Ref. Ares(2022)4804476 - 30/06/2022

EnerMan

Energy Efficient Manufacturing System Management

D5.1 - Preliminary EnerMan Prototype Integration Report

Date	: 30/06/2022
Deliverable No	: D5.1
Responsible Partner	: MAG
Dissemination Level	: Public

HORIZON 2020
This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

EnerMan

Short Description
This deliverable describes the first report towards an integrated solution and the realization of the EnerMan framework. It includes the analysis of technical and implementation requirements of all modules until M18 as well as the deployment and management of necessary resources for the implementation requirements. The described work gives emphasis to the gathering of the requirements for each EnerMan module and prototype, to ensure a smooth and effective integration of the components

Project Information	
Project Acronym:	EnerMan
Project Title:	ENERgy-efficient manufacturing system MANagement
Project Coordinator:	Dr. Ing. Giuseppe D'Angelo CRF giuseppe.dangelo@crf.it
Duration:	36 months

Document Information & Version Management			
Document Title:	D5.1 Preliminary EnerMan Prototype Integration Report		
Document Type:	Report		
Main Author(s):	Panagiotis Katrakazas, Astik Samal (MAG)		
Contributor(s):	CRF, FHOOE, UNINA, SUPM, STS, TSI, ITMLCY, ISI, AEGIS, UCY, UOP		
Reviewed by:	Johann Bachler (AVL), Liam Moore (DPS)		
Approved by:	Dr. Ing. Giuseppe D'Angelo (CRF)		
Version	**Date**	**Modified by**	**Comments**
V0.1	01/06/2022	Katrakazas (MAG)	Initial Version
V0.2	13/06/2022	Bouklas, Marmpena (ITMLCY), Rega (UNINA)	Updated Content
V0.3	15/06/2022	Katrakazas (MAG)	Revised Content, Major Updates
V0.4	23/06/2022	Katrakazas, Casanova, Samal, Pruccoli (MAG), Marmpena (ITMLCY), Morianos (TSI), Hildenbrand (SIMPLAN), Birbas, Alefragkis (UOP)	Reviewed Content, Minor Updates
V0.5	24/06/2022	Samal, Katrakazas (MAG)	Use Cases and Conclusion Sections Finalised
V0.6	29/06/2022	Moore (DPS), Bachler (AVL)	Reviewed Version
V0.7	29/06/2022	Samal, Katrakazas (MAG)	Comments of Reviewers Addressed Final Version to be Submitted
V0.8	30/06/2022	Nilay Yalcinkaya Yoruk (INTRACT) Kubra Yurduseven (INTRACT)	Format review
V0.9	30/06/2022	Ing. Giuseppe D'Angelo (CRF)	Submitted version

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

Disclaimer
This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation, or both. The publication reflects the author's views. The European Commission is not liable for any use that may be made of the information contained therein.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

TABLE OF CONTENTS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

LIST OF FIGURES

LIST OF TABLES

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

EXECUTIVE SUMMARY

The current document describes the first report towards an integrated solution and the realization of the EnerMan framework. It includes the analysis of technical and implementation requirements of all modules until M18 as well as the deployment and management of necessary resources for the implementation requirements. The described work gives emphasis to the gathering of the requirements for each EnerMan module and prototype, to ensure a smooth and effective integration of the components.

Deliverable 5.1 (D5.1) provides an overview of the work that has been carried out towards the delivery of the preliminary EnerMan prototype integrated version. It includes the update of the end-to-end architecture of EnerMan components (D1.3 Preliminary EnerMan technical specifications and end-to-end architecture) into a simple, yet sustainable integrated system. Specifically, this deliverable showcases the implementation and deployment requirements of the EnerMan framework that realizes the technology convergence. It describes in detail how the individual technical elements of the EnerMan solution will be adapted and integrated under a common framework.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

GLOSSARY OF ACRONYMS

Table 1 Acronyms used throughout T5.1

Acronym	Definition
AI	Artificial Intelligence
API	Application Programming Interface
BDAE	Big Data Analytics Engine
CI/CD	Continuous Integration/Continuous Development
CLI	Command Line Interface
CPS	Cyber Physical System
CSV	Comma Separated Values
DB	Database
DevOps	Software Development (Dev) and IT operations (Ops)
DL	Deep Learning
GDPR	General Data Protection Regulation
HST	Hardware Security Token
TSDB	Timeseries Database
eVSM	Energy-aware Value Stream Modelling
iDSS	intelligent Decision Support System
IFF	If and only if
IIoT	Industrial Internet of Things
IMVS	Industrial Management Visualization System
ML	Machine Learning
I2DS	Industrial Intrusion Detection System
MPSoC	Multiprocessor system on a chip
RAM	Random Access Memory
SPE	Simulation and Predictive Engine
UI	User Interface
VF	Virtual Factory
VR	Virtual Reality
SCM	Source Code Manager
KPI	Key Performance Indicator

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

Table 2 List of EnerMan Beneficiaries and Short Names

Name	Short Name
CENTRO RICERCHE FIAT SCPA	CRF
DEPUY (IRELAND) UNLIMITED DPS	DPS
FH OO FORSCHUNGS & ENTWICKLUNGS GMBH	FH OOE
UNIVERSITA DEGLI STUDI DI NAPOLI FEDERICO II	UNINA
INFINEON TECHNOLOGIES AG	IFAG
ASAS ALUMINYUM SANAYI VETICARET ANONIM SIRKETI	ASAS
INSTITUT SUPERIEUR DE MECANIQUE DE PARIS	SUPM
MAGGIOLI SPA	MAG
AVL LIST GMBH	AVL
SPHYNX TECHNOLOGY SOLUTIONS AG	STS
EREVNITIKO PANEPISTIMIAKO INSTITOUTO TILEPIKONONIAKON SYSTIMATON	TSI
SIMPLAN AG	SIMPLAN
YIOTIS ANONIMOS EMPORIKI & VIOMIXANIKI ETAIREIA	YIOTIS
IOTAM INTERNET OF THINGS APPLICATIONS AND MULTI LAYER DEVELOPMENT LTD	ITMLCY
PRIMA ELECTRO SPA	PE
STOMANA INDUSTRY SA	STN
ATHINA-EREVNITIKO KENTRO KAINOTOMIAS STIS TECHNOLOGIES TIS PLIROFORIAS, TON EPIKOINONION KAI TIS GNOSIS	ISI
AEGIS IT RESEARCH GMBH	AEGIS
UNIVERSITY OF CYPRUS	UCY
3D NEW TECHNOLOGIES SRL	3DNT
INTRACT INOVASYON VE DANISMANLIK LIMITED SIRKETI	INTRACT
PANEPISTIMIO PATRON	UOP

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

1. INTRODUCTION

Task 5.1 (T5.1) of the EnerMan project has the objective of integrating the various components of the EnerMan framework and constructing the EnerMan prototype solution to be tested in the various pilots in Work Package (WP) 6.

According to the Grant Agreement (GA) Document, T5.1 has the following description:

"Platform integration will bring all the technologies, software components, libraries and tools provided by WP2-4 into a unified software platform. Continuous integration of new and updated solution components is supported up to a level of daily builds if necessary. Modern integration tools like Jenkins will be used. Automated integration tests will cover basic component start-up and component interoperability tests by applying simulated interfaces and test applications. In addition, this task will address the overall software documentation including the architectural documents and instructions on how to use the various components, functionalities, tools, and interfaces. There will also be sample code with instructions. Installation instructions and support for deployment will be provided by the integration engineers, developers, and architects. The EnerMan framework will be constantly evolving but during the project we will create three versions of it: partly functional v0 for the creation of a laboratory prototype, fully functional v1 (prototype version) for initial integration in the pilot platforms, and the final release EnerMan framework version of the project that will be realized in T5.3"

The main purpose of the current document is to describe a first version of the integration activities that are going to be developed by the technical partners and subsequently tested by the pilots. Moreover, it will describe a preliminary technical risk analysis, to address identified risks that have been presented throughout the first year of the project, in relation to the adoption of technical approaches and the functional operational requirements needed to procced over the next years.

The rest of the document is structured as following:

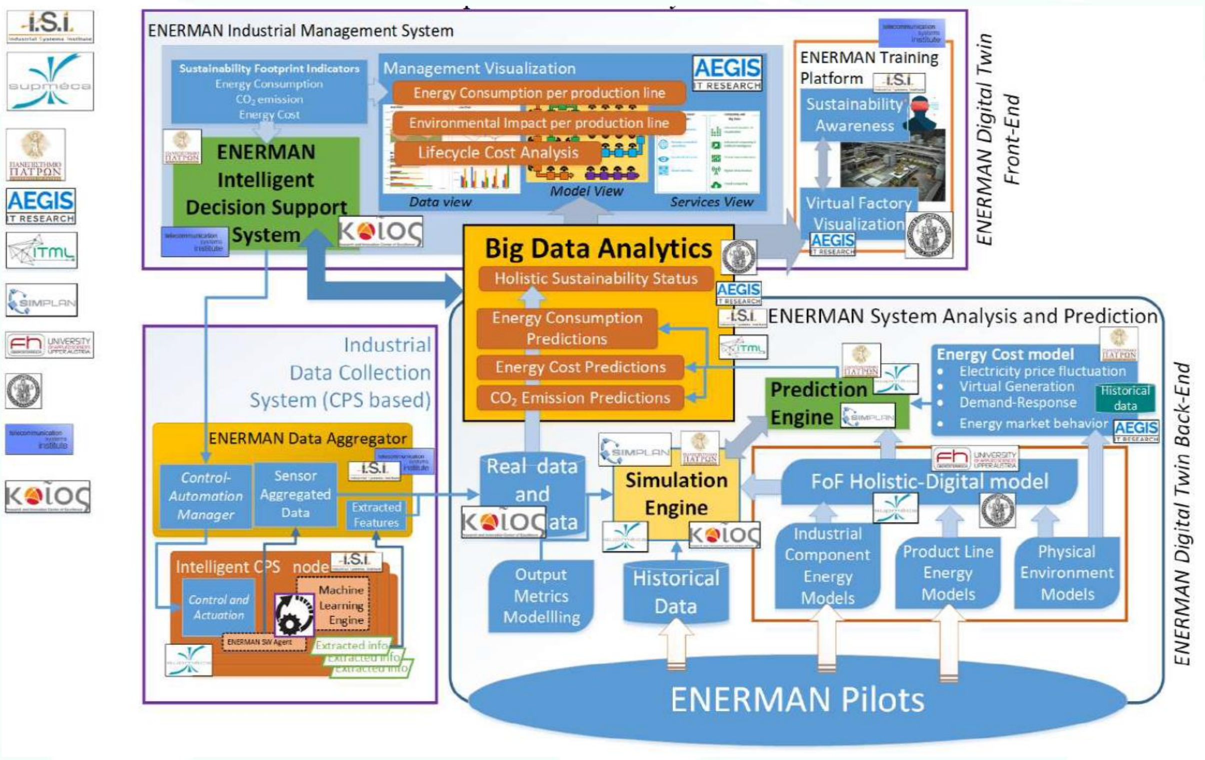
- Section 2 provides a brief overview of the EnerMan modules as well as their updated interconnection
- Section 3 presents two characteristic use cases scenarios to highlight the interaction among the EnerMan modules in an isolated and holistic manner
- Section 4 presents the suggested EnerMan integration and testing approach that should be followed in the upcoming months of the integration process.
- Section 5 concludes the current deliverable.
- Finally, the Appendix (Section 6) includes the Technical Questionnaire that was shared among the EnerMan technical partners.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

