WP4 – DIGITALIZATION OF OPERATIONAL PROCESSES, SMART SENSORS CONFIGURATION AND DSS IMPLEMENTATION

T 4.4 – DSS Engine

Person responsible / Author:	Dimitris Ntalaperas (SINGULAR), Danai Vergeti (SINGULAR), George Vafeiadis (SINGULAR)
Deliverable No.:	4.4
Work Package No.:	WP4
Date:	31.12.2019
Project No.:	760792
Classification:	Public
File name:	FENIX_D4.4 report_v1
Number of pages:	14

Status of deliverable

Action	By	Date (dd.mm.yyyy)
Submitted (author(s))	Dimitris Ntalaperas (SINGULAR), Danai Vergeti (SINGULAR), George Vafeiadis (SINGULAR)	31.12.2019
Responsible (WP Leader)	Dimitris Ntalaperas (SINGULAR)	

Revision History

Date (dd.mm.yyyy)	Revision version	Author	Comments
2.12.2019	v0.1	Dimitris Ntalaperas (SINGULAR), Danai Vergeti (SINGULAR)	Definition of Table of Contents
6.12.2019	v0.3	Dimitris Ntalaperas (SINGULAR), Danai Vergeti (SINGULAR)	Contribution to section 1, 2
13.12.2019	v0.6	Dimitris Ntalaperas (SINGULAR), Danai Vergeti (SINGULAR)George Vafeiadis (SINGULAR)	Contribution to section 3, 4
20.12.2019	v0.8	Dimitris Ntalaperas (SINGULAR), Danai Vergeti (SINGULAR)	Contribution to section 5, 6

Author(s) contact information

Name	Organisation	E-mail	Tel
Dimitris Ntalaperas	SINGULAR	dimitris.ntalaperas@gmail.com	+30 6944029042
Danai Vergeti	SINGULAR	vergetid@gmail.com	+30 6974074051
George Vafeiadis	SINGULAR	Vafeiadis.giorgos@gmail.com	+30 6947310978

ABSTRACT

The current document reports the development of the mechanism responsible for assisting the end user in decision support by providing a set of recommendations based on the current status of the system under study, implementing a DSS engine. The stream data as well as the legacy data, originating from sensors or other external devices and relational databases or files containing structured data are collected by the sensors networking and data fusion mechanisms, processed and stored to a times series database which can be queried by the DSS to analyse patterns or identify outliers that can generate alerts. The DSS Engine is a rule engine based on Drools and produces alerts and recommendations by combining real time data and results from the DSS Analytics Services.

Table of Contents

INTRODUCTION **6**

1. REQUIREMENTS ENGINEERING **6**

1.1. FUNCTIONAL REQUIREMENTS 6

1.2. NON-FUNCTIONAL REQUIREMENTS 8

1.3. TECHNICAL REQUIREMENTS 8

2. CONCEPTUAL ARCHITECTURE OF THE FENIX DSS PLATFORM **9**

3. ARCHITECTURE OF DSS ENGINE **10**

4. USED TECHNOLOGIES **11**

4.1. RECOMMENDATION ENGINE 11

4.2. SYSTEM INTEGRATION AND SOFTWARE MAINTENANCE 12

5. DSS ENGINE PROTOTYPE **13**

6. CONCLUSION **14**

Figures

Figure 1: Conceptual Architecture 9

Figure 2: Reference Architecture of the DSS Engine 10

Figure 3: Reference Architecture of the DSS Engine 12

Figure 3: DSS Engine – Alerts inbox 13

Figure 4: DSS Engine – Recommendations templates 14

Figure 4: DSS Engine – Recommendations Rules 14

Abbreviations and Acronyms:	
DSS	Decision Support System
JSON	JavaScript Object Notation
REST	REpresentational State Transfer

INTRODUCTION

Deliverable 4.4: DSS Engine documents the design and the implementation of the DSS Engine of the Fenix Decision Support System. The document is structured as follows:

- Section 1 presents the requirements engineering which provided the technical requirements and driving the design of the DSS Engine.
- Section 2 presents the final version of the Fenix conceptual architecture.
- Section 3 analyses the reference architecture of the DSS Engine and explains the main workflow of the module.
- Section 4 presents the used technologies for the implementation of the DSS Engine.
- Section 5 offers a visual representation of the DSS Engine prototype.
- Section 6 finally, gives the main conclusions of the present work.

1. REQUIREMENTS ENGINEERING

The current section analyses the business requirements of WP1, shows how they are mapped to technical requirements and, how the last are reflected to the current architecture of the DSS Engine. The core functional requirements as well as the non-functional requirements, and the outcomes of D4.1, D4.2, as well as, the outcomes of WP3 have been further elaborated and defined a new set of technical requirements addressed by the DSS Engine. Finally, each technical requirement is mapped to the relevant subcomponent of the architecture of the DSS Engine.

