hub

We can assume that the throughput that can be achieved for this evaluation is 340 msg/sec. Memory CPU and memory utilisation is provided in the table below. All values are provided in %.

Table 21: PIXEL IH KPI for CPU and memory

Test ID	CPU mean	CPU max	Memory mean	Memory max
1	14.17	24.00	20.01	20.25
2	17.19	30.92	20.30	20.47
3	25.80	39.46	20.69	20.86
4	30.06	44.60	21.10	21.26
5	14.25	16.99	21.33	21.37
6	25.57	42.72	21.42	21.53
7	**70.92**	**83.52**	**22.27**	**22.65**
8	**86.95**	**90.74**	**23.23**	**23.53**
9	15.50	21.28	23.07	23.11
10	37.34	40.15	23.09	23.43
11	**87.04**	**91.31**	**24.12**	**24.39**
12	**86.78**	**90.59**	**24.35**	**24.62**

We can observe that the memory usage is constant, around 20 % - 25%, while CPU usage increases significantly for the critical test cases 7, 8, 11, 12. We can assume that the mean CPU utilisation in acceptable cases is around 30%. The same figures can be observed in the graph below:

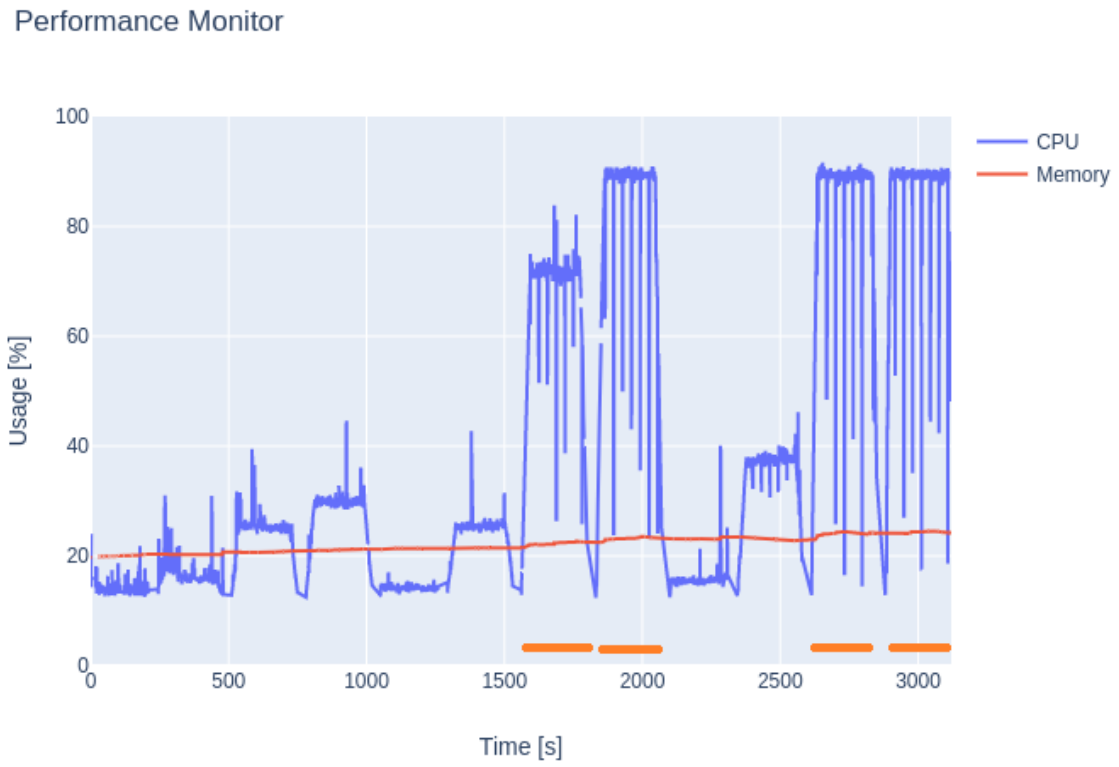


Figure 16: CPU and Memory usage of the Information Hub (scenario B)

We tested the system for up to 70 simultaneous requests, although it seems that this number could be significantly increased, depending on the frequency of requests, which seems to be a limiting factor.

Table 22: Summary of automated data collection results

KPI	Result
Performance efficiency	
Maximum number of connected data sources	A: 100
Maximum database size	A: Not measured - depends on Elasticsearch deployment.
Average latency	B: 20 ms
Throughput	B: 340 msg/s
Mean CPU Utilisation	A: 66% B: 30%
Mean memory usage	A: 51% B: 22%
Maximum memory usage	A: 51% B: 25%
Maximum processing power used	A: 94% B: 42%-86%
Reliability	
Simultaneous requests	B: 70

3.6.3. Problems Faced and Recommendations

The main challenge when evaluating the IH mainly to the centrality of the IH in the PIXEL ICT ecosystem and dependencies among PIXEL modules. IH is a general-purpose module where data should be collected, processed, stored and made available to client applications.

This means that although IH has been evaluated in this deliverable, an additional evaluation is needed in D8.3 where the IH will be evaluated as an integral part of the PIXEL platform applied to the four pilots in WP7.

The previous argument is important for *Functional suitability*, that has been evaluated in this deliverable, but 19 out of 21 requirements are tightly related to the execution of the pilots. Although the IH supports those functionalities, the final evaluation will come from the pilots.

Furthermore, *Reliability* and *Portability* relate to deployments in (near)-production environments, so those evaluations have been postponed to D8.3.

3.7. PIXEL Operational Tools

3.7.1. Assessment scenario

An appropriate picture of the Operational Tools (OT) can be established through:

- Deliverable D6.2, which represents the final update of the architecture describing all modules including the OT.
- Deliverable D8.1, which describes (among others) the main plan for the technical assessment for the different modules, including the OT.
- Deliverable D6.3, which describes the current implementation of the different architecture modules, including the OT.

Technical assessment is performed by means of a defined list of KPIs (see table below) that are evaluated either by *expert judgement* or by *automated tools*. **Functional suitability** and **Maintainability** (see table below) will be estimated using the expert judgement approach, whereas **Performance efficiency** and **Reliability** will follow the automated approach. The expected automated tools are **JMeter** and **PerfMon**, intended to provide performance measurements based on predefined test probes.

In a first phase (this deliverable), all measurements will be performed under laboratory conditions, as the deployment sites are still not ready (lack of input data, finalization of software and services, etc.). This will be solved in the second phase (D8.3), where everything will be ready, even the assessment at WP7 level. This implies that certain KPIs will not be measured in this deliverable (see table below), such as **Portability** and **Reliability**.

Table 23: KPI for Operational Tools

KPI	Measurement method
Functional suitability	
Straightforward task accomplishment	Expert judgement
The portion of completed requirements	Expert judgement
Performance efficiency	
Average latency	JMeter
Mean CPU Utilisation	JMeter, PerfMon

Mean memory usage	JMeter, PerfMon
Maximum memory usage	JMeter, PerfMon
Maximum processing power used	JMeter, PErfMon
Reliability	
Simultaneous requests	JMeter
% Monthly availability	Phase 2 (D8.3)
Success rate	Phase 2 /D8.3)
Maintainability	
% of modularity	Expert judgement
% of reusable assets	Expert judgement
% of update	Phase 2 (D8.3)
Level of analysability	Expert judgement
Portability	
Mean number of errors per hardware or OS change/ upgrade	Phase 2 (D8.3)
Mean number of errors per software change/ update	Phase 2 (D8.3)

Mean number of errors per software install	Phase 2 (D8.3)
Mean number of errors per software uninstall	Phase 2 (D8.3)

3.7.2. KPI Data Collection and Results

3.7.2.1. Data Collection - Expert judgement method

Table 24: Expert judgment method for Operational Tools

KPI	Measurement approach	Measurement method
Functional suitability		
Straightforward task accomplishment	A process to add a new model or predictive algorithm will be analysed to verify that the process does not include unnecessary steps. Boolean response (Yes/No) A process to run/schedule a new model or predictive algorithm will be analysed to verify that the process does not include unnecessary steps. Boolean response (Yes/No) A process to configure events (patterns/anomalies) will be analysed to verify that the process does not include unnecessary steps. Boolean response (Yes/No)	Expert judgement
The portion of completed requirements	Deliverable D3.2 will be taken as input in order to extract all requirements specifically targeting T6.4. So far, specific requirements relate to: • Interaction with models [41] • Interaction with Catalogue [62] • Anomaly and event list [44] • Anomaly and event detection [45] • Detection of anomalies [63]	Expert judgement
Maintainability		

% of modularity	Will be measured by reporting all the independent components that are part of the operational tools module and comparing them to the number of all components in the data acquisition module. Individual operation means that a component can offer a complete function with meaningful information in the context of PIXEL.	Expert judgement
% of reusable assets	Will be measured by reporting all the reusable components that are part of the operational tools module and comparing them to the number of all components in the data acquisition module. A reusable component is considered any that can be applied in a different context of PIXEL with no modifications of the source code.	Expert judgement
% of update	Will be measured by reporting the level of success in software updates on the operational tools module. It compares successfully completed updates versus all executed updates.	Expert judgement Phase 2 (D8.3)
Level of analysability	Will be measured by reporting the ratio between the numbers of items inside the operational tools for which logging is implemented compared to the number of items for which the specifications require logging.	Expert judgement