2. BRIEF OVERVIEW OF THE ENERMAN MODULES

2.1 EnerMan Modules Interconnection

The initial EnerMan concept architecture (Fig. 1) and its modules integration approach (Fig. 2) were revisited via a series of technical partners' meeting from the beginning of T5.1 (M12) to formulate an updated version of the interconnected modules (Fig. 3)



ENERMAN Industrial Management System
Sustainability Footprint Indicators
Energy Consumption
CO₂ emission
Energy Cost
Management Visualization
Energy Consumption per production line
Environmental Impact per production line
Lifecycle Cost Analysis
Data view
Model View
Services View
ENERMAN Intelligent Decision Support System
ENERMAN Training Platform
Sustainability Awareness
Virtual Factory Visualization
ENERMAN Digital Twin Front-End
Big Data Analytics
Holistic Sustainability Status
Energy Consumption Predictions
Energy Cost Predictions
CO₂ Emission Predictions
ENERMAN System Analysis and Prediction
Energy Cost model
Electricity price fluctuation
Virtual Generation
Demand-Response
Energy market behavior
Historical data
Prediction Engine
Industrial Data Collection System (CPS based)
ENERMAN Data Aggregator
Control-Automation Manager
Sensor Aggregated Data
Extracted Features
Intelligent CPS node
Control and Actuation
Machine Learning Engine
ENERMAN SW Agent
Extracted info
Real data and Data
Simulation Engine
FoF Holistic-Digital model
Industrial Component Energy Models
Product Line Energy Models
Physical Environment Models
Output Metrics Modellling
Historical Data
ENERMAN Pilots
ENERMAN Digital Twin Back-End

Figure 1 Initial Version of EnerMan Concept Architecture

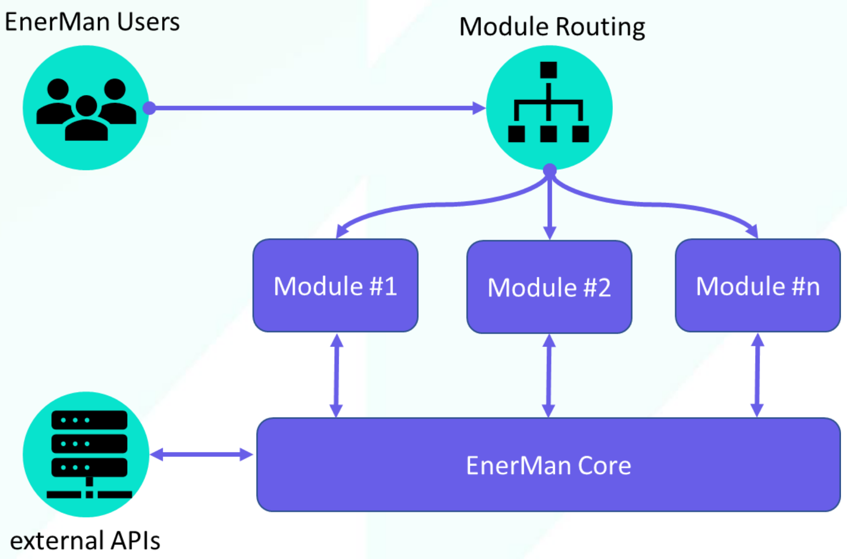


Figure 2 End-to-end view of the high-level architecture (as shown in D1.3)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

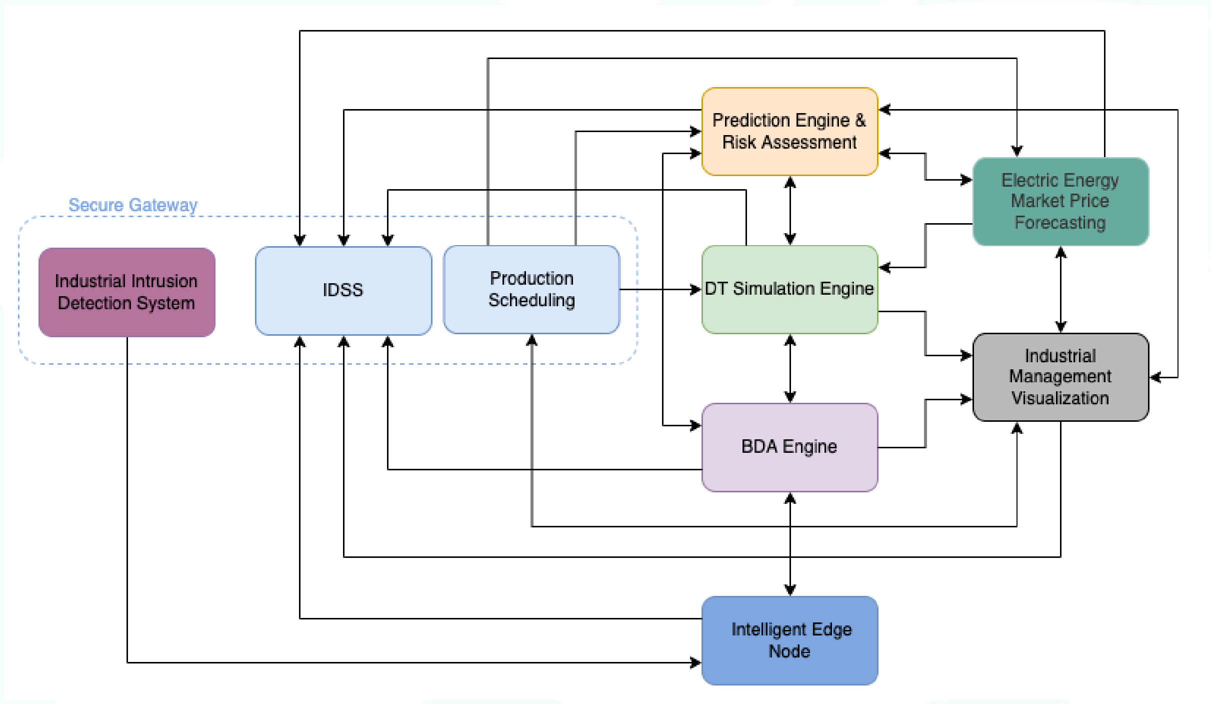

Secure Gateway
Industrial Intrusion Detection System
IDSS
Production Scheduling
Prediction Engine & Risk Assessment
DT Simulation Engine
BDA Engine
Electric Energy Market Price Forecasting
Industrial Management Visualization
Intelligent Edge Node

Figure 3 Updated Interconnection of the EnerMan Modules

2.2 EnerMan Intelligent Node

The EnerMan intelligent edge node component has the following functionalities:

- Collection of sensor data at the edge layer using the data that has been accumulated by each pilot in the pilot's industrial data collection point.
- Collection of sensor data using dedicated sensors that can be provided by the EnerMan intelligent edge node solution.
- Data harmonization of pilot sensor data and propagation of the harmonized data to the Big Data Analytics engine
- Data completion using missing data imputation autoencoder techniques
- Harmonized data Machine Learning (ML) / Deep Learning (DL) analysis to infer machine health status (when appropriate data is provided by the pilot or appropriate data can be collected by the EnerMan deployed sensor)
- Harmonized data ML/DL analysis to predict energy consumption for the industrial components linked to each edge device
- Collected thermal camera image processing to detect mean radiant temperature of industrial indoor spaces
- Security services and quantum safe secure communication (authentication and message integrity included)
- Collection of industrial processes configurations from the EnerMan intelligent Decision Support System (iDSS) and propagation to the associated industrial components

Table 3 Intelligent Node Overview

Component	**Intelligent Node**
Role (Service, Binary or Source Code)	The EnerMan intelligent edge node as a whole cannot be provided as a service since it is a physical device. Some of the node's subcomponents can be provided as source code or binary

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

Component	**Intelligent Node**
Dependencies	The EnerMan intelligent edge node is expected to interact mostly with other WP3 modules, namely: • Big Data Analytics Engine for retrieving relevant data • EnerMan Intelligent Decision Support System • VR environment of the Virtual Factory (if needed) The EnerMan intelligent edge node may also interact complementary to the WP4 components, related with energy consumption prediction
Interactions	The EnerMan intelligent edge node does not require user interaction. However, we may provide a simple interface (depending on the final EnerMan approach) that can be used in order for the user to configure some aspects of the collection, harmonization and processing performed on the EnerMan intelligent edge node. This may also include which pilot the device will be operating for as well as where the data will be enrouted (which IP address).
Hardware Requirements	The component is working on a dedicated embedded system board that acts as an edge device. The main requirements of such a board are the following: • MultiProcessors System on chip with ARM cortex A processor (preferably A53) and with FPGA fabric. • Petalinux OS Distribution (supported by Xilinx -AMD) • RAM: >=1GB • Disk space: >=16GB External requirements: • Internet connection (through wifi or ethernet networks) • BLE or Bluetooth for communication with external sensor devices (if this is needed) • Usb communication for some of attached sensors to the EnerMan intelligent edge node
Software Requirements	Petalinux OS distribution Xilinx Runtime execution environment (XRT) on board (on top of the Petalinux) In case that additional subcomponents need to be installed/setup to the EnerMan intelligent edge node, this process can be done using the edge node remote deployment and execution and/or use of docker containers. In such a scenario a remote server associated with the EnerMan intelligent edge node should be able to support the relevant EnerMan software deployment agent. The EnerMan intelligent edge node should have docker and its dependencies installed. End-users only need a browser to access the Industrial Management Visualisation.
Available API/Service	There is no full demonstration of the overall EnerMan intelligent edge node component. However, since the data aggregator includes several different technologies and subcomponents, all those subcomponents are being described and demonstrated (including several tutorials) in the WP2 deliverables (mainly D2.1 and D2.2). By the end of WP2 the code of those components will become available in a git repository. The EnerMan intelligent edge node will be able to work in a federated learning environment in order to use and update ML/DL models. As part of the security functionality instilled in the EnerMan intelligent edge node, by using the ISI Hardware Security Token (HST) module there is a Command Line Interface (CLI) and an API to access the HST functionality, when needed. The data aggregator

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

Component	**Intelligent Node**
	functionality is forwarding harmonized and/or aggregated industrial data using JSON files complying with the Big Data analytics engine APIs
Inputs [Datatype]	The EnerMan intelligent edge node is collecting data through the edge node daemon software running in each industrial pilot's site expects the following inputs (per use-case): • energy consumption sensor data per sub-process or sub-component of the industrial environment (historical, real-time -if available) • other sensor data per sub-process or sub-component of the industrial environment (inside temperature, indoor relative humidity, outside temperature, outdoor relative humidity, pressure, vibration, any other sensor data needed by a pilot) This data can be both historical and real-time (if available) • Industrial device configurations as those generated by the iDSS
Outputs [Datatype]	The EnerMan intelligent edge node generates harmonized and/or aggregated sensor data from the pilot sites as well as extracted features from the intelligent agents running in the node. The output is consumed mainly by the Big Data analytics engine. The EnerMan intelligent edge node also generates and outputs any security related information for cyberattack prevention and detection (including security certificates, detected anomalies etc.).