3.7.2.2. Data Collection using JMeter

Table 25: JMeter method for Operational Tools

KPI	Measurement approach	
Performance efficiency		
Average latency	With JMeter, requests to multiple services (encapsulating models and PAs) will be launched and average response time will be measured. We will differentiate 3 operational ranges: • Low: least estimation of models and PAs • Medium: average estimation of models and PAs • High: worst-case estimation of models and PAs	JMeter
Mean CPU Utilisation	The same approach as for average latency is valid. In order to get the mean CPU usage, PerfMon will be used for the same JMeter tests.	Jmeter, PerfMon

Mean memory usage	The same approach as for average latency is valid. In order to get the mean CPU usage, PerfMon will be used for the same JMeter tests.	Jmeter, PerfMon
Maximum memory usage	The same approach as for average latency is valid. In order to get the mean CPU usage, PerfMon will be used for the same JMeter tests.	Jmeter, PerfMon
Maximum processing power used	The same approach as for average latency is valid. In order to get the mean CPU usage, PerfMon will be used for the same JMeter tests.	Jmeter, PerfMon
Reliability		
Simultaneous requests	The same approach as for average latency is valid and **JMeter** will be used. Here JMeter probes will be defined to increase the number of concurrent requests progressively until the load arrives at a certain threshold.	JMeter

3.7.2.3. Data collection using other automated tools

Table 26: Automated tools for Operational Tools

KPI	Measurement approach	
Reliability		
% Monthly availability	For health status, an availability probe must be defined per each model and PA with minimal impact on performance. A test input might be provided by model/PA. A periodic process will check regularly (e.g. every hour) if a model/PA is available (e.g. sending the test input and getting an expected successful response). Statistics will be collected and be available per month. If there is unavailability from a service (model), it will try to recover automatically, otherwise, a notification (to the administrator) will be sent.	Phase 2 (D8.3) (custom)

Success rate	For each execution of the service (model), the success or failure will be stored, serving as statistics indicator.	Phase 2 (D8.3) (custom)
Portability		
Mean number of errors per hardware or OS change/ upgrade	Will be measured by analysing the system & application error logs.	Phase 2 (D8.3)
Mean number of errors per software change/ update	Will be measured by analysing the system & application error logs.	(custom)
Mean number of errors per software install	Will be measured by analysing the system & application error logs.	Phase 2
Mean number of errors per software uninstall	Will be measured by analysing the system & application error logs.	Phase 2 (D8.3) (custom)

3.7.2.4. Results

Table 27: KPI results for Operational Tools

KPI	Result
Functional suitability	
Straightforward task accomplishment	*Process to add a new model*: **Yes (100%)** The process only requires filling a form with all necessary data. REST API also available *Process to run a new model*: **Yes (75%)**

	Models can be run in real-time and non-real-time mode. The OT is able to handle this but all models need to be integrated to verify it. *Process to schedule a new model*: **Yes (80%).** Models can be scheduled by date (every X minute/hour/day/week/month). Pending is still the triggering by events *Process to run a predictive algorithm*: **No (50%)** Predictive algorithms are still being developed within WP4, a similar process as for models is available, but will require confirmation *Process to schedule a predictive algorithm*: **No (50%)** Predictive algorithms are still being developed within WP4, a similar process as for models is available, but will require confirmation *Process to configure events*: **No (25%)** This functionality still lacks proper access to data, proper definition of relevant events and also integration with the Dashboard which also incorporates a similar module (ElasAlert)

The portion of completed requirements	Following D3.2 requirements for the Operational Tools: *Interaction with models [41]:* **80%** The OT Engine is able to discover available models in the PIXEL platform, executes the involved model(s) and obtain a valid response. However, the complete integration with the IH API is pending. Catalogue of models [62]: **60%** As for the backend functionality there is no major further development; however, models from WP4 still needs to be converted into services to be published/deployed in the PIXEL platform *Anomaly and event list [44]*: **15 %** Due to lack of data availability through the IH, there is still no current list of anomalies to be detected, except for some basic ones (e.g. arrival of anew ship to the port) *Anomaly and event detection [45]:* **30%** Requirement 44 limits the fulfillment of this requirement, even if the engine is available *Detection of anomalies [63]:* **10 %** Similar situation as for requirement 44; we should detect peaks but need the models up and running in ports and set the thresholds for each use case
Performance efficiency (more detailed in Appendix C)	
Average latency	OT-API-Write Operation: **3.1 ms** OT-API-Read Operation: **3.1 ms** OT-Model deployment: **69 s** (PAS), **49 s** (PEI) OT-Execution: **1072 ms** (single), **26028 ms** (multiple)
Mean CPU Utilisation	OT-API-Write Operation: **3.4 %** OT-API-Read Operation: **2.9 %** OT-Model deployment: **6,7 %** (PAS), **7.7 %** (PEI) OT-Execution: **21 %** (single), **25 %** (multiple)

Mean memory usage	OT-API-Write Operation: **13.5 %** OT-API-Read Operation: **13.5 %** OT-Model deployment: **30 %** (PAS), **25.5 %** (PEI) OT-Execution: **11.9 %** (single), **16 %** (multiple)
Maximum memory usage	OT-API-Write Operation: **13.5 %** (no impact) OT-API-Read Operation: **13.5 %** (no impact) OT-Model deployment: **31.6 %** (PAS), **26.5 %** (PEI) OT-Execution: **11.95 %** (single), **19.5 %** (multiple)
Maximum processing power used	OT-API-Write Operation: **11 %** OT-API-Read Operation: **25 %** OT-Model deployment: **31.49 %** (PAS), **35 %** (PEI) OT-Execution: **26 %** (single), **27.2 %** (multiple)
Reliability	
Simultaneous requests	Phase 2 (D8.3)
% Monthly availability	Phase 2 (D8.3)
Success rate	Phase 2 (D8.3)
Maintainability	
% of modularity	**80%** OT-API. Individual component (requires DDBB- Mongo) OT-UI. Requires OT-API

	OT-planner. Individual component (requires DDBB- Mongo) OT- deployer. Individual component (requires DDBB- Mongo) OT- Event processing – Individual component
% of reusable assets	**55%** OT-API. Reusable 25% (ad hoc component, needs adaptation) OT-UI. Reusable 25% (ad hoc component, needs adaptation) OT-planner. Reusable 50% (ad hoc component, but based as scheduler) OT- deployer. Reusable 75% (ad hoc component, but based on scripting) OT-Event processing –ElastAlert. Reusable 100% (open source software)
Level of analysability	**100%** OT-API. Logging supported (Log4j) OT-UI. Logging not supported (not needed) OT-planner. Logging supported (Log4j) OT-deployer. Logging supported (script- append) OT- Event processing –ElastAlert. Logging support (verbose execution)
Portability	
Mean number of errors per hardware or OS change/ upgrade	Phase 2 (D8.3)
Mean number of errors per software change/ update	Phase 2 (D8.3)
Mean number of errors per software install	Phase 2 (D8.3)

Mean number of errors per software uninstall	Phase 2 (D8.3)

3.7.3. Problems Faced and Recommendations

The main problem faced when assessing the Operational Tools (OTs) mainly relate to their dependency on other modules of the architecture, therefore it is not possible to provide real final numbers until the end of the development.

Some test data can be used to evaluate the performance of the functionalities provided by the Operational Tools; however, they can be considered tricky until a final deployment (during the pilots in WP7) is set. Note that the OTs glues things together and the final puzzle needs to be available for a proper evaluation. Considering the time constraints and the evolution of the project at the time of releasing this deliverable, a partial evaluation has been done to provide initial results.

The main problems encountered for a full evaluation of the Operational Tools are:

- Lack of real time data available in the IH to test all different use cases and scenarios. This does not mean that there is no data in the IH, it is the catenation of various current limitations (data at ports, data integrated through the DAL, then to the IH, etc.) that represents a real issue for an end-to-end testing. It is envisioned to be solved in the upcoming weeks during the final integration of modules including real live data from ports.
- Lack of models and predictive algorithms converted into services capable to be published through the operational Tools.

The previous problems are a consequence of the schedule of the project (e.g. models have recently been released as executables in D4.2, but require adaptation, whereas predictive algorithms are still being implemented). Such problems will not persist and do not represent a risk. The next release (and final version) of this deliverable will incorporate a final evaluation of the Operational Tools, as all models and predictive algorithms will have been deployed and tested in all pilot ports.

3.8. PIXEL Integrated Dashboard and Notification

3.8.1. Assessment scenario

KPIs assigned to the assessment (PIXEL Integrated Dashboard and Notifications) were described in D8.1, while the current implementation of the different architecture modules (PIXEL Integrated Dashboard and Notifications) was defined in D6.3.

In this section this methodology is further elaborated, and the assessment scenario is described in detail.

Technical assessment is performed by means of a defined list of KPIs (see table below) that are evaluated either by *expert judgement* or by *automated tools* (using the approach defined in D8.1/D6.3).

Technical assessment is performed by means of a defined list of KPIs (see table below) that are evaluated either by *expert judgement* or by *automated tools*. **Functional suitability** and **Maintainability** (see table below) will be estimated using the expert judgement approach, whereas **Performance efficiency** and **Reliability** will follow the automated approach. The expected automated tool is:

- **Apache JMeter™**. Open source software designed to load test functional behaviour and measure performance.

In order to assess the performance in the port area, measurements will be performed with a predefined set of realistic input data relevant to port operations. In the beginning, all measurements will be performed under laboratory conditions, and on the infrastructure, which will be defined in WP7.