More details on the EnerMan Intelligent Node are provided in Deliverables D2.2 (Final Version of EnerMan Data Collection and Management Components, M18) and D2.3 (Holistic Data Processing and CPS Assisted Intelligence Report, M18).

2.3 Big Data Analytics Engine

The Big Data Analytics Engine consists of:

- the *enerman-data-harmonization module*: A python module to process the raw data collected in the edge. Dedicated harmonizers for each use-case to standardize the datasets in a unified representation and store them locally and in a timeseries database (TSDB).
- the *enerman-big-data-analytics API*: The API provides endpoints to retrieve and store data to the TSDB, data preprocessing and transformation functions, and unsupervised models.

Currently the Harmonization Module is fully developed and customized for several use cases. Future steps, involve module customization to support the rest of the EnerMan use-cases. Potential minor modifications might be required depending on the final deployment setup depending on the connection between the edge node and Big Data Analytics Engine in the cloud. Regarding the Big Data Analytics Engine, the current implementation includes the enerML module to support analytics' operations and an API to access the required data and metadata storages and the analytics results. Several endpoints for these operations are already available to be used from the rest of the EnerMan components and downstream tasks. Development will continue with more endpoints, datasets pre-processing, and analytics added as the integration plan is progressing.

Also, this component consists of R code developed for the analysis of functional data related to energy consumption profiles, with the aim to forecast the energy consumption daily profiles or to build control chart to detect unusual conditions in the manufaturing process when special causes of variation act on the energy consumption. R code can be integrated with other components through other platforms.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

Table 4 Big Data Analytics Engine

Component	**Big Data Analytics Engine**
Role (Service, Binary or Source Code)	**enerman-data-harmonization module:** source code (https://gitlab.com/i2805/enerman_edge_node/enerman_data_harmonization/-/tree/edge-dev) **BDAE:** service
Dependencies	**enerman-data-harmonization module:** Deployed in the Intelligent Node, GET/POST requests to BDAE API for accessing the data models, loads harmonized data to TSDB vis SFTP. **BDAE:** The Industrial Management Visualization System sends GET requests to BDAE API to consume data and analytics. The Intelligent Node, GET/POST requests to BDAE API for accessing the data models used for harmonization at the edge and SFTP to transfer data to the cloud TSDB. Digital Twin Simulation Engine, the Intelligent Decision Support System, and the Prediction Engine will potentially follow the same pattern to read and write data.
Interactions	**enerman-data-harmonization module:** a user can use the module as follows: python main.py -o ORGANIZATION -dmr DATA_MODEL_REPO -dp DATA_PATH -uc USE_CASE -p PROCESS -lhp LOCAL_HARMONIZED_PATH -db REMOTE_DB **BDAE:** a user can access data and analytics by providing the appropriate query parameters, either via a Swagger UI, Curl, or Request URLs. Redoc API documentation is also available. Incoming harmonized data are detected and automatically loaded to the TSDB.
Programming Languages	Python 3.8, Python libraries defined in requirements.txt
Hardware Requirements	**BDAE:** CPUs: =8 (GPUs might be needed depending on how big the training data is or if deep learning is required) RAM: >=16GB (depending on the data volume) Disk space: >=20GB (depending on the data volume)
Software Requirements	**enerman-data-harmonization module:** Linux/Windows, Python 3.8, Python libraries defined in requirements.txt **BDAE:** Linux/Windows, Docker, Python 3.8, Python libraries defined in requirements.txt, TimescaleDB, PostgreSQL, MinIO, FastAPI, MLflow
Available API/Service	**enerman-data-harmonization module:** No **BDAE:** Yes (OpenAPI, Swagger UI, Redoc)
Inputs [Datatype]	**enerman-data-harmonization module:** raw CSV files [CSV] **BDAE:** GET request with query params, POST requests [serialized model, JSON], harmonized data [CSV]
Outputs [Datatype]	**enerman-data-harmonization module:** harmonized CSV files [CSV] **BDAE:** data, predictions, analytics [CSV, JSON]

More details on the Harmonization Module are provided in D2.1 (Preliminary version of EnerMan Data Collection and Management Components, M12) and D2.2 (Final Version of EnerMan Data Collection and Management Components, M18). More details on the Big Data Analytics Engine are provided in D3.1 (Big Data Collection and Analytics Platform and Analytics Report, M18).

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

2.4 Production Scheduling

The Production Scheduling component will be part of the EnerMan Intelligent Decision Support System (iDSS) and will support a number of energy aware production models and will provide a set of mapping and scheduling algorithms that will support:

- Independent jobs that can be performed on multiple (possibly heterogenous) machines with availability and deadline information. The supported objectives will be energy cost and CO_2 minimization.
- Task graphs that will support:
  - intermediate product storage capacity constraints
  - time windows for intermediate product storage
  - intermediate product volume dependent time and energy transfer cost
- Job compatibility constraints that will support variable setup time between jobs on the same machine

Table 5 Production Scheduling

Component	**Production Scheduling**
Role (Service, Binary or Source Code)	The component will initially be provided as a service and will finally be provided as a dockerized binary component.
Dependencies	The Production Scheduling Component of the EnerMan Intelligent Decision Support System will mainly interact with the foll owing components: • EnerMan Electric Energy Market Price Forecasting for creating the evaluation function of the optimization models • EnerMan Energy Prediction for estimating the energy consumption for each task, machine and operation mode combination • EnerMan Simulation engine to validate / evaluate the generated solution • EnerMan Industrial Management Visualisation will use the component to generate a production schedule and present KPIs for energy cost and CO2 emissions
Interactions	Not directly as a platform to the users. The EnerMan GUI should provide a configuration screen to allow the users to select the objective function and set constraints. The component may be called multiple times to generate solutions to evaluate alternative production scenarios.
Hardware Requirements	CPUs: >=4 (the more the better) RAM: >=16GB (the more the better) Disk space: <=2GB
Software Requirements	When it will be offered as a service the other components should be able to access it as a RESTful web service. When a docker image will be provided, the server should have docker support and its dependencies installed.
Available API/Service	A RESTful API will be the preferred way of communicating with other EnerMan modules (Not yet available)
Inputs [Datatype]	The **Production Scheduling Component** expects the following inputs: • Job / Task related data ○ Energy consumption per job/task for each compatible machine (from T4.2/T4.4) ○ Execution time for each compatible machine and operation mode (from user input) ○ If it is an obligatory or optional job / task

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

Component	**Production Scheduling**
	○ Task execution flexibility (availability, deadline, time windows) ● Machine related data ○ Energy consumption function at different operational states (from T4.2/4.4) ○ Machine availability (planned maintenance, operating personnel etc) ○ Machine pre-allocated capacity ● Production description data ○ Feasibility of schedulable time periods (Available working days & hours) ○ Time horizon (see also Section 2.6) and related penalties if not a schedule meets makespan (the elapsed time between the start and finish of a sequence of operations in a set of machines or the completion time of the last job or task in an operation or process) requirements ○ Dependencies between jobs/tasks ○ Intermediate products storage availability and feasible time windows ○ Transfer time and energy cost to move intermediate products between machines / sites ● Energy prices per hour for the relative energy markets (from T4.4) ● Energy production mix per hour for the relative energy markets (from T4.4) It must be realized than not all inputs will be required for all use case and there may be additional requirements per use case that will require further information and extended problem modelling.
Outputs [Datatype]	● Production assignment and scheduling of tasks to machines ○ Estimated total energy consumption ○ Estimated total energy cost ○ Estimated total CO_2 emissions

More details on the Production Scheduling module will be provided in D3.4 (EnerMan Intelligent Decision Support System report, M20).

2.5 Energy-aware Value Stream Modelling (eVSM) and Simulation Engine

The energy-aware value stream modelling (eVSM) module is a generic offering for all pilots with logistic/production aspects in the use-cases covered. It is based on the SimVSM-app from SimPlan and developed together with the Simulation Engine as part of T4.3. It offers a web-based user interface (UI) where users can model their value streams (including energy aspects). Making use of the Simulation Engine (see second SimPlan contribution), these value streams can be simulated returning both logistic as well as energy-related Key Performance Indicators (KPIs). These KPIs are presented in graphical form to the user.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

Table 6 Energy-aware Value Stream Modelling and Simulation Engine

Component	**Energy-aware Value Stream Modelling and Simulation Engine**
Role (Service, Binary or Source Code)	It will most probably be offered as a service running on a SimPlan computer. Should there be a central EnerMan platform we could also install the components in a Windows VM on this platform.
Dependencies	● **Prediction engine**: predictions (such as energy prices) can be used as part of a simulation. ● **Prediction engine/iDSS**: can create/parameterize simulation runs via the Simulation Engine's API. ● **Visualization Engine**: should offer links to the eVSM. Maybe can later on visualize results from the simulation. ● **Big Data Analytics**: can act as a database for the simulation for both input data of the simulation run as well as to store simulation results.
Interactions	The eVSM app has a web-based user interface to model the value stream, create simulation runs and finally to inspect and analyse results.
Hardware Requirements	N/A (Not relevant as it will be offered as a web service/web-based application.)
Software Requirements	N/A (Windows, Plant Simulation), not really relevant, because of service-based offering.
Available API/Service	The simulation engine offers a REST/HTTP-based API to access its functionality. The eVSM application will be able to open a certain scenario using a specific URL format.
Inputs [Datatype]	For the Simulation Engine's API there will be a detailed description of the expected messages/data format offered in the next 2-3 weeks together with a test installation of the component. The eVSM app will offer a web-based UI that can be linked to using an appropriate URL. Details of the expected format will be provided together with the test installation.
Outputs [Datatype]	The Simulation Engine component offers an API to create, parameterize and run simulations. Results of these simulation runs are returned via the API. The Simulation Engine is implemented by a SimPlan middleware software called "SimController". SimController takes care of all external communications and creates/controls simulation runs as needed

More details on the eVSM and the simulation engine module are provided in D4.1 for a summary of the modelling approach itself (Energy-related flows and process energy sustainability modelling, M18). More details on the implementation are provided in D4.2 (Simulation approach/mechanism for providing energy related indicators, M18).

2.6 Prediction Engine and Risk Assessment

This component will predict energy consumption and associated costs (economic and CO_2 emissions) based on a production plan and scheduling. Due to the various uncertainties (market variability, internal disturbances, quality of data and models, etc.), it should also determine the level of confidence of the predictions and the associated risks. A prediction engine for the analysis of energy

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

consumption will be also available through R code, which can be integrated with other components through platforms such as Jenkins. In particular, R code will be provided for the statistical analysis of functional data, with the objective to forecast energy consumption daily profiles.

Table 7 Prediction Engine and Risk Assessment

Component	**Prediction Engine and Risk Assessment**
Role (Service, Binary or Source Code)	The demo version of the component will be provided as source code.
Dependencies	The **Prediction Engine and Risk Assessment** will interact with: • Pilot Holistic digital model • Industrial Data collection system • Electric Energy Market Price Forecasting • Simulation Engine • Big data Analytics • Intelligent Decision Support System • Industrial Management Visualisation
Interactions	Provision of a production schedule and the relevant production system by the end users. This assumes that the data/information from the production system is integrated into the EnerMan system
Hardware Requirements	The size of the data and the number of requests not yet well established. We cannot yet provide these specifications
Software Requirements	The development of the final version of our component will probably be done by another partner in the project. For the demo version, we do not anticipate any constraints.
Available API/Service	No
Inputs [Datatype]	The **Prediction Engine and Risk Assessment** needs the following inputs: • Pilot use-case information: production lines and processes description, correlation: weather-equipment's consumption, Production plan/schedule for a specific time horizon • Available energy information: mix, cost, correlation: Energy-CO2, etc
Outputs [Datatype]	The **Prediction Engine and Risk Assessment** will provide the following outputs: • Energy consumption, cost and CO2 predictions for a specific time horizon • Decision-making recommendations

More details on the prediction engine and risk assessment module will be provided in D4.3 (Computational prediction engine report, M24).

2.7 Intelligent Decision Support System

The EnerMan intelligent Decision Support System (iDSS) assesses the compliance of the predictions, with predetermined, by the pilot operators, energy sustainability indicators using an innovative event-based decision engine. In case of non-compliance, the i-DSS using an ontology-based actuator models picks on-the-fly the appropriate control algorithm to be used and from a library of configuration measures chooses the appropriate ones that may achieve compliance. This is forwarded to the EnerMan prediction engine that then predicts the energy sustainability outcome of such configurations. The EnerMan iDSS consists of:

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

- Database: An SQL based storage space that will store process target models and configuration analysis results.
- API: A RESTful Application Programming Interface (API) that can be accessed by other software solutions to get or post data.
- Backend: A backend platform used to process data input manually by the user, or data that is received by other software solutions using the iDSS API.
- Frontend: A web-service interface that can be used by the user to view data or input data

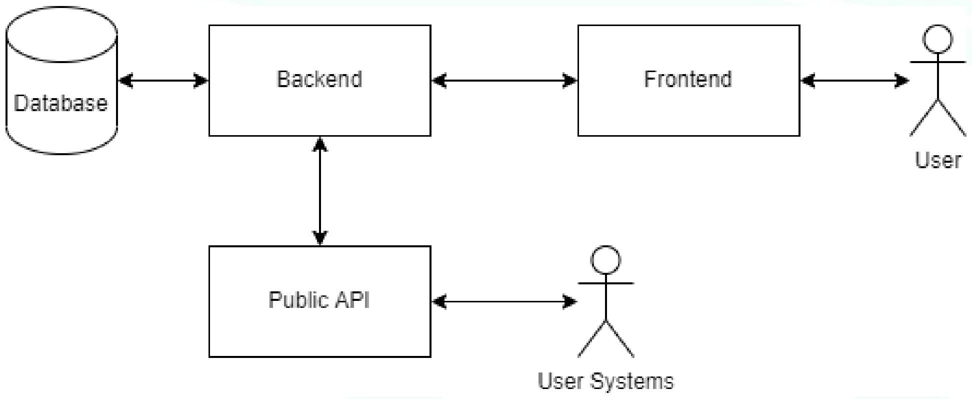


Figure 4 EnerMan Intelligent Decision Support System Overview

Table 8 Intelligent Decision Support System

Component	**Intelligent Decision Support System**
Role (Service, Binary or Source Code)	The EnerMan DSS will be provided as a service, without excluding the option to be provided as source code.
Dependencies	• **Digital Twin**: Different control-loop configurations can be simulated on the digital twin to produce energy analytics. • **Big Data Analytics**: Historic data analysis to tank the different control-loop configurations.
Interactions	• Knowledge Model: A knowledge model that describes the target process is given as input to the iDSS. The knowledge model can be developed prior to the deployment of the iDSS, by the industrial unit engineer, or in collaboration with the KIOS experts developing the DSS. The model can be changed by adding/removing components following the changes of the target process and its control components. • Queries: A query in a simple query language. The query requests from the iDSS reasoning module to return a list of control loop configurations that control a set of physical properties.
Hardware Requirements	For 25 concurrent users: (a) 2GB of RAM, (b) 2 CPU Cores, (c) 1GB of storage
Software Requirements	OS to support Docker containers. NGINX web server, Gunicorn WSGI
Available API/Service	it will include OpenAPI Specification (OAS) Version 3 and a Swagger page
Inputs [Datatype]	[JSON]
Outputs [Datatype]	[JSON, Prolog, PNG]

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

More details on the prediction engine and risk assessment module will be provided in D3.4 (EnerMan Intelligent Decision Support System report, M20).

2.8 Industrial Management Visualization System

The Industrial Management Visualisation System will offer three types of functionality:

- The "service view" allowing the user to launch the desired EnerMan service, e.g., monitor the energy consumption of a certain industrial process, run "what-if" scenarios for rescheduling of production, etc.
- The "data view" visualising the energy consumption, energy cost, environmental impact and internal/external environmental conditions for different granularity levels (e.g., both "target" subprocesses or for individual equipment) and for different time periods
- The "model view" presenting the status of the target process and configurations of machines as well as providing alerts for the chosen device, e.g., if energy drifts are detected.

The three-tier architecture of Industrial Management Visualisation appears in Figure 5 and includes the following:

- In terms of data storage capabilities, the Industrial Management Visualisation can operate with its own storage facility (mainly Elasticsearch search engine), but as already mentioned it can interoperate with any other state-of-the-art data source
- The middleware application is built on Node.js for transforming the data and performing any business logic activities.
- The frontend is based on Angular.io framework in conjunction with a set of visualisation libraries.

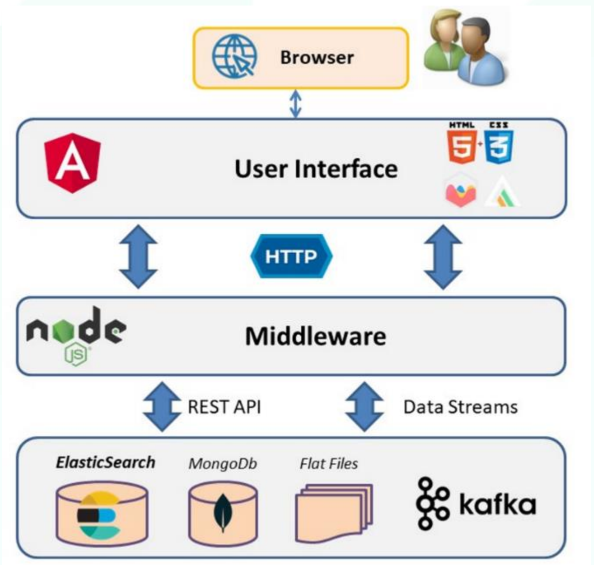


Figure 5 The Industrial Management Visualisation Technology Stack

Table 9 Industrial Management Visualization System

Component	**Industrial Management Visualization System**
Role (Service, Binary or Source Code)	A dockerised version of the Industrial Management Visualisation is the preferred option, but it can be also provided as a service.
Dependencies	The Industrial Management Visualisation is expected to interact mostly with other WP3 modules, namely:

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 958478

Component	**Industrial Management Visualization System**
	● Big Data Analytics Engine for retrieving relevant data ● EnerMan Intelligent Decision Support System The Industrial Management Visualisation might also interact with WP4 components, e.g., Electric Energy Market Price Forecasting (if the latter does not store its outputs to the Big Data Analytics Engine). Furthermore, it can allow the user to access the Energy-aware value stream modelling (eVSM) from SIMPLAN.
Interactions	Yes, the Industrial Management Visualisation is the main user interface to the EnerMan services. The user will be able to: ● Navigate to the “target” subprocess (e.g., paint shop degreasing tank) and choose the desired EnerMan service ● See information on various energy sustainability KPIs (e.g., energy consumption, energy cost, environmental footprint), external data streams (e.g., historical energy prices), etc. and interact with the visualisations (e.g., focus on a selected KPIs, choose a certain time window, etc.) in order to get situational awareness. ● See alerts on anomalous system states, initiate a “what-if” analysis, see simulation outputs and choose the most appropriate mitigation action or approve the proposed one.
Hardware Requirements	CPUs: >=2 RAM: >=4GB (depending on the data volume and utilised visualisations) Disk space: >=20GB (depending on the data volume and utilised visualisations)
Software Requirements	Assuming that a docker image will be provided, the server should have docker and its dependencies installed. End-users only need a browser to access the Industrial Management Visualisation.
Available API/Service	No, it is not expected to offer any APIs to other EnerMan modules. The Industrial Management Visualisation is only consuming APIs.
Inputs [Datatype]	The Industrial Management Visualisation expects the following inputs (per use-case): ● energy consumption data per sub-process or sub-component ○ historical ○ real-time (if available) ○ predictions (from WP4) ● manufacturing production data (historical, real-time [if available to be extracted] and predictions) ○ historical ○ real-time (if available) ○ planned ● energy prices per sub-process or sub-component (for the ones that depend on wholesale electricity prices) ○ historical ○ real-time (if available) ○ predictions (from WP4) ● environmental data (historical) ○ inside temperature, relative humidity, etc. ○ outside temperature, relative humidity, etc.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement **No 958478**

Component	**Industrial Management Visualization System**
	▪ UoP uses openweathermap ▪ this info is critical for controlling HVACs etc ● energy mix data o historical o real-time (if available) ▪ not actual data o predictions (from WP4) ● alerts (if available) ● machine configurations (if available)
Outputs [Datatype]	The Industrial Management Visualisation does not generate any outputs to other EnerMan modules, but it can collect user inputs that are useful for other technologies, especially WP2, such as: • Preferred option to be executed (when the user is presented with the simulation outputs of several alternative "what-if" scenarios) • Selected machine configurations (i.e., in case of manual configuration)

More details on the Industrial Management Visualisation System module are provided in D3.2 (EnerMan Visualization and Management Framework Design, M18).