In a first phase (this deliverable), all measurements will be performed under laboratory conditions, as the deployment sites are still not ready (lack of input data, finalization of software and services, etc.). This will be solved in the second phase (D8.3), where everything will be ready, even the assessment at WP7 level. This implies that certain KPIs will not be measured in this deliverable, such as **Portability** and **Reliability**.

Table 28: KPI for Dashboard and Notifications

KPI	Measurement method
Functional suitability	
Straightforward task accomplishment	Expert judgement
The portion of completed requirements	Expert judgement
Performance efficiency	
Mean CPU Utilisation	JMeter
Mean memory usage	JMeter
Maximum memory usage	JMeter
Maximum processing power used	JMeter
Reliability	
Simultaneous requests	JMeter
% Monthly availability	Phase 2 (8.3)
Security	
Incidents of ownership changes and accessing prohibited data	Expert judgement
Incidents of authentication mechanisms breaches	Expert judgement
Level of User authenticity	Expert judgement

Maintainability	
% of modularity	Expert judgement
% of reusable assets	Expert judgement
% of update	Expert judgement, Phase 2
Level of analysability	Expert judgement
Portability	
Mean number of errors per hardware or OS change/ upgrade	Phase 2 (8.3)
Mean number of errors per software change/ update	Phase 2 (8.3)
Mean number of errors per software install	Phase 2 (8.3)
Mean number of errors per software uninstall	Phase 2 (8.3)

3.8.2. KPI Data Collection and Results

3.8.2.1 Data Collection - Expert judgement method

Table 29: Expert judgment method for Dashboard and Notifications

KPI	Measurement approach	Measurement method
Functional suitability		
Straightforward task accomplishment	A process to add / configure widgets will be analysed to verify that the process does not include unnecessary steps. Boolean response (Yes/No) A process to send / receive notifications will be analysed to verify that the process does not include unnecessary steps. Boolean response (Yes/No) A process to create new alerts will be analysed to verify that the process does not include unnecessary steps	Expert judgement Phase 2 (8.3)

	Boolean response (Yes/No)	
The portion of completed requirements	Deliverable D3.2 will be taken as input in order to extract all requirements specifically targeting T6.4.	Expert judgement Phase 2 (8.3)
Maintainability		
% of modularity	Will be measured by reporting all the independent components that are part of the dashboard and notifications module. Individual operation means that a component can offer a complete function with meaningful information in the context of PIXEL.	Expert judgement
% of reusable assets	Will be measured by reporting all the reusable components that are part of the dashboard and notifications module. A reusable component is considered any that can be applied in a different context of PIXEL with no modifications of the source code.	Expert judgement
% of update	Will be measured by reporting the level of success in software updates on the dashboard and notifications module. It compares successfully completed updates versus all executed updates.	Expert judgement Phase 2 (D8.3)
Level of analysability	Will be measured by reporting the ratio between the numbers of items inside the dashboard and notifications for which logging is implemented compared to the number of items for which the specifications require logging.	Expert judgement Phase 2 (8.3)

3.8.2.2. Data Collection using JMeter

Table 30: JMeter method for Dashboard and Notifications

KPI	Measurement approach	
Performance efficiency		
Mean CPU Utilisation	With JMeterto get the mean CPU usage. PerfMon will be used for the same JMeter tests.	JMeter, PerfMon

Mean memory usage	The same approach as for average latency is valid. In order to get the mean memory usage. The PerfMon will be used for the same JMeter tests.	JMeter, PerfMon
Maximum memory usage	The same approach as for average latency is valid. In order to get the maximum memory usage, the **PerfMon JMeter plugin** will be used for the same JMeter tests.	JMeter, PerfMon
Maximum processing power used	The same approach as for average latency is valid. In order to get the maximum CPU usage, the **PerfMon JMeter plugin** will be used for the same JMeter tests.	JMeter, PerfMon
Reliability		
Simultaneous requests	**JMeter** will be used. Here JMeter probes will be defined to increase the number of concurrent requests progressively until the load arrives at a certain threshold.	JMeter

3.8.2.3. Data collection using other automated tools

Table 31: Automated tools for Dashboard and Notifications

KPI	Measurement approach	
Reliability		
% Monthly availability	For health status, an **availability probe** must be defined with minimal impact on performance.	Phase 2 (D8.3)
Portability		
Mean number of errors per hardware or OS change/ upgrade	Will be measured by analysing the system & application error logs.	Phase 2 (D8.3)

Mean number of errors per software change/ update	Will be measured by analysing the system & application error logs.	(custom)
Mean number of errors per software install	Will be measured by analysing the system & application error logs.	Phase 2
Mean number of errors per software uninstall	Will be measured by analysing the system & application error logs.	Phase 2 (D8.3)

3.8.2.4 Results

Table 32: Results for Dashboard and Notifications

KPI	Result
Functional suitability	
Straightforward task accomplishment	*Process to add / configure widgets*: **Yes (80%)** The process only requires filling a form with all necessary data. REST API also available *Process to send / receive notifications*: **No (25%)** This process is still being developed. Will need a REST API. *Process to create new alerts*: **Yes (30%)** The process will need to fill a form with all necessary data. The Engine alert has been deployed. For the UI will be used Praeco that has been deployed.
The portion of completed requirements	Following D3.2 requirements for the Dashboard and Notifications: *Multilanguage support [43]:* **50%** The Dashboard and Notifications will be able to work with different languages (English, Spanish, French, Italian and Greek).

	Web UI [100]: **50%** The Dashboard and Notifications must provide for each of its tools, a Web based UI. This UI will be developed attending these points: • Adoption of web standards (e.g. HTML, CSS, JavaScript) • Portability on different devices (e.g. responsiveness) • Readability • Easy to use *Portability [103]:* **50%** The Dashboard and Notifications components will need to be generic enough to be easily deployable for any port. *Configurable Dashboard [66]:* **50%** PIXEL Dashboard module will be configurable. Will allow incorporating widgets to visualize a great amount of information. *Visualization of data [105]:* **50%** PIXEL Dashboard will offer options (widgets) for visualizing data with different visualization options for a better readability. *PEI Dashboard – Time Series [96]:* **10%** PIXEL Dashboard will provide an effective web interface to present to the stakeholders the calculate PEI, for each use case including the evolution of PEI over time
Performance efficiency	
Average latency	Dashboard and Notifications-API-Write Operations (widgets) : **71 ms** Dashboard and Notifications -API-Read Operations (widgets): **70 ms** Dashboard and Notifications-API-Write Operations (notifications): **71 ms** Dashboard and Notifications -API-Read Operations (notifications): **70 ms**

Mean CPU Utilisation	Dashboard and Notifications-API-Write Operations (widgets) : **1 %** Dashboard and Notifications -API-Read Operations (widgets): **1.5 %** Dashboard and Notifications-API-Write Operations (notifications): **1 %** Dashboard and Notifications -API-Read Operations (notifications): **1.5 %**
Mean memory usage	Dashboard and Notifications-API-Write Operations (widgets) : **1.21 %** Dashboard and Notifications -API-Read Operations (widgets): **1.3 %** Dashboard and Notifications-API-Write Operations (notifications): **1.21 %** Dashboard and Notifications -API-Read Operations (notifications): **1.3 %**
Maximum memory usage	Dashboard and Notifications-API-Write Operations (widgets) : **1.21 %** Dashboard and Notifications -API-Read Operations (widgets): **1.3 %** Dashboard and Notifications-API-Write Operations (notifications): **1.21 %** Dashboard and Notifications -API-Read Operations (notifications): **1.3 %**
Maximum processing power used	Dashboard and Notifications-API-Write Operations (widgets) : **1.21 %** Dashboard and Notifications -API-Read Operations (widgets): **1.3 %** Dashboard and Notifications-API-Write Operations (notifications): **1.21 %** Dashboard and Notifications -API-Read Operations (notifications): **1.3 %**
Reliability	

Simultaneous requests	Phase 2 (D8.3)
% Monthly availability	Phase 2 (D8.3)
Success rate	Phase 2 (D8.3)
Maintainability	
% of modularity	**80%** Dashboard and Notifications-API. Individual component (requires DDBB- Mongo) Dashboard and Notifications-UI. Requires Dashboard and Notifications-API Dashboard and Notifications- Event processing – Individual component
% of reusable assets	**85%** Dashboard and Notifications-API. Reusable 75% (ad hoc component, needs adaptation) Dashboard and Notifications-UI. Reusable 75% (ad hoc component, needs adaptation) Dashboard and Notifications-Engine Alert –ElastAlert. Reusable 100% (open source software)
Level of analysability	Phase 2 (8.3)
Portability	
Mean number of errors per hardware or OS change/ upgrade	Phase 2 (D8.3)

Mean number of errors per software change/ update	Phase 2 (D8.3)
Mean number of errors per software install	Phase 2 (D8.3)
Mean number of errors per software uninstall	Phase 2 (D8.3)

3.8.3. Problems Faced and Recommendations

The main problem faced at the time to assess the Dashboard and Notifications is their dependency on other modules. Some functionalities will not be tested until a final deployment (WP7, integration pilots).

Since the objective of the dashboard is to show the results on the screen. It is necessary that there is a complete interaction between all the modules to achieve an end-to-end scenario.

The main problems encountered for a full evaluation of this module are:

- Lack of real time data available to recover information and show results (visualizations, widgets).
- Lack of models and predictive algorithms converted into services. This means that there can be no communication with OTools to publish their data.

The previous problems are:

- Consequence of the schedule of the project (e.g. models have recently been released as executables in D4.2, but require adaptation, whereas predictive algorithms are still being implemented).
- Time necessary of integration among different modules.