2.9 Secure Gateway

The secure gateway consists of API Gateways enforce policies which control security aspects such as the authentication, authorization or traffic management. The API Gateway is comparable to a gatekeeper guarding the underlying data. The following figure shows an example for the secure Gateway as exploited for API calls from/to data aggregator

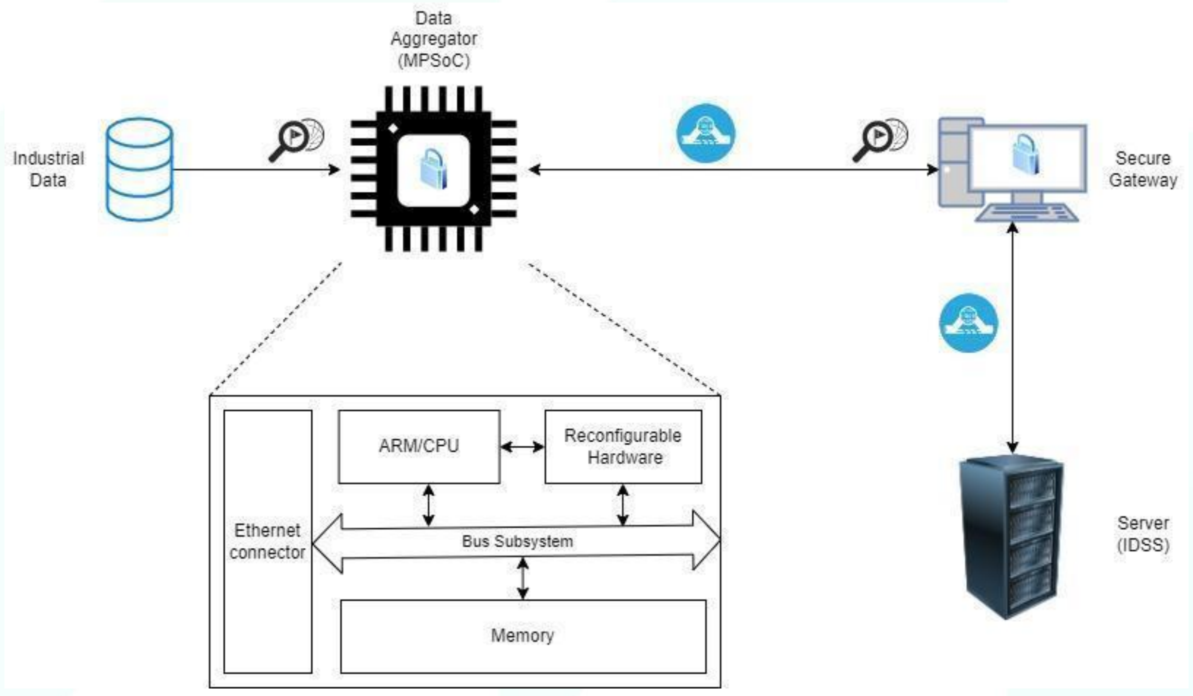


Figure 6 Secure Gateway

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

Table 10 Secure Gateway

Component	Secure Gateway
Role (Service, Binary or Source Code)	It will be provided as a docker image or as a VM image along with its configuration that will define the enforced policy
Dependencies	Should be utilized by all tools that have APIs in order to provide secure exposure of those APIs
Interactions	No
Programming Languages	The GW will based on opensource tools, tentative are WSO2 https://wso2.com/api-manager/ ang Kpng https://konghq.com/kong
Hardware Requirements	Expected to need: 2 cores 4GB RAM, can be Dockerized
Software Requirements	A VM with a Linux kernel
Available API/Service	The API Gateway is forseen to expose EnerMan's API based on policies, especially in the case that API calls are executed over crossdomains
Inputs [Datatype]	N/A
Outputs [Datatype]	N/A

2.10 Industrial Intrusion Detection System

The Industrial Intrusion Detection System (I2DS) is the EnerMan intrusion detection mechanism that will be deployed across the system and at its very edges, i.e., it is the ML-IDSS mechanism that will operate on the data coming in from the architecture's edge devices. I2DS is going to be hosted by the EnerMan MPSoC (Multiprocessor system on a chip) devices, which are devices that employ FPGA (Field-programmable gate array) technology. Hence, this first layer of intrusion detection capability is going to be implemented directly on (reconfigurable) hardware.

Table 11 Industrial Intrusion Detection System

Component	Industrial Intrusion Detection System
Role (Service, Binary or Source Code)	
Dependencies	The I2DS will mainly interact with WP2 components: ● Data security token ● Security Gateway
Interactions	No
Programming Languages	Python, C++ and VHDL/Verilog
Hardware Requirements	ZCU104 (MPSoC) with ARM processors and FPGA fabric.
Software Requirements	Pynq image on the SD card.
Available API/Service	No

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

Inputs [Datatype]	N/A
Outputs [Datatype]	The main output will be a bitfile that will be downloaded on the MPSoC [bitfile]

2.11 Electric Energy Price Forecasting

The Electric Energy Market Price Forecasting service will be part of the EnerMan Prediction Engine and will initially offer three types of predicted values:

- The "day-ahead cost" function that will allow other EnerMan components to query about the estimated cost for the next day time horizon for a supported energy market.
- The "day-ahead energy mix" function that will allow other EnerMan components to query about the estimated percentage of renewable energy and related CO_2 emissions for the next day 24 hours time horizon for a supported energy market.
- The "time horizon cost" function that will allow other EnerMan components to query about the electric energy cost for the specified time horizon for a supported energy market. If the time horizon is in the past, it will return realized prices, while if the time horizon is in the future it will return a rough estimate of the energy cost per hour.

The **Electric Energy Market Price Forecasting** will be available as a RESTful service to other EnerMan modules, we estimate that will be relatively easy to integrate with other modules.

Table 12 Electric Energy Price Forecasting

Component	**Electric Energy Price Forecasting**
Role (Service, Binary or Source Code)	The component will initially be provided as a service as in the current implementation uses a commercial software component to augment the quality of the results. We plan to remove the dependency to the commercial software and provide a dockerised open source version by the end of the project.
Dependencies	The **Electric Energy Market Price Forecasting** will be part of the Prediction Engine and will provide predictions of the energy cost and the energy mix that will be realized in a specific energy market that the industry is operating and sources the energy that will be used in the production process. All EnerMan components may use this information (horizontal service) but we expect the main components that will use it will be: • EnerMan Intelligent Decision Support System for creating the evaluation function of the optimization model • EnerMan Simulation engine to translate energy consumption to energy cost • EnerMan Industrial Management Visualisation to present KPIs for energy cost and CO_2 emissions
Interactions	No, the Electric Energy Market Price Forecasting service will be used by other EnerMan components to help them translate energy consumption to energy cost.
Hardware Requirements	CPUs: >=1 (the more the better) RAM: >=4GB Disk space: <=2GB
Software Requirements	When it will be offered as a service the other components should be able to access it as a RESTful web service. When a docker image will be provided, the server should have docker support and its dependencies installed.
Available API/Service	A RESTful API will be the preferred way of communicating with other EnerMan modules.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

Component	**Electric Energy Price Forecasting**
Inputs [Datatype]	The **Electric Energy Market Price Forecasting** service expects the following inputs: ● Electric energy production data per energy market supported ○ Day ahead historical predictions ○ Realized historical production ○ Connected energy markets predictions if available ○ Published energy production estimations ● Electric energy demand data per energy market supported ○ Day ahead historical predictions ○ Realized historical consumption ○ Up and Down Reserves ○ Connected energy markets predictions if available ○ Published energy demand by energy market operator ● System Marginal Prices per energy market supported ○ Day ahead historical predictions ○ Realized historical SMP ○ Connected energy markets SMP if available ● Weather data ○ Temperature, wind speed, solar radiation etc. ○ outside temperature, relative humidity, etc. ● Misc data ○ Holidays, working days, weekdays, year period All input will be sourced from publicly available data sources.
Outputs [Datatype]	The **Electric Energy Market Price Forecasting** service will provide the following outputs: ● Energy prices per hour for the supported energy markets ○ historical ○ predictions ▪ Day ahead with high confidence ▪ Time horizon with variable confidence ● Energy production mix per hour for the supported energy markets ○ historical ○ predictions ▪ Day ahead with high confidence ▪ Time horizon with variable confidence

More details on the Electric Energy Price Forecasting module will be provided in D4.3 (Computational prediction engine report, M24).