Such problems do not represent a risk for the project. The deliverable D8.3 will incorporate a final evaluation of this module.

3.9. PIXEL Security

3.9.1. Assessment scenario

KPIs assigned to the assessment (PIXEL Security assessment) have been described in D8.1, while the tools and methods for their collection has been defined in D6.3. In this section this methodology is further elaborated, and the assessment scenario is described in detail.

KPIs are estimated either by *expert judgement* or by the development of tools for *automated measurements*.

Expert judgement is performed using desk research, where an expert evaluates the KPI using the approach defined in D8.1/D6.3. **Functional suitability** and **Maintainability** will be estimated using this approach.

Automated measurements are performed either by usage of existing evaluation software or by development of custom tools for this purpose.

Part of the KPIs will this be collected using **JMeter measurements.** The Apache JMeter™ application is an open-source software designed to load test functional behaviour and measure performance. **Performance efficiency** and **Reliability** have been measured using this approach.

In order to assess the performance in the port area, measurements will be performed with a predefined set of realistic input data relevant to port operations. In the beginning, all measurements will be performed under laboratory conditions, and on the infrastructure, which will be defined in WP7 (cloud environment v.s. on-premises installation and other parameters).

Custom modules: reliability and **portability** are going to be measured used custom modules.

Reliability, portability and few other KPIs depend on the deployment of the modules in an operational scenario in order to measure them, as they are mostly statics related to an operational environment.

Table 33: KPI for Security

KPI	Measurement method	Reporting
Functional suitability		
Straightforward task accomplishment	Expert judgement	D8.2, D8.3
The portion of completed requirements	Expert judgement	D8.2, D8.3
Performance efficiency		
Maximum number of connected data sources	JMeter	D8.3
Maximum database size	(JMeter)	D8.3
Average latency	JMeter	D8.3
Throughput	JMeter	D8.3
Mean CPU Utilisation	JMeter	D8.3
Mean memory usage	JMeter	D8.3
Maximum memory usage	JMeter	D8.3
Maximum processing power used	JMeter	D8.3
Security		
Incidents of ownership changes and accessing prohibited data	Expert judgement	D8.3
Incidents of authentication mechanisms breaches	Expert judgement	D8.3
Level of User authenticity	Expert judgement	D8.3
Reliability		
Simultaneous requests	JMeter	D8.3

% Monthly availability	Custom module, Phase 2 based on Orion API	D8.3
Success rate	Custom module, Phase 2 based on Orion API	D8.3
Maintainability		
% of modularity	Expert judgement	D8.2
% of reusable assets	Expert judgement	D8.2
% of update	Expert judgement, Phase 2	D8.3
Level of analysability	Expert judgement	D8.2
Portability		
Mean number of errors per hardware or OS change/ upgrade	Custom module, Phase 2	D8.3
Mean number of errors per software change/ update	Custom module, Phase 2	D8.3
Mean number of errors per software install	Custom module, Phase 2	D8.3
Mean number of errors per software uninstall	Custom module, Phase 2	D8.3

3.9.2. KPI Data Collection and Results

3.9.2.1. Expert judgement method

Expert judgement has been used for those KPIs that are either too complicated to automate and an expert approach is more efficient, or where a more qualitative evaluation approach is needed. In the following section we report the assessment procedure and the result of the expert judgement.

Functional suitability

Straightforward task accomplishment: "*Processes for authentication and authorization will be analyzed to verify that they do not include unnecessary steps. "*
The process for authentication and authorization use the standard Oauth2 protocol and the mechanism developed by the FIWARE Foundation, with its Identity Management Components. Those protocol and mechanism are compliant with the state of the art. **Overall, for the listed functionalities the value is yes.**

The portion of completed requirements: *"Should have" and "Must have" requirements from deliverable D3.2 will be taken as input in order to extract all requirements specifically targeting T6.6.*

Table 34 lists all PIXEL requirements related to the Security Layer that have the priority set to *"Should have"* or *"Must have"*. It also lists other PIXEL software modules related to the requirements and the status of development in the Security. The status does not assess the fulfilment of the requirement in other modules.

Table 34: PIXEL Security Requirements ("Should have" and "Must have") and implementation status

Requirement	Addressed in additional modules	Implemented in Security component
Common non-functional requirements		
Security communications between components [68] *Status: The Security Layer provides API to manage user and role. It provides also component that can be used as API Gateway.*	DAL, IH , OT, DB	partial
Data source API connectivity [85] *Status: NGSI Agent that exposed an API are accessible through the PEP Proxy (OAuth2) using HTTPS.*	DAL	yes
Access Security [97] *Status: Securiy Layer provides components to secure the access to the PIXEL platform. Work still have to be done to secure it from internal access.*	DAL, IH , OT, DB	partial

Legend:

- yes: common functional requirements that are implemented in the Data Acquisition Layer
- partial: work in progress
- no: The functionality is not yet available.

Result: A total of 3 requirements are related to functionality provided by the Security. Out of those, 1 are fully available, 2 in progress.

- Total requirements: 3
- Fulfilled requirements (functionality available): 1
- Portion of completed requirements: 30%.

This KPI has to be re-checked in D8.3 in order to verify that the provided functionality fulfils the acceptance criteria in specific pilot executions.

Maintainability

% of modularity: *Will be measured by reporting all the independent components that are part of the security module and comparing them to the number of all components in the security module. Individual operation means that a component can offer a complete function with meaningful information in the context of PIXEL.* As defined in WP6 deliverables, PIXEL Security Layer is composed of several components that provide different feature of the Security implementation. Those components are FIWARE Generics Enabler that implements Identity Management (Keyrock), Authorization (AuthZForce) and Access control (PEP Proxy Wilma)

Result: modularity is 100%.

% of reusable assets: *Will be measured by reporting all the reusable components that are part of the security layer module and comparing them to the number of all components in the Security. A reusable component is considered any that can be applied in a different context of PIXEL with no modifications of the source code.*

All modules in the PIXEL Security Layer are FIWARE Generic Enablers that could be reuse on any FIWARE compatible projects.

Result: 100%.

Level of analysability: *Will be measured by reporting the ratio between the numbers of items inside the Security for which logging is implemented compared to the number of items for which the specifications require logging.*

All PIXEL Security components provide logging capabilities.

Result: 100%.

Summary of results are provided in table 35.

Table 35: Summary security results

KPI	Measurement approach
Functional suitability	
Straightforward task accomplishment	yes
The portion of completed requirements	30%, conditional on the implementation of pilots in WP7.
Maintainability	
% of modularity	100%
% of reusable assets	100%
Level of analysability	100%

3.9.2.2. Automated data collection and results

PIXEL Data Acquisition Layer relies on FIWARE Generic Enabler: Keyrock, Wilma and AuthzForce. The FIWARE foundation provides Performance testing result and script for them, a full test session will be organized after the pilot deployment.

3.9.3. Problems Faced and Recommendations

The main problem faced at the time to assess the Security layer is to identify the exact features needed by the PIXEL Infrastructure to secure the access between each component. As we are still working on the overall integration it is not easy to identify the right component to ensure the good level of security without impacting the performance and functionality of the all infrastructure. But as all components communicate with each other using REST API, a lot of solution is available to address this point.

The Security KPI evaluation needs to be done after the WP7 deployment in order to be able to analyse log information. Such problems do not represent a risk for the project as we have identity out of the box candidate to fulfil our needs. The deliverable D8.3 will incorporate a final evaluation of this module.

4. Technical Impact Assessment of the PIXEL Use Cases

The scope of the present chapter is to present the technical impact assessment of the implementation of the PIXEL Platform in the four use cases of the Project. The assessment takes into consideration the different measures that have been implemented in each port. In the sections that follow, the state of integration in each port is described, along with the data collection and data analysis methods that will be used to achieve the set goals.

4.1. State of the Integration

Integration of the PIXEL platform has started in the ports in M16. D7.1 gives us a good view of what has been done up to the release of the document.

However, integration is not over, and we didn't reach a sufficient level in any of the use-cases in order to process the technical impact assessment of the PIXEL Use-cases. We will instead briefly summarize in the sections below what state did the integration reached. We can then have a feeling of how close we are from evaluating user perception over the platform and show what is done in the integration to satisfy user's needs.

4.1.1. Energy Management Use Case - GPMB

The PAS Modelling has been developed considering GPMB's first supply chain configuration. It allows us to calculate a first version of the energy demand for the port. Data pipelines development in order to acquire electrical data consumptions also started to be implemented and we are confident to have a first working version upon the release of this deliverable.

The different partners involved iterate in close relationship with the port, allowing us to take final users interests into consideration while proceeding to the integration.

4.1.2. Intermodal Transport Use Case - ASPM / SDAG

The intermodal transport model has been developed by considering the peculiarities of the Monfalcone port, that are: no container transport, use of the port area as warehouse and huge traffic of slabs from port to industrial districts. The model will be integrated in the future in order to extract information directly from the IH, concerning vessels and truck traffic. Integration is currently under development with both PMIS and SILI information systems. The model has already been described and presented to the different operators of the Port in order to understand its effectiveness and, at the same time, to evaluate how it could proficiently use to prevent and/or simulate critical events (e.g.: truck congestion at port entrance).

4.1.3. Port City Integration Use Case - THPA

The Port and City environmental management model and the PAS were developed considering the special attributes of the port and parameters summarizing the port activity mechanisms, as well as end users' preferences. They can estimate the port activity scenarios and identify the main areas affected. The integration phase is still in progress, but so far, the models that can reflect the current situation or some potential future scenario, of the impacts caused by port operations, have been developed and will eventually be incorporated in the PIXEL platform. The aim is to connect, test and validate all software components developed in PIXEL that are useful for the Thessaloniki pilot trial, including the integration of the data sources provided by the port.