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

3. USE CASES SCENARIOS

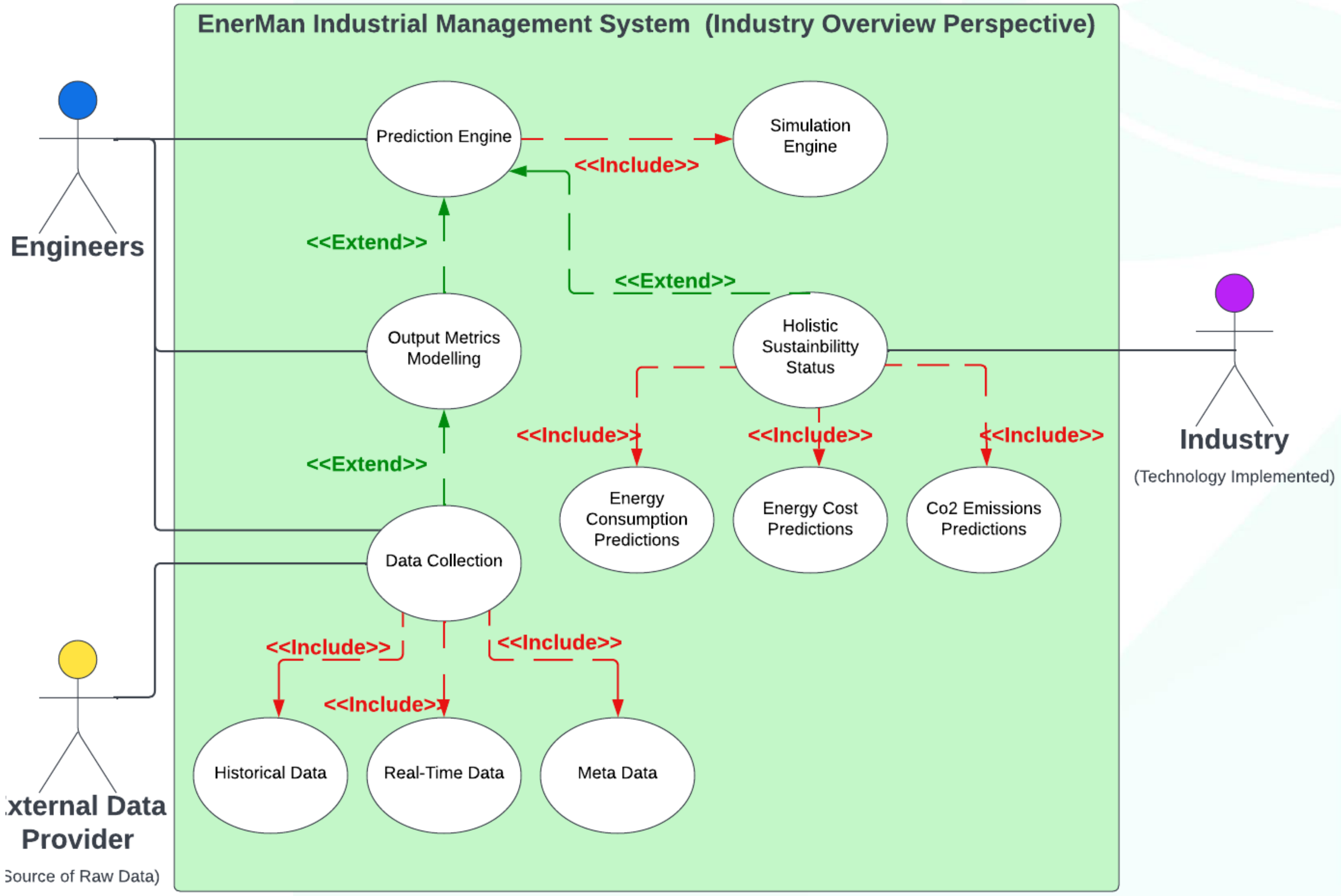


Figure 7 EnerMan Industrial Management System

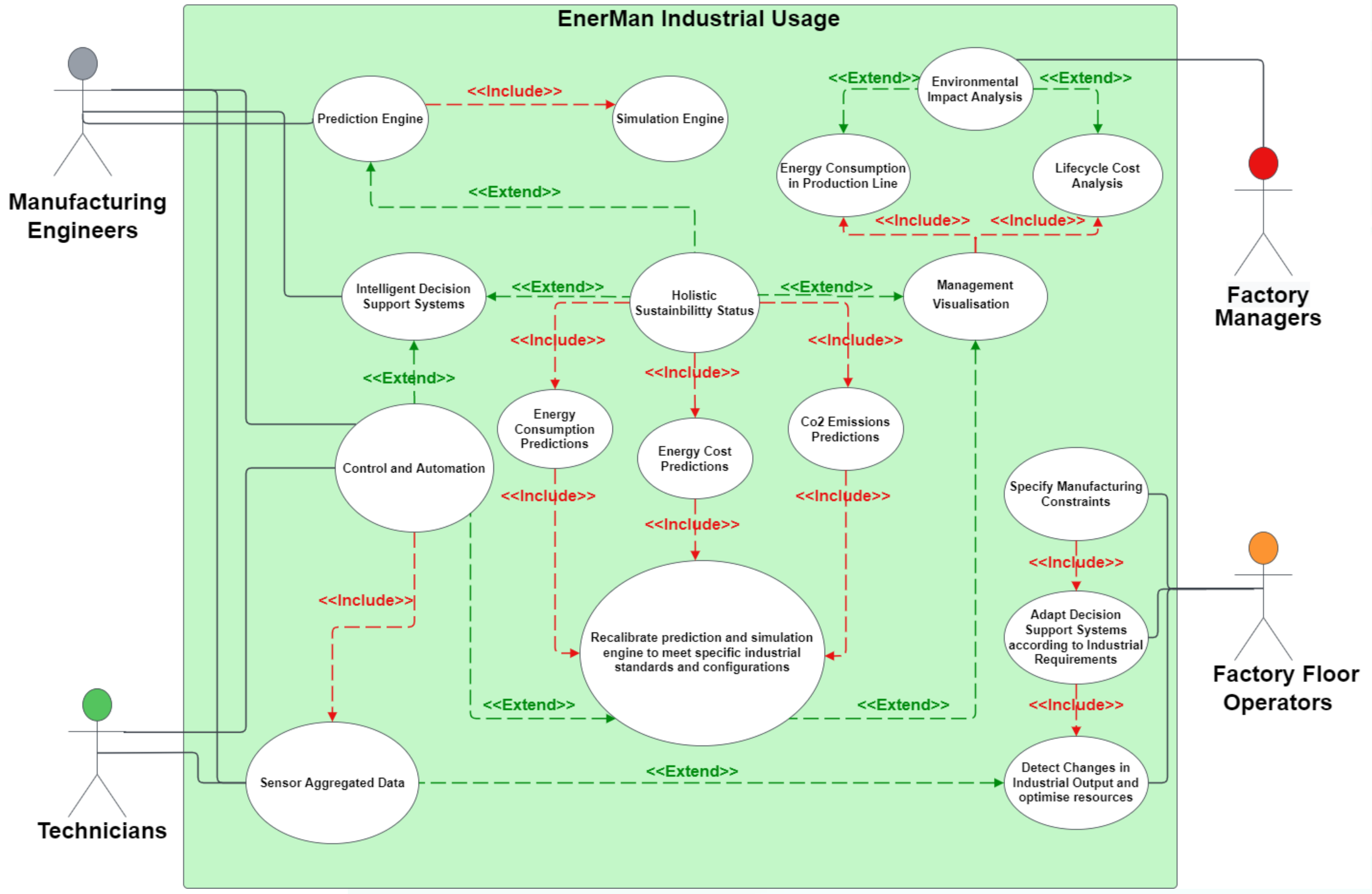


Figure 8 EnerMan Industrial Usage

To further investigate the interconnection among models, we hereby present two use cases scenarios (linking the WP5 integration aspects with WP1 activities and especially the ongoing work towards D1.4 “Final EnerMan technical specifications and end-to-end architecture, to be submitted on M24), one focusing on specific elements of the EnerMan system and the other providing a holistic aspect of the EnerMan platform.

The ‘EnerMan Industrial Management Overview’ Use Case in Figure 7 provides a generic idea of the flow of raw data from the industry site to be crunched by the EnerMan prediction/simulation engine modules (Sections 2.5 and 2.6 of the current document) and the solution being used by the industries in their day-to-day activities. Triggering effects occur in terms of data flow and the information perceived by the actors/subjects (namely **Engineers** and **External Data Provider actors**). The industry acts as the source of raw data in the form of Historical data, Real-Time Data and Meta Data being collected to be processed by the Output Metrics Model designed for the prediction engine and simulation engine. A holistic sustainability status report is generated which is in-turn being used by the industry under the EnerMan Industrial Management System so as to predict the energy consumptions, cost of the energy and CO_2 emissions. It is a circular industrial case where the raw data generated from the industry is processed, compiled, and run through the prediction and simulation engines to implement the resulting technological solution back into the industry. This general overview has been elaborated further in the next EnerMan industrial use case where the actors using the EnerMan system within a particular industry have been elucidated under their specific usages within their job profiles.

The ‘EnerMan Industrial Usage’ Use Case provides information regarding how the final users at factory/industry level shall use the EnerMan system. In Figure 8, four actors/subjects are taken into consideration which are **Factory Managers**, **Manufacturing Engineers**, **Technicians** and **Factory Floor Operators**. Overall activities cumulate towards achieving the core objectives of EnerMan which are energy consumption predictions, energy cost predictions and CO_2 emissions predictions, and how these holistic sustainability status reports are instrumental to be used by the industry at ground level. Here each actor has their own specific functionalities and usage of the EnerMan system, however all the actors are intertwined at some levels so as to enhance the collaborative potential and synergies within the industry and manufacturing/production line.

A Factory Floor Operator has the responsibility to specify the manufacturing constraints or anomalies that occured during the industrial processes and then report them to the system so as to adapt the events in the further industrial processes. By extension of these usages, the Industry Technician is also responsible for detecting the changes in the industrial output and optimise the resources according to the changes observed in the sensor aggregated data. The Manufacturing Engineers in the backend are responsible for smooth running of the prediction engine and Intelligent decision support systems with the regular data feedback and update the systems after running the simulation engine. The Manufacturing engineers who are responsible for the control and automation, by the help of Intelligent Decision Support Systems and sensor data can provide recalibrated directions to the prediction and simulation engine so as to meet the specific industrial standards and configurations.

The Holistic Sustainability Status includes the data reports for energy consumption predictions, energy cost predictions and CO_2 emissions predictions which help the Manufacturing engineers and Factory Managers in decision making if a machine component or production line needs to be revamped or sustained according to the green energy status the industry aims to achieve. This comes under management visualisation where the actors that are manufacturing engineers and factory managers use the system to reinstate the environmental impact analysis where the energy consumption data in the production line is observed and subsequently the lifecycle cost analysis is performed, to have an overview over the industrial energy consumption rates. The decrease in energy costs and consumption

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement **No 958478**

as well as increase in lifecycle of the machinery combinedly benefits the industry in the long run. The industry can benefit from these applications of the EnerMan system to understand the production line or the manufacturing unit which needs to be overhauled to meet the green objectives and increase the industrial machinery efficiency to realise its maximum potential.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

4. INTEGRATION ASPECTS

4.1 Adoption of a DevOps Culture in EnerMan

Running EnerMan integration with a DevOps culture is all about adopting the right culture to allow developers and the operations team to work together. A DevOps culture advocates the implementation of several engineering best practices, by relying on several tools and technologies, that will be briefly described hereinafter.

A list of initial DevOps Tools (applications that help automate the software development process) identified for the purposes of the EnerMan framework are hereinafter presented in Table 13. These tools mainly focus on communication and collaboration between product management, software development, and operations professionals. DevOps tools also enablesteams to automate most of the software development processes like build, conflict management, dependency management, deployment, etc. and helps reduce manual efforts.

Table 13 Initial List of DevOps Tools Identified

DevOps Tool	Key Features
QuerySurge Link: https://www.querysurge.com/	Robust API with 60+ calls Seamlessly integrates into the DevOps pipeline for continuous testing Verifies large amounts of data quickly Validates difficult transformation rules between multiple source and target systems Detects requirements and code changes, updates tests accordingly and alerts team members of said changes Provides detailed data intelligence & data analytics
Jenkins Link: https://www.jenkins.io/	It increases the scale of automation Jenkins requires little maintenance and has built-in GUI tool for easy updates. It offers 400 plugins to support building and testing virtually any project. It is Java-based program ready to run with Operating systems like Windows, Mac OS X, and UNIX It supports continuous integration and continuous delivery It can easily set up and configured via web interface It can distribute tasks across multiple machines thereby increasing concurrency.
Prometheus Link: https://prometheus.io/	Flexible query language for slicing collected time series data to generate tables, graphs, and alerts Stores time series, streams of timestamped values belonging to the same metric, and the same set of labeled dimensions Stores time series in memory and also on local disk It has easy-to-implement custom libraries Alert manager handles notifications and silencing
Ganglia Link: http://ganglia.info/	Free and open source tool Scalable monitoring system based on a hierarchical design Achieves low per-node overheads for high concurrency It can handle clusters with 2,000 nodes

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

Snort Link: https://www.snort.org /	Performs protocol analysis and content searching It allows signature-based detection of attacks by analyzing packets It offers real-time traffic analysis and packet logging Detects buffer overflows, stealth port scans, and OS fingerprinting attempts, etc.
Docker Link: https://www.docker.com/	CaaS Ready platform running with built in orchestration Flexible image management with a private registry to store, manage images and configure image caches Isolates apps in containers to eliminate conflicts for enhancing security

Key characteristics of a DevOps culture

A DevOps culture relies on a certain number of principles. These principles are to source control (version control) everything, automate whatever is possible, and measure everything.

4.1.1. *Source control everything*

Revision control software has been around for many decades now, but too often, only the product code is checked. When practicing DevOps, not only is the application code checked, but configurations, tests, documentation, and all the infrastructure automation needed to deploy the application in all environments, are also checked. Everything goes through the regular review process by the **Source Code Manager** (**SCM**). Within the EnerMan project, this role goes to MAG (as WP5 leader). A GitLab repository has already been set up, to provide the necessary infrastructure environment for integration purposes.