For the purposes of the PIXEL project, THPA has developed an API, in order to share its operational data. Data acquisition through the API is from a number of different THPA data sources, the Statistics application, the

TOS (FRETIS) and the in-house application for gates' traffic, all now connected through the API with the Data Acquisition Layer. By using weather data simulations and the models developed, they can assist the port manager/operator in the decision-making process in order to optimize various activities within the port, as well as recourses allocation and minimize their impact on the environment.

As far as hardware is concerned, the incorporation of new sensors will be required, since for the time being, only a wind meter is available. Apart from that, a possible need in storage will be required, since some of the optional output files of concentration data can be rather large.

4.1.4. Port City Integration Use Case - PPA

PPA has been collecting many datasets of various vessels types statistics from both internal PPA systems and subscription database services. The data has become available to the technology partners in various formats. One of the different prediction tasks developed by Prodevelop in Pixel environment is the road traffic prediction at the PPA port surrounding area and the impact that port activity has in traffic behaviour. For this purpose, different free services as TomTom API has been used in order to collect traffic data in real time and Prophet for time series forecasting. In addition to baseline information, some extra attributes are being used in order to explain some variations in traffic and improve the accuracy of the model. Some of these additional attributes are weather information, traffic incidents and port activity as cruise arrivals, between other types of vessels.

PEI data has also been collected and became available for testing to the responsible partners while there is an ongoing effort to obtain and integrate data automatically into the PEI tool. Acquisition of new sensors to obtain real time data of air pollution and noise levels is also under way.

4.1.5. Port Environmental Index Use Case

The calculation method of the PEI has been elaborated/developed and few environmental indicators have been proposed/selected from the PIXEL' pilot ports (use case). The selected Port Environmental indicators are small and different from one port to another.

The PEI work is in progress and the future tasks are to integrate the data from ports, to produce an interactive PEI tool using a single composite indicator for realising and assessing the environmental footprint of the PIXEL's ports.

4.2. Data Collection Methodology

For the technical assessment of the PIXEL Use-Cases, we chose and agreed in D8.1 on the KPIs to evaluate. The Quality in-use model and the Data Quality model related KPIs can be either measured using a quantitative method or retrieve through a more qualitative method.

The evaluation of the quantitative KPIs is straightforward as it involves the same process than for the technical impact assessment of the PIXEL platform. Users only must answer facts, making the results completely objectives.

For the qualitative KPIs, however, we will use questions defined or derived from the TAM3 and the AIMQ models:

- TAM3 is an information model theory that aim to model how end-users of a system may come to accept and use it. It primarily tries to model factors that could influence their decisions, which are the "Perceived usefulness" and the "Perceived ease-of-use".
- AIMQ is a complete methodology for information quality assessment.

The data collection will be mainly based on the questionnaires that will be created for each one of the use cases. Different questionnaires, either using the TAM3 or the AIMQ model, have been created in order to assess the quality in use and the data quality for each one of the Use Cases. The questions included in the questionnaires have been based on the KPIs that have been identified in the Evaluation Plan. The questions aim to assess the following issues:

- Timeliness of the data;
- The correctness, accuracy and reliability of the data;
- The credibility of the data;
- The accuracy, precision of data;
- The traceability of data;
- The easiness with which data is made available and accessible;
- The comprehensibility of data;
- The degree to which the availability of data provides the user with an advantage;
- The relevance of data;
- The concise representation of data;
- The easiness with which data is interpreted;
- The consistency and completeness of data.

4.3. Data Analysis Methodology

When we chose to evaluate the PIXEL Use Cases with ISO/IEC methodologies, we defined which characteristics/sub-characteristics were of interest.

In order to evaluate those characteristics, TAM-3 and AIMQ questionnaires methodologies allow us to define questions and give a KPI per characteristics. While many KPIs talk by themselves (i.e. *% of completed user stories*, *Efficiency level*, etc…), those that come from questionnaires may not be as direct as the quantitative ones. In the following, we will define techniques to analyse data received from the questionnaires.

4.3.1. TAM-3 Data Analysis

While the TAM-3 model presents the determinants that influence the Behavioural Intention to use the product, and so the Use Behaviour regarding the product, the paper also introduces the items that are used to assess the model.

Reusing the items and deriving new items allow us to obtain scores from questionnaires that form our KPIs. In order to quantify and obtain a feeling on how well people are confident using the platform, we can compare a study that will be done right before starting to use the platform with another study done when people will have a bit of expertise with using the platform.

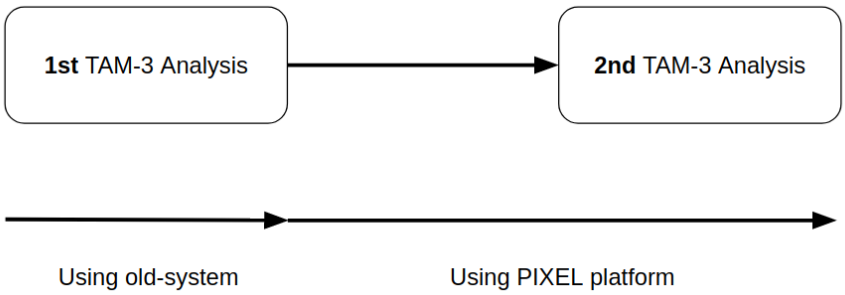


Figure 17: The two-steps TAM-3 Questionnaires for data analysis

4.3.2. AIMQ Data Analysis

The authors of the AIMQ method classified the characteristics into four quadrants, which are defined below:

Table 36: PSP/IQ 4 quadrants with the corresponding characteristics (characteristics in grey are not evaluated in PIXEL context)

Sound information	Useful information	Dependable information	Usable information
Free-of-error	Appropriate amount	Timeliness	Believability
Concise representation	Relevancy	Security	Accessibility
Completeness	Understandability		Ease of operation
Consistent representation	Interpretability		Reputation
	Objectivity		

From the above shown characteristics, we will have at our disposal those that are not greyed out, allowing us to calculate the different quadrants of the PSP/IQ model, defined by the AIMQ methodology.

We can then use those quadrants values to calculate “Usability Benchmark Gaps” per use-cases and check the axis of improvement for every use-cases.

We can also compute the Usability Role Gap, using the roles (primary/secondary/indirect) defined in D8.1. As we will need characteristics that are evaluated by different roles, and referencing to Table 9 of D8.1, we see that primary and secondary users only evaluate two commons characteristics, while primary and indirect users evaluate 11 common characteristics. Thus, we will be able to calculate a Usability Role Gap between primary and indirect users.

5. Conclusion and future work

This document shows the first results of performance, user acceptance and security evaluation of the PIXEL platform. It will be use by PIXEL partner to improve development in the next months. The technical evaluation will be considered in every development and the full technical impact assessment will be provided at the end of the project (M36) in the second version of Technical Evaluation (Deliverable D8.3)

Appendix A - Technical Impact Assessment Survey

In order to identify which characteristics or sub-characteristics are relevant for PIXEL, a survey has been shared with all partners. The objective was to select the most adequate characteristics and sub-characteristics. Results of the study, for both models, are shown below. Only characteristics/sub-characteristics that were at least consider "Could have" are evaluated.

Table 37: Consortium answers to the application of Product Quality Model characteristics to PIXEL platform (green: must be assessed, yellow: should be assessed, orange: could be assessed, red: won't be assessed)

Product Quality Model		
Functional suitability		
Functional appropriateness	92%	
Functional completeness	83%	
Functional correctness	50%	
Performance Efficiency		
Capacity	75%	
Time behaviour	67%	
Resource utilisation	67%	
Compatibility		
Interoperability	100%	
Co-existence	33%	
Operability		
Ease of use	83%	
Technical Accessibility	75%	
User interface aesthetics	50%	
User error protection	42%	
Appropriateness recognisability	33%	
Technical Learnability	33%	
Reliability		
Maturity	83%	
Availability	83%	
Recoverability	50%	
Fault tolerance	17%	
Security		
Confidentiality	100%	
Integrity	100%	
Authenticity	67%	

Accountability	42%	
Non-repudiation	25%	
Maintainability		
Modularity	92%	
Reusability	83%	
Modifiability	75%	
Analysability	58%	
Testability	42%	
Portability		
Adaptability	92%	
Installability	75%	
Replaceability	17%	

Table 38: Consortium answers to the application of "Quality In Use Model" and "Data Quality Model" characteristics to PIXEL platform use cases (green: must be assessed, yellow: should be assessed, orange: could be assessed, red: won't be assessed)

Quality in Use Model				**Data Quality Model**		
Effectiveness				**Information Accuracy**		
Effectiveness	100%			Currentness	83%	
Efficiency				Correctness	75%	
Efficiency	100%			Credibility	75%	
Satisfaction				Precision	75%	
Usefulness	92%			Traceability	58%	
Trust	92%			**Information Accessibility**		
Comfort	42%			Accessibility	92%	
Pleasure	17%			**Information Appropriateness**		
Safety				Understandability	100%	
Environmental harm risk	42%			Value Added	92%	
Economic damage risk	33%			Representational Adequacy	83%	
Health and safety risk	33%			Consistency	75%	
Usability				Completeness	58%	
Flexibility	83%			**Efficiency**		
Learnability	75%			Efficiency	58%	
Accessibility	67%			**Availability**		
Content conformity	67%			Availability	100%	
				Portability		
				Portability	75%	
				Recoverability		
				Recoverability	50%	

Appendix B - KPI evaluation for PIXEL tasks

The table below shows the association between the different tasks and the calculable KPIs for the Product Quality Model. It is often referred as table 5 of D8.1 in this document.

Table 39: KPI evaluation for PIXEL tasks results (Table 5 of D8.1)

KPIs	T4.1	T4.2	T4.3	T4.4	T4.5	T5.3	T6.2	T6.3	T6.4	T6.5	T6.6
Straightforward task accomplishment	X	X	X	X	X	X	X	X	X	X	X
Portion of completed requirements	X	X	X	X	X	X	X	X	X	X	X
Maximum number of connected data sources					X		X				
Maximum database size					X						
Average latency					X	X	X	X	X	X	
Throughput					X		X	X			
Mean CPU Utilisation	X	X	X	X	X	X	X	X	X	X	
Mean memory usage	X	X	X	X	X	X	X	X	X	X	
Maximum memory usage	X	X	X	X	X	X	X	X	X	X	
Maximum processing power used	X	X	X	X	X	X	X	X	X	X	
% of APIs coverage							X				
Ability to acquire data from different data formats							X				
Ability to support different IoT platforms							X				
Ability to export different data formats							X				
Dashboard availability									X	X	
Notifications system availability										X	
GUI module availability						X	X	X	X	X	X
WCAG 2.0 Conformance Level						X	X	X	X	X	X
Maximum Concurrent users											
Simultaneous requests	X	X	X	X	X	X	X	X	X	X	X
% Monthly availability							X	X	X	X	X
Success rate							X	X	X	X	X
Incidents of ownership changes and accessing prohibited data											X

Incidents of authentication mechanisms breaches											X
Level of User authenticity											X
% of modularity	X	X	X	X	X	X			X	X	X
% of reusable assets	X	X	X	X	X	X			X	X	X
% of update									X	X	X
Level of analysability									X	X	
Mean number of errors per hardware or OS change/ upgrade									X	X	X
Mean number of errors per software change/ update									X	X	X
Mean number of errors per software install									X	X	X
Mean number of errors per software uninstall									X	X	X

Appendix C – Performance metrics for the Operational Tools

1. **Publishing a model (write operation)**

For a read performance test, we have selected to publish a new model in MaaS (Model as a Service) mode For this test we have used Jmeter with the following configuration:

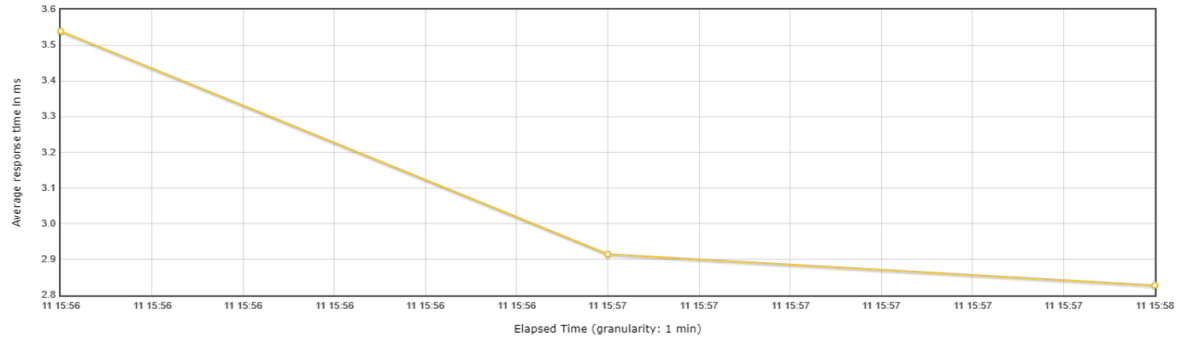
- Number of Threads (users): 30
- Ramp-up period (Seconds): 50
- Duration (seconds): 150

The results are presented below (Jmeter provides plenty of results but here we will provide the most relevant ones):

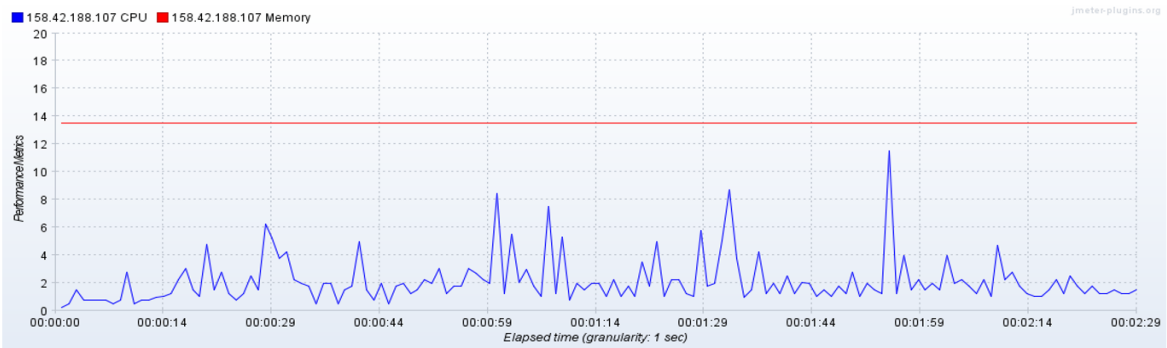
Statistics

Requests	Executions			Response Times (ms)						Throughput	Network (KB/sec)	
Label	#Samples	KO	Error %	Average	Min	Max	90th pct	95th pct	99th pct	Transactions/s	Received	Sent
Total	**3594**	**0**	**0.00%**	**3.01**	**1**	**144**	**4.00**	**4.00**	**6.00**	**23.97**	**20.86**	**22.31**
Energy Model	3594	0	0.00%	3.01	1	144	4.00	4.00	6.00	23.97	20.86	22.31

Response Times Over Time



CPU and RAM (PerfMon)



2. Get model information (read operation)

For a read performance test, we have selected to get the list of all available models. For this test we have used Jmeter with the following configuration:

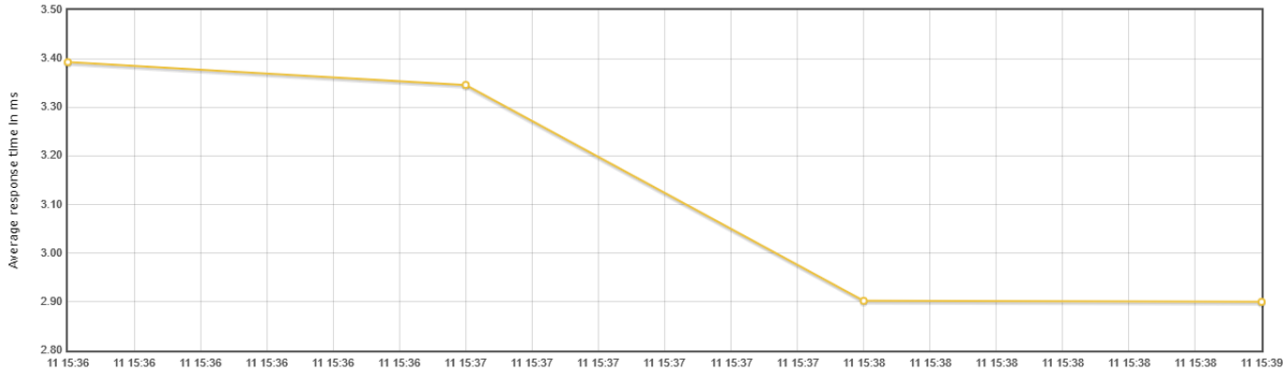
- Number of Threads (users): 30
- Ramp-up period (Seconds): 50
- Duration (seconds): 150

The results are presented below (Jmeter provides plenty of results but here we will provide the most relevant ones):

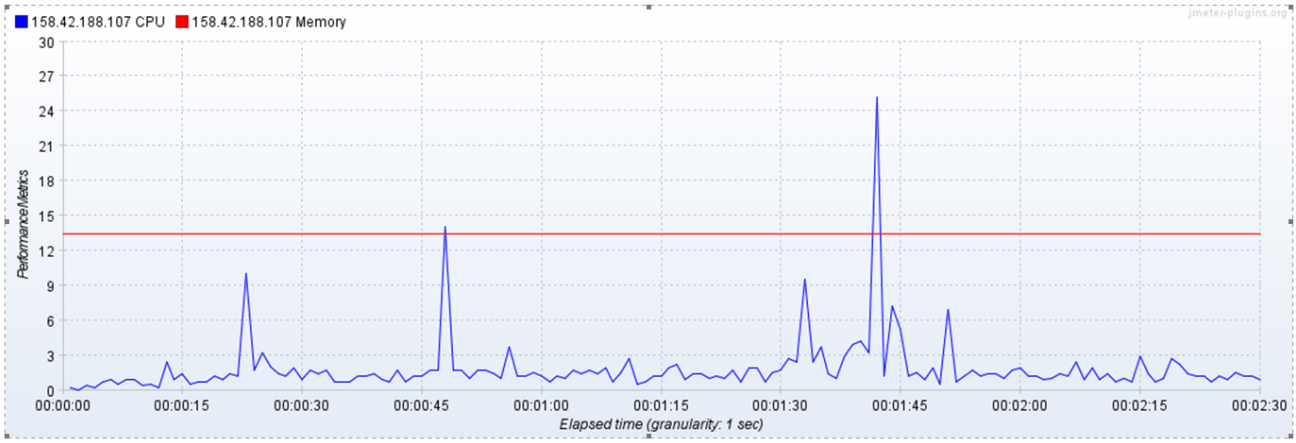
Statistics

Requests	Executions			Response Times (ms)						Throughput	Network (KB/sec)	
Label	#Samples	KO	Error %	Average	Min	Max	90th pct	95th pct	99th pct	Transactions/s	Received	Sent
Total	3596	0	0.00%	3.10	1	106	4.00	5.00	6.03	23.98	95.11	5.04
Energy Model	3596	0	0.00%	3.10	1	106	4.00	5.00	6.03	23.98	95.11	5.04