4.1.2. *Automating testing*

Automated software testing predates the history of DevOps, but it is a good starting point. Too often, developers focus on implementing features and forget to add a test to their code. In a DevOps environment, developers are responsible for adding proper testing to their code. QA teams can still exist; however, like other engineering teams, they work on building automation around testing. There are four levels of testing automation to focus on, in order to successfully implement DevOps:

1. **Unit testing**: This is to test the functionality of each code block and function.
2. **Integration testing**: This is to make sure that services and components work together.
3. **User interface testing**: This is often the most challenging component to successfully implement.
4. **System testing**: This is end-to-end testing.

4.1.3. *Automating infrastructure provisioning and configuration*

Managing infrastructure on an ad-hoc basis, as was once possible, is very error-prone. In a DevOps culture, the provisioning and configuration of servers, networks, and services in general, are performed through automation. Configuration management is often what the DevOps movement is known for. However, this is just a small piece of a big puzzle.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

4.1.4. *Automating deployment*

It is easier to write software in small chunks and deploy the new chunks as soon as possible, to make sure that they are working. To get there, projects practicing DevOps rely on continuous integration and continuous deployment pipelines. Whenever a new chunk of code is ready, the continuous integration pipeline kicks off. Through an automated testing system, the new code is run through all the relevant, available tests. If the new code shows no obvious regression, it is considered valid and can be merged to the main code base. At that point, without further involvement from the developer, a new version of the service (or application) that includes those new changes will be created and handed off to a system called a **continuous deployment system**. The continuous deployment system will take the new builds and automatically deploy them to the different environments that are available. Depending on the complexity of the deployment pipeline, this might include a staging environment, an integration environment, and sometimes, a pre-production environment. Ultimately, if everything goes as planned (without any manual intervention), this new build will get deployed to production.

One aspect about practicing continuous integration and continuous deployment that often gets misunderstood is that new features do not have to be accessible to users as soon as they are developed. In this paradigm, developers heavily rely on feature flagging and dark launches. Essentially, whenever someone develops new code and wants to hide it from the end users, they set a flag in the service configuration to describe who gets access to the new feature, and how. At the engineering level, by dark launching a new feature this way, they can send production traffic to the service, but hide it from the UI, to see the impact it has on the database or on performance, for example. At the product level, this can be used to decide to enable the new feature for only a small percentage of the intended EnerMan users, to see if the new feature is working correctly and if the users who have access to the new feature are more engaged than the control group, for example.

4.1.5. *Measuring everything*

Measuring everything is the last major principle that DevOps-driven companies adopt. DevOps is an ever-evolving process and methodology that feeds off those metrics to assess and improve the overall quality of the product and the team working on it. From a tooling and operating standpoint, the following are some of the metrics most organizations look at:

- How many builds are pushed to production a day?
- How often you need to roll back production in your production environment (this is indicated when your testing didn't catch an important issue)
- The percentage of code coverage
- The frequency of alerts resulting in paging the on-call engineers for immediate attention
- The frequency of outages- Application performance
- The **Mean Time to Resolution** (**MTTR**), which is the speed at which an outage or a performance issue can be fixed

At the organizational level, it is also interesting to measure the impact of shifting to a DevOps culture. While this is a lot harder to measure, in the EnerMan we will try to also consider the following points:

- The amount of collaboration across teams
- Team autonomy
- Cross-functional work and team efforts
- Fluidity in the product
- How often Dev and Ops communicate

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

- Attitudes towards automation
- Obsession with metrics

Having a DevOps culture means, first, changing the traditional mindset that developers and operations are two separate silos, and making the teams collaborate more, during all phases of the software development life cycle.

In addition to a new mindset, DevOps culture requires a specific set of tools geared toward automation, deployment, and monitoring

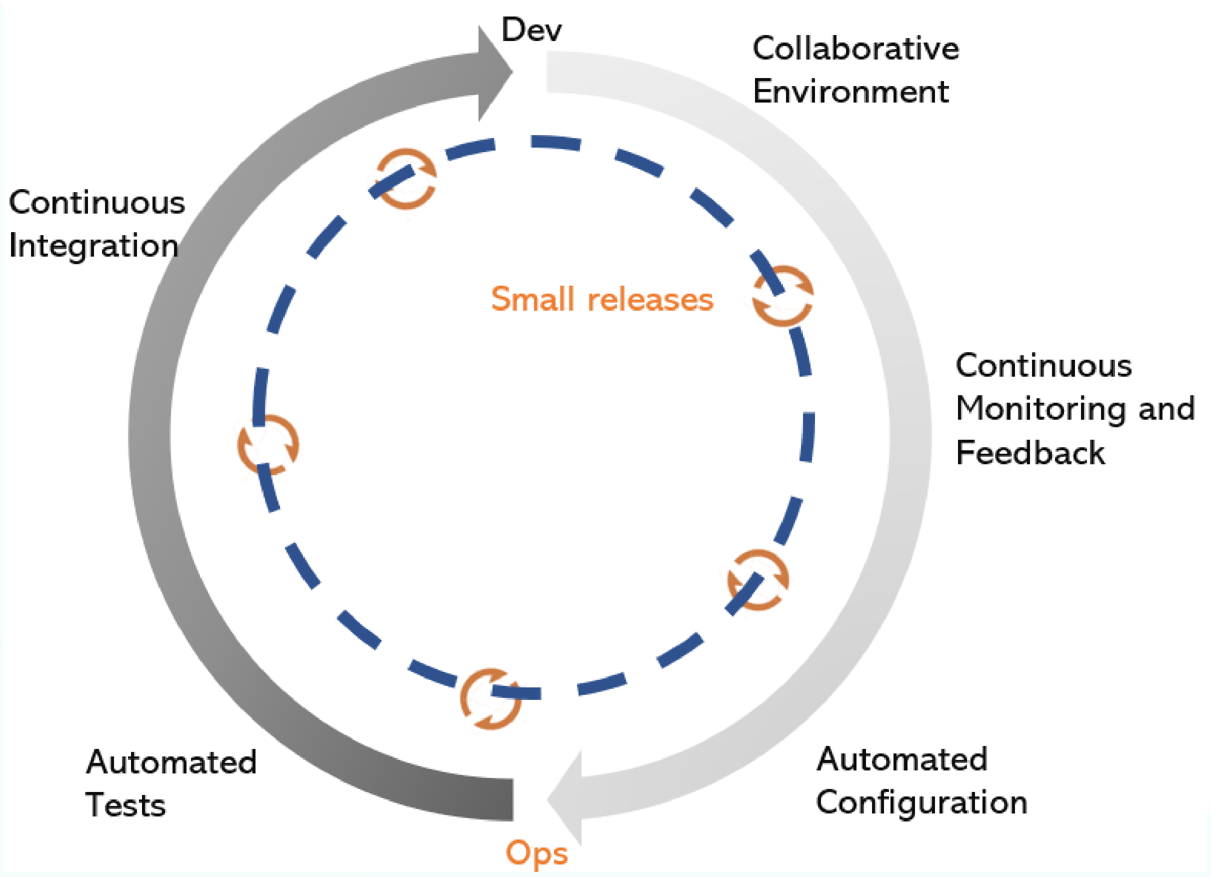


Figure 9 The DevOps Integration Approach Principles

4.2 Integration testing roadmap

This section serves as a roadmap related to the pre-release processes involved for the EnerMan modules. The following steps are defined in order to ensure a smooth integration among all key components and facilitate a regular and up-to-date upgrade/renewal of involved technical functionalities.

i. For each updated EnerMan component, local unit test(s) to be performed should be defined by each EnerMan component owner
    a. A Local Unit Test refers to a predefined test set, which must be performed automatically if possible
    b. For each component, to-do actions include:
        i. Checking of automatic test feasibility
        ii. Definition of eligible tools, constraints, etc.
        iii. Definition of a test set, to be updated when component major releases occur
ii. For each updated (or new) functionality, test on such functionality should be performed (integrating all needed EnerMan components) mostly on a manual manner

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

        a. For each case, definition of a use case test or extension of an existing one, if already drafted should be performed,

iii. For a predefined set of «main» use cases, non-regression tests should be (automatically) performed

        a. **For a predefined set of «main» use cases** non-regression tests are (automatically) performed by:

            i. Defining a short set of main (vital) use cases, including «blocking» functionalities EnerMan services cannot 'live' without

            ii. a first use case test should be in definition for all components

4.3 Automated integration tests and pilots' phase (WP6)

The idea is to use a pipeline phase/job on an environment (e.g., GitLab's CI/CD functionality). This means, after deployment of the components, and waiting to reload, to perform a series of query operations against the test environment (DB/APIs) to confirm the expected results. The how and when to do it depends totally on the implemented infrastructure (access, ports, etc.) and the deployment details of each component and the API manager, where the rest of components should also be migrated (the whole EnerMan platform), completely deployed and working.

For the pilots' phase, there is a suggestion of using a Feedback Space, by identifying the number of users who are going to be using/testing the EnerMan system per pilot, based on some kind of centralized internal process for reporting issues in each pilot site. The Feedback Space can be deployed e.g., by deploying a front-end, which can be used to push feedback, issue and support inputs (e.g., into a Trello board or a ClickUp workspace, where such inputs are then managed through related cards and can be integrated with GitLab).

The following technical questions will be also assessed through:

- What is the critical/important functionality to be tested? (Note: 1st version of the test plan will define important use cases)
- Critical/important (for the kick-off of the Pilots) functionality has been deployed within the first version of the integrated EnerMan platform?
- Did all the integration tests succeed?
- Did all the baseline security/privacy assessments pass without identifying potential vulnerabilities?
- Did usability/effectiveness evaluation pass? If not, did the development cycle incorporate changes based on those usability test results (e.g., issues identified, corrections to the design needed)?
- Does any of the issues identified require change(s) to the EnerMan Architecture?
- Repeat all the above for the final version.

Based on the critical test use cases to be defined, a set of actions will be associated with trying to emulate user behaviour by using realist and varied inputs (i.e., dashboard/module/app utilisation), or machine2machine tasks performed in the background (e.g., usage data/notifications/trained models' transmissions, notifications generation, data backups). The description of each test use case should provide a detailed description, all basic actions needed to be triggered, and expected results based on given input. Each case will be described by using the following test use case template:

- Test Use case ID: unique identifier of each test
- Test Use case Name: user friendly name
- Objective: short description of the EnerMan functionalities to be tested
- Prerequisite Tests: identifier(s) of test(s) that must have been previously and successfully executed
- Involved Components: EnerMan components and or module/app functionality associated with
- Procedure: basic steps to be performed
- Input: If and only if (IFF) values to be used

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

- Expected Result: anticipated successful result
- Outcome: (S)uccess/(F)ailure/(PS)Partially successful

Following this, four (4) phases are planned:

- **Phase I**: Generic identification of tasks (e.g., use cases, baseline analytics considering synthetic data, (semi-)automated testing routines) and KPIs for the evaluation of the under-development functionality;
- **Phase II**: Early usability and functional evaluation of interaction elements and implemented functionality based on synthetic and/or real data (prototype version), that will allow the consortium to consider this early feedback, in order to repair/adjust/enhance technical elements accordingly. Several KPIs will be evaluated (including: definition, creation and execution of basic descriptive analytics)
- **Phase III**: Usability and functional evaluation involving end-users of the EnerMan platform to provide an assessment on whether the first implemented version (v1: first version of the integrated EnerMan platform) fulfils the objectives of the project, effectiveness, impact and sustainability. Main technical KPIs will be evaluated
- **Phase IV**: Assessing the full list of KPIs to provide an assessment on whether the final version of the integrated EnerMan platform fulfils the objectives of the project, effectiveness, impact and sustainability

4.4 Process Modelling

Following the experience gained form other EU projects, and in particular FACTLOG[1], MAG being the WP5 leader will adopt in the context of EnerMan, the **Process Modelling** approach. Process Modelling denotes a generic approach with all related methods, algorithms, mechanisms, services and tools it directly uses, integrated into an overall modelling application or platform. In any specialized use case, these methods, algorithms, and mechanisms do not change. It is just the Model (see below) representation per se that changes.