Response Times Over Time



CPU and RAM (PerfMon)



3. Deployment of a model

For this test we have tested the script of the Operational Tools required to download a Docker image (in our test from Docker hub) and run the Docker image, in terms of CPU and load consumption. This is not a service

requested directly through the OT API. The user, when publishing a model in MaaD (Model as Service) mode, does not wait until the end, the answer is immediate, and the model is scheduled to be deployed in the next iteration of the script. Therefore, we will here only test the script and check the impact on RAM and CPU. Obviously, this can be variable from model to model in terms of RAM usage (requirements). In order to have some comparable values, we have also evaluated another model, which is a simplified version of the PEI (note that it is still being developed and it is only an initial draft). For this test we have not used Jmeter (no real API endpoint for that), but the SAR (system Activity Report), a GNU/Linux tool able to provide activity status (CPU, memory, disk) throughout a period of time (and therefore obtain data and graphs).

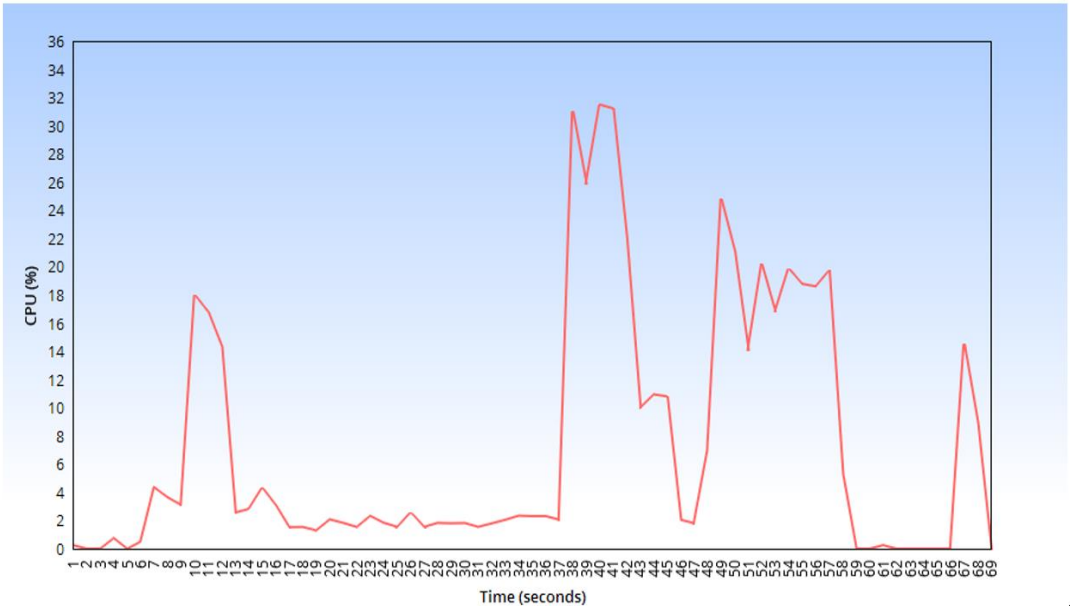
PAS- ENERGY MODEL

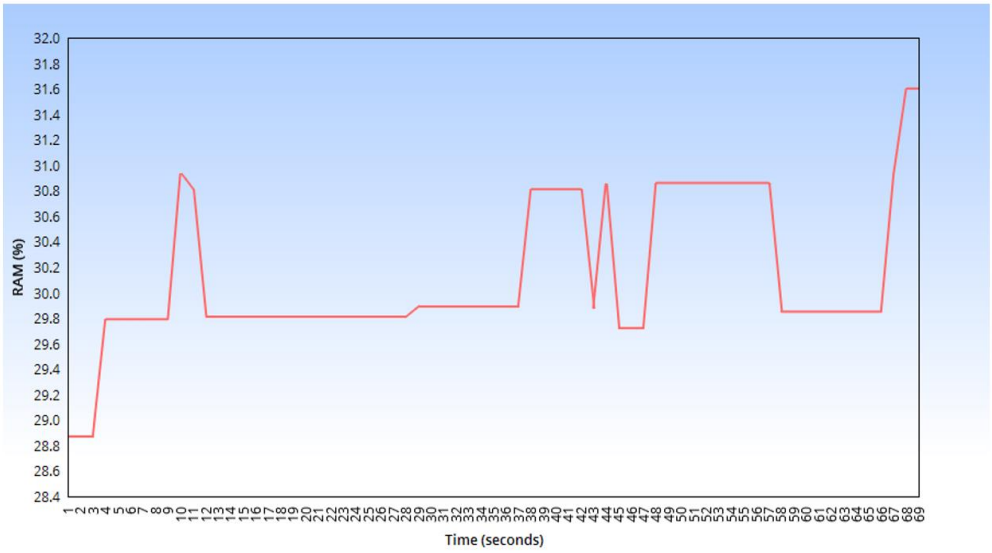
Command: docker run -d -p 5000:5000 pixelh2020/pas_energy:0.0.1

Duration:69 seconds

CPU: max:31,49 % min0% avg 6,7%

RAM increment: 2,73% 550,73 MB





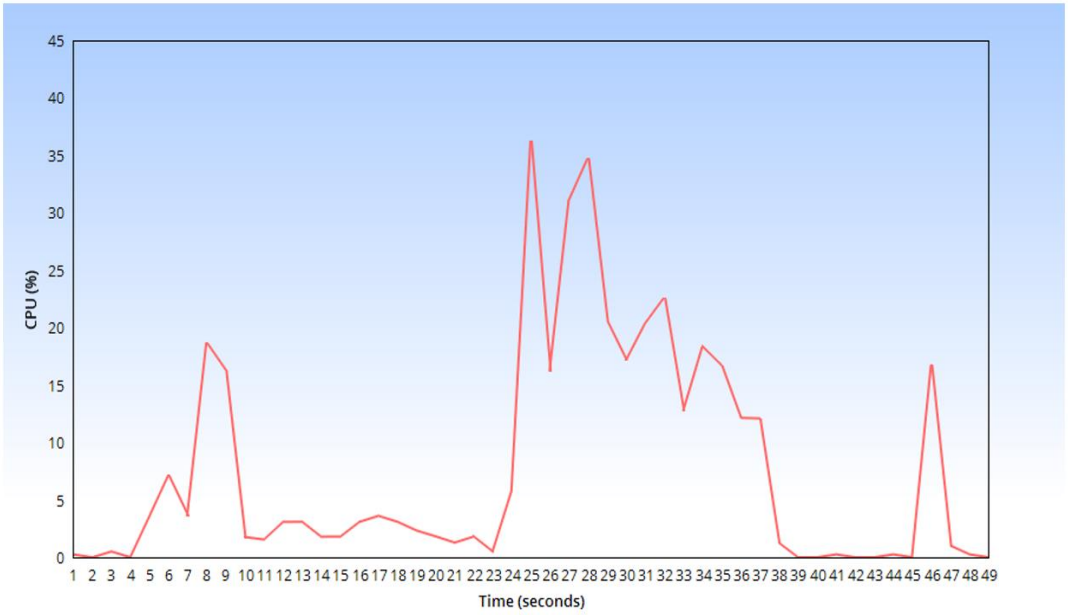
PEI

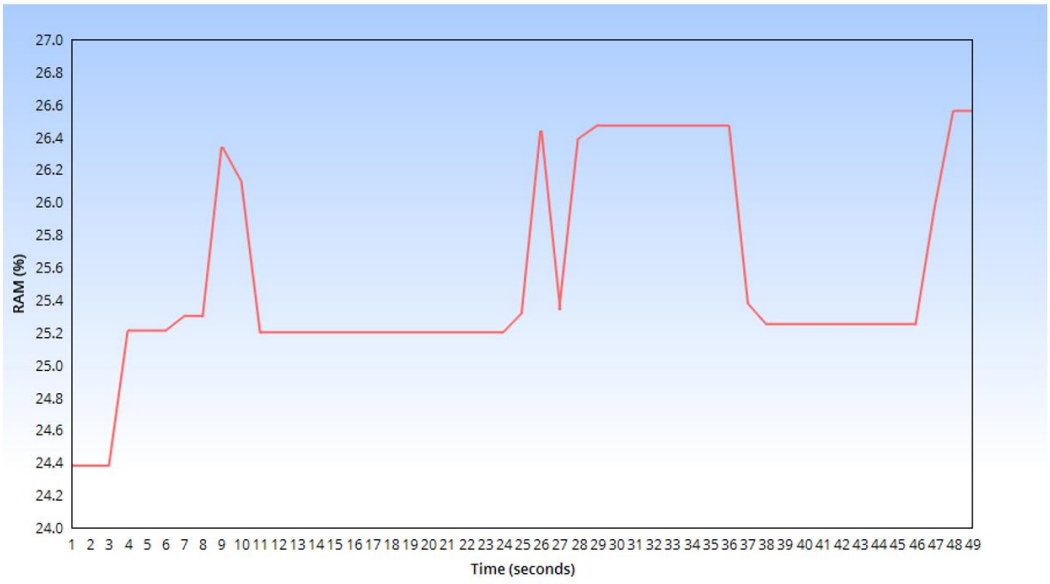
Command: docker run -p 8082:8080 -d pixelh2020/pei:0.0.3

Duration: 49 seconds

CPU: max: 36,32% min:0% avg: 7,7%

RAM increment: 2,18% -- 438,69 MB





4. Execution of a model

For the execution of the model we have tested the PAS-Energy model and checked the activity under two different scenarios (inputs):

- Considering the income of a new ship into the port. This might correspond to a recalculation whenever a new ship announces its intention to come to the port for loading/unloading operations, so that the port operator might check the impact.