In the same context, a **Model** is the representation of an industrial production system using either the continuous (flowchart) or the discrete (petri-net) process modelling methodologies. Key process control settings, namely, parameters may continuously update, by connecting to a real-time monitoring system (sensor network) or by human input, maintaining a digital shadow of the physical system. Parameters that are not monitored are dynamically estimated with the Process Modelling algorithms, creating a detailed representation of the current industrial system in each case.

Another important concept that needs to be introduced here is the concept of **Model Instance**. Model instance is an exact copy of the model in question, representing the physical system's state now of instantiation. It can be modified (change parameters values) without affecting the actual model (and the corresponding physical system). Model instances are used for scenario building, simulation and assessment and optimisation support. The model instance is of imperative importance as it is going to be used extensively during the deployment of the EnerMan platform.

The **Process Simulation and Modelling (PSM) Tool**, is the core modelling tool of the EnerMan system, employing the Process Modelling approach and assisting in building, deploying and using a Model of the industrial system considered. There are two realisations of the PSM tool.

1. The standalone PSM Tool, operating as an interactive graphical modelling environment and used for the creation, deployment and update of a Model.
2. The PSM API, used for interconnection with external AI tools, Optimization tools, Analytics tools, etc.

The integration of the process modelling with the rest of the project is intended to occur with the help of the PSM API. Both for discrete and continuous process modelling, the API will expose all the required functionality and provide the interface to achieve the interaction between different project

[1] https://www.factlog.eu/

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

modules. The integration guidelines presented here cover up to API level. In order to make clear what the API offers and how it is intended to be used in EnerMan, Table 14 summarises the most important functions required to interact with this API.

Table 14 Description of the API functions

API Functions	**Description of Function**
Model Management	
RegisterModel (name, modelSpec)	Adds a new model in the registry. The *modelSpec* is the .xml representation of the model, as produced by the standalone PSM tool.
UpdateModel (modelID, name, modelSpec)	Updates the specified model.
RemoveModel (modelID)	Removes the specified model from the registry.
Create Instance (modelID)	Creates a new instance of the specified model. The instance is an isolated exact copy of the model that can be used for scenario simulation, optimization, etc., without affecting the model (and consequently the physical system).
RemoveInstance (instanceID)	Removes the specified instance from the registry.
System Parameters	
GetParameter (instanceID, symbol)	Returns the current value of the specified model parameter.
GetParameters (instanceID)	Returns a collection of all model (system wide) parameters, including their values.
SetParameter (instanceID, symbol, value)	Updates the value to the specified parameter.
SetParameters (instanceID, parameters)	Bulk updates the values of all process parameters.
Calculations	
Calculate (instanceID)	Performs a recalculation of the specified model instance. Returns a structure of model results, including all parameters.
CalculateUnit (instanceID, unit)	Performs a local recalculation of the specified process. Returns a structure of the specified process results only.

These functions expose the functionalities of the Process Modelling and Simulation (PMS) tool through the API built to serve the EnerMan project. As it is clear from **Table 14**, the API described is useful for every part of the lifecycle; starting from the deployment of the model created in the standalone PMS tool up to the end-of-life of the model with the model removal. With the help of Figure 10 we can identify the connections between the API functions and the steps on the lifecycle of a model from Deployment through Scenario Assessment and Optimization to System Update and eventually to the end-of-life.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

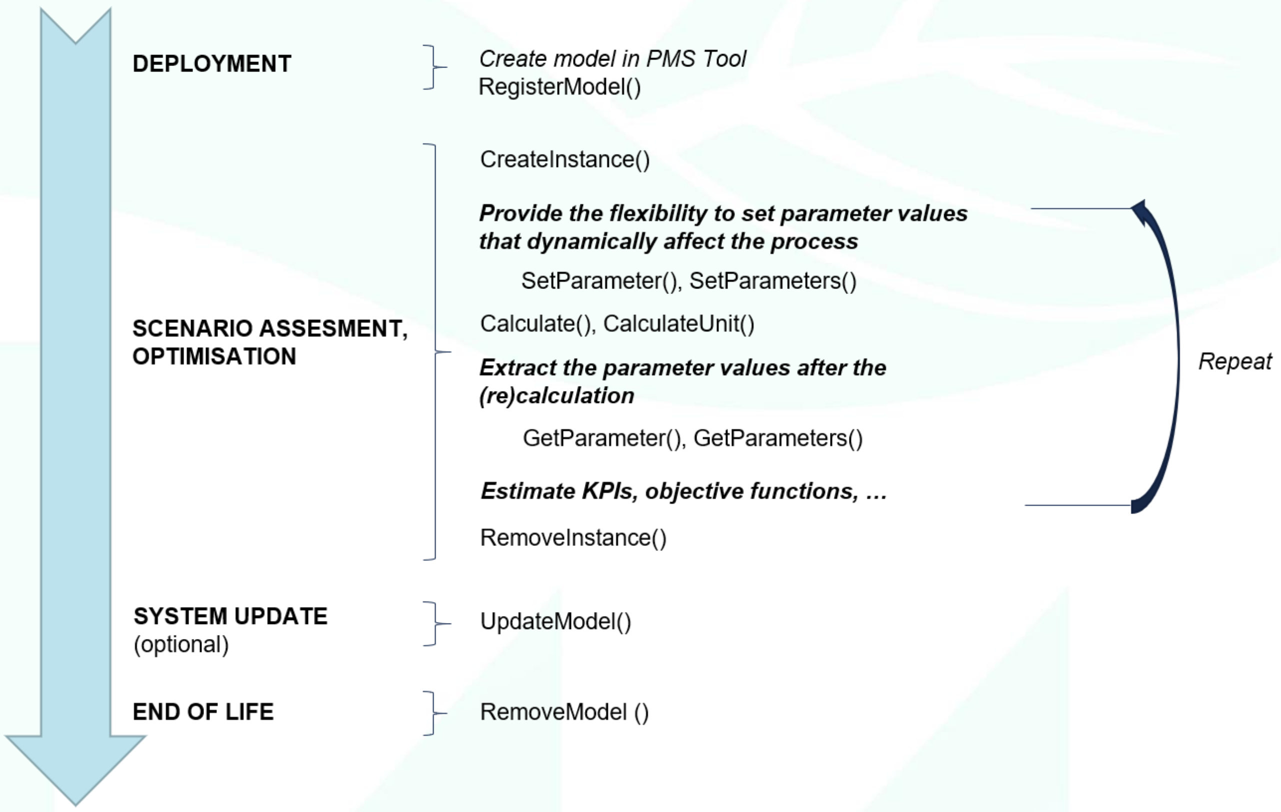

DEPLOYMENT
Create model in PMS Tool
RegisterModel()
SCENARIO ASSESMENT, OPTIMISATION
CreateInstance()
Provide the flexibility to set parameter values that dynamically affect the process
SetParameter(), SetParameters()
Calculate(), CalculateUnit()
Extract the parameter values after the (re)calculation
GetParameter(), GetParameters()
Estimate KPIs, objective functions, …
RemoveInstance()
Repeat
SYSTEM UPDATE (optional)
UpdateModel()
END OF LIFE
RemoveModel ()

Figure 10 Life Cycle of the Process Modelling Tool

4.5 Next Actions

There is a necessity to establish a protocol (including testing procedures, timeframe and end users' communication) for passing components from testing to production. This should be pointed out to run on the default daily (midnight) schedule.

A back-up methodology and frequency need to be discussed. To better find a balance between limited infrastructure capacity and backups frequency, initially back-ups should be set up manually and capability storage needs to be evaluated. Based on that info, a back-up policy will be proposed.

As a first initial approach, it is proposed daily back-ups that will be deleted after one-week, final decision is subject to the analysis. The future infrastructure needs will be estimated by monitoring the Virtual Machines (VM) performance.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

5. CONCLUSION

In this document we have presented a preliminary version of the technical specifications needed for the integration of the different components that comprise the EnerMan solution

The components of the EnerMan solution were presented, with emphasis to the information required as input of each individual step, the information delivered by each module, as well as hints on the information representation as shared between the individual components.

The goal of this deliverable is to describe the initial attempts towards an integrated EnerMan platform. In parallel, focusing on the integration of the different components, tools and applications developed by WP2, WP3 and WP4 into a single EnerMan solution, this deliverable underlines the interoperable and extensible integrated system methodology to be based on open architecture principles.

The main goal of this document is to present:

- Updates on the EnerMan modules interconnection
- Functionality enhancements driven by the analysis of the feedback received by the technical partners in the meetings held to meet additional requirements of EnerMan-related interest
- Information on the integration plan and the testing methodology to be followed

Since this is the first deliverable presenting the attempts towards the v0 of the integrated EnerMan platform, two use case scenarios which are the overview of Industrial Management System and elaborated Industrial Usage from End-User Point of View, described in detail break down the entire scope of services supported. These scenarios aim to feature the role of each involved component and mainly to demonstrate how these components interact with each other.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

6. APPENDIX

Hereinafter the Technical Questionnaire is presented, as it was shared among the partners participating in WP5 meetings.

WP5 – Technical Questionnaire

1. Questions about your component

1.1. Please name your component and provide a short description of functionality you plan to deliver in the project.

1.2. Does your component support / require user interaction? If yes, please provide a short description of the expected user input.

1.3. Will the component be provided as a service, binary or source code?

1.4. Is there any publicly available demo of the component (URL, GitHub repository etc.)?

2. Development Needs

2.1. Which are the hardware requirements for your component to run (i.e., any restrictions in memory use, required CPU, etc.)?

2.2. Which are the software requirements for your component to run (i.e., operating system, servers like tomcat or Apache, etc.)

2.3. Does your component have any database requirements? If yes, please provide the following:

i. required design (i.e., SQL-like, NoSQL):

ii. database server & version:

3. Interoperability Requirements

3.1. Will your component offer an API to work as a service?

3.2. Is your component (or can it be) containerized (e.g., be provided as docker image)?

3.3. Which are the foreseen dependencies / interactions with other EnerMan software components, if any?

3.4. What are the expected inputs/outputs of the component and in what format (provide samples if available)

4. Additional Information

4.1. Is there any other relevant information required for the operation of the module or its integration to the platform?

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

Energy Efficient Manufacturing System Management

enerman-H2020.eu

EnerMan

enermanh2020

enermanh2020

enermanh2020

HORIZON 2020
This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement **No 958478**

EnerMan