- Considering the income of several ships (78) into the port. This refers to a scenario where the port operator might want to make an analysis about a certain period of time.

For these tests we have used Jmeter with the following configuration:

- Number of Threads (users): 30
- Ramp-up period (Seconds): 50
- Duration (seconds): 150

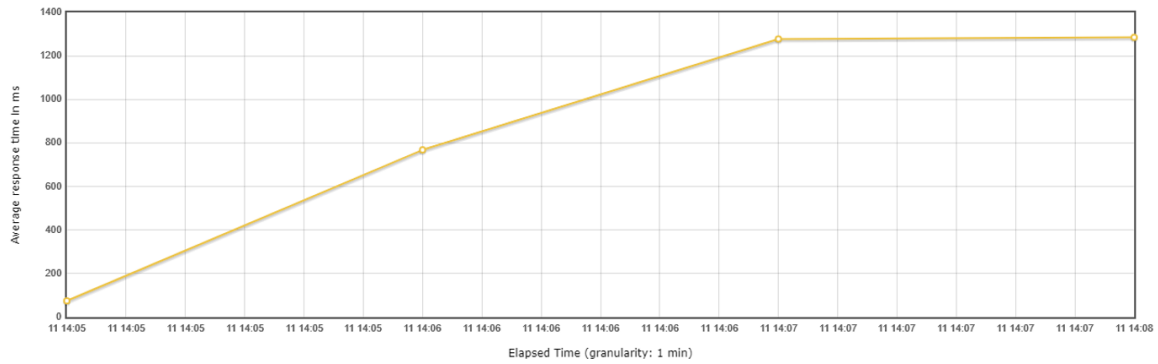
The results are presented below (Jmeter provides plenty of results but here we will provide the most relevant ones):

Single ship

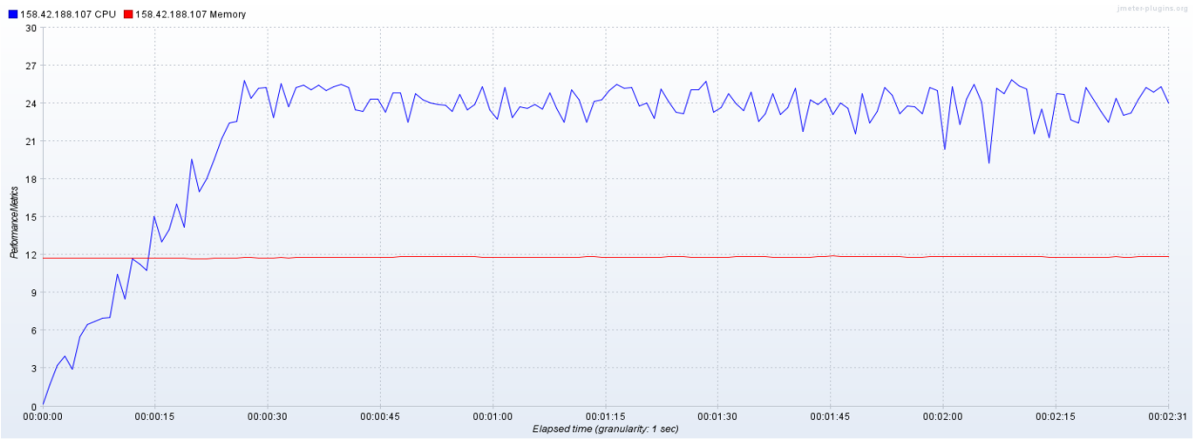
Statistics

Requests	Executions			Response Times (ms)						Throughput	Network (KB/sec)	
Label	#Samples	KO	Error %	Average	Min	Max	90th pct	95th pct	99th pct	Transactions/s	Received	Sent
Total	**1794**	**0**	**0.00%**	**1072.02**	**63**	**2329**	**1601.50**	**1725.00**	**1947.50**	**11.87**	**946.03**	**209.25**
Energy Model	1794	0	0.00%	1072.02	63	2329	1601.50	1725.00	1947.50	11.87	946.03	209.25

Response Times Over Time



CPU and RAM (PerfMon)

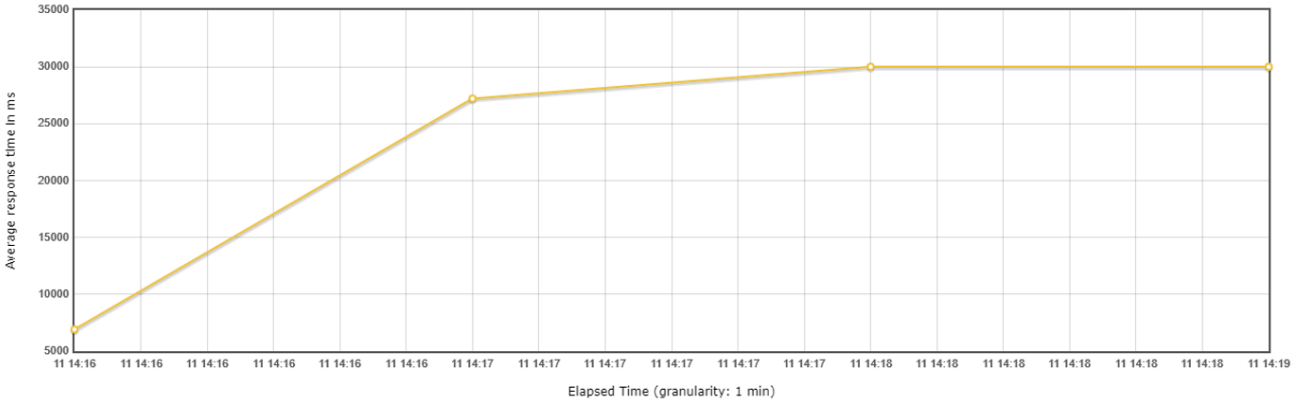


Multiple ships

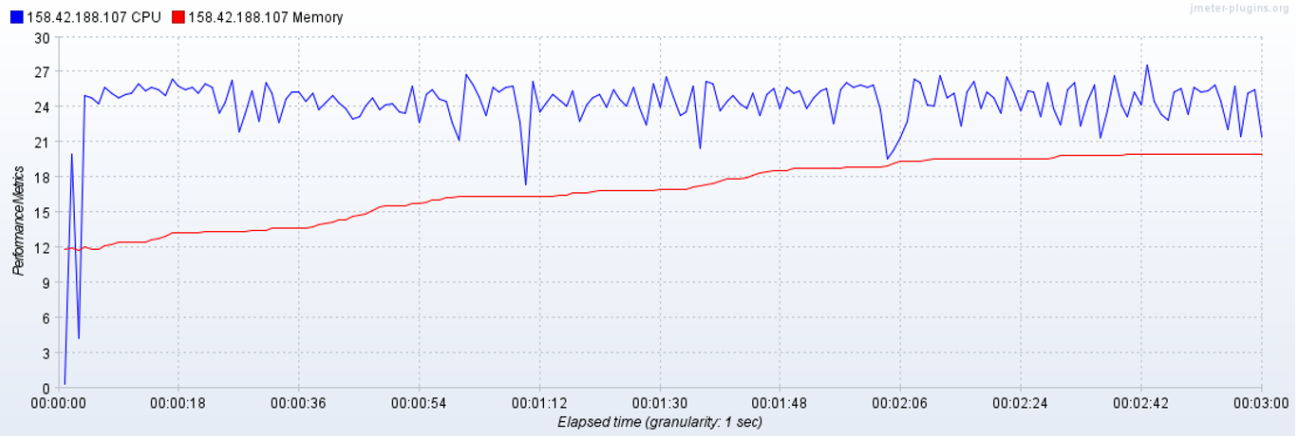
Statistics

Requests	Executions			Response Times (ms)						Throughput	Network (KB/sec)	
Label	#Samples	KO	Error %	Average	Min	Max	90th pct	95th pct	99th pct	Transactions/s	Received	Sent
Total	**157**	**121**	**77.07%**	**26028.61**	**880**	**30119**	**30013.00**	**30020.00**	**30083.62**	**0.87**	**179.33**	**13.16**
Energy Model	157	121	77.07%	26028.61	880	30119	30013.00	30020.00	30083.62	0.87	179.33	13.16

Response Times Over Time



CPU and RAM (PerfMon)



Note: results for response times are valid only at the beginning, it seems to saturate early to a value of 30 seconds (Socket timeout) when the number of concurrent users increases. A lot of errors appear in JMeter after that. Therefore, it is not advisable to launch multiple large/historical executions in parallel. For this scenario, we have also monitored the usage of resources via the Portainer tool (see figure below). Here one can easily see that the model saturates one CPU core and increase considerably the memory usage.