1.1. Functional Requirements

In D4.2 an analytic presentation of the functional requirements took place for the Fenix DSS Platform, including the functional requirements related to the DSS Engine. Table 1 presents the updated core functional requirements related to the DSS Engine, as they are derived from the analysis of the WP1 outcomes, as well as, T4.1, T4.2, T4.3 and WP3.

Table 1: Functional Requirements

ID	Need	Description	Component	Input/Use case
FR8	Estimation of energy consumption	FENIX DSS Platform should assist in the estimation of energy consumption	DSS Data Fusion, DSS Engine	D1.2
FR9	Estimation of recyclability index	FENIX DSS Platform should assist in the estimation of the material recyclability	DSS Analytics, DSS Engine	D1.2 (recyclability index)
FR10	Estimation of circularity indicators	FENIX DSS Platform should assist in the estimation of the circularity indicators	DSS Analytics, DSS Engine	D1.2 (circularity indicators)
FR11	Product Requirements definition	FENIX DSS Platform should provide the ability to define the product requirements from customer's side	DSS Engine	UC1

FR12	Correlation models definition	FENIX DSS Platform should provide the ability to define rules and correlations between the recycled raw material characteristics to final dowser properties	DSS Engine	UC1
FR13	Real time user validation support	FENIX DSS Platform should provide the ability to define rules based on user validation and assessment input of the business models	DSS Engine	UC3
FR14	Health and safety assessment	FENIX DSS Platform should provide the ability to support decision support for health and safety assessment	DSS Analytics, DSS Engine	UC3
FR15	Provision of performance-based models and metrics about regarding efficiency and sustainability	FENIX DSS Platform should provide performance-based models and metrics about regarding efficiency and sustainability	DSS Analytics	WP1, all use cases
FR16	Provision of patterns recognition techniques	FENIX DSS Platform should provide patterns recognition techniques for recognizing specific trends	DSS Analytics	WP1, all use cases
FR17	Provision of algorithms for statistical analysis, trends and forecasting analysis, classification algorithms etc.	FENIX DSS Platform should provide algorithms for statistical analysis, trends and forecasting analysis, classification algorithms etc.	DSS Analytics	WP1, all use cases
FR18	Alerting and signaling mechanisms for patterns analysis and outliers' identification	FENIX DSS Platform should provide alerting and signaling mechanisms for patterns analysis and outliers identification	DSS Analytics, DSS Engine	WP1, all use cases, T4.3
FRDSS1	Alerting and signaling mechanisms based on forecasting using time series data	FENIX DSS Platform should provide alerting and signaling mechanisms based on forecasting using time series data	DSS Analytics, DSS Engine	WP1, all use cases, T4.3

1.2. Non-Functional Requirements

The non-functional requirements of the Fenix DSS Platform were presented in D4.2 in section 1.2. The non-functional requirements of the DSS Engine are covered by the list which is already defined in D4.2. For further information, please, refer to the relevant section of the deliverable.

1.3. Technical Requirements

The technical requirements were driven from the list of functional and non-functional requirements as defined in the previous section and with the mission to utilize the goals and expectations of the users and the stakeholders.

The following list defines the technical requirements that will pave the way for the design and the implementation of the DSS Engine. Each technical requirement is mapped to the relevant functional and non-functional requirements, as well as, to one or more architectural sub-components of the DSS Engine to assure that all technical requirements are covered by the component architecture (Figure 1).

Table 2: Technical Requirements

ID	Functional, Non-functional	Need	Description	Sub-component
TRDSS1	FR8-FR10, FR12, FR14, FR15, FR13, FR18, FRDSS1	Assistance for decision making	FENIX DSS Platform should be able to provide alerting mechanisms	Drools engine, Rules Management
TRDSS2	FR13, FR18, FRDSS1	Signaling mechanisms for forecasting	FENIX DSS Platform should be able to receive updates from the analytics engine	Context Listener
TRDSS3	FR13, FR18, FRDSS1	Signaling mechanisms	FENIX DSS Platform should be able to receive updates from the FENIX data fusion layer	Context Listener
TRDSS4	FR8-FR10, FR12, FR14, FR15, FR13, FR18, FRDSS1	Reliability and smooth integration	FENIX DSS Platform should be able to provide reliability of the alerting mechanism	Context Listener, REST API

2. CONCEPTUAL ARCHITECTURE OF THE FENIX DSS PLATFORM

The final version of the conceptual architecture of the FENIX DSS Platform is provided in D4.3 which is provided in Figure 1, for reference. In summary, the architecture consists of four layers; the persistency layers for storage, the middle layer that provides interfaces and core services between storage and higher-level functionalities, the enterprise layer which implements and exposes the main services implementing core business logic and, finally, the presentation layer that integrates all services and provides graphic elements for end users.

For details regarding the functionality of each layer please refer to D4.2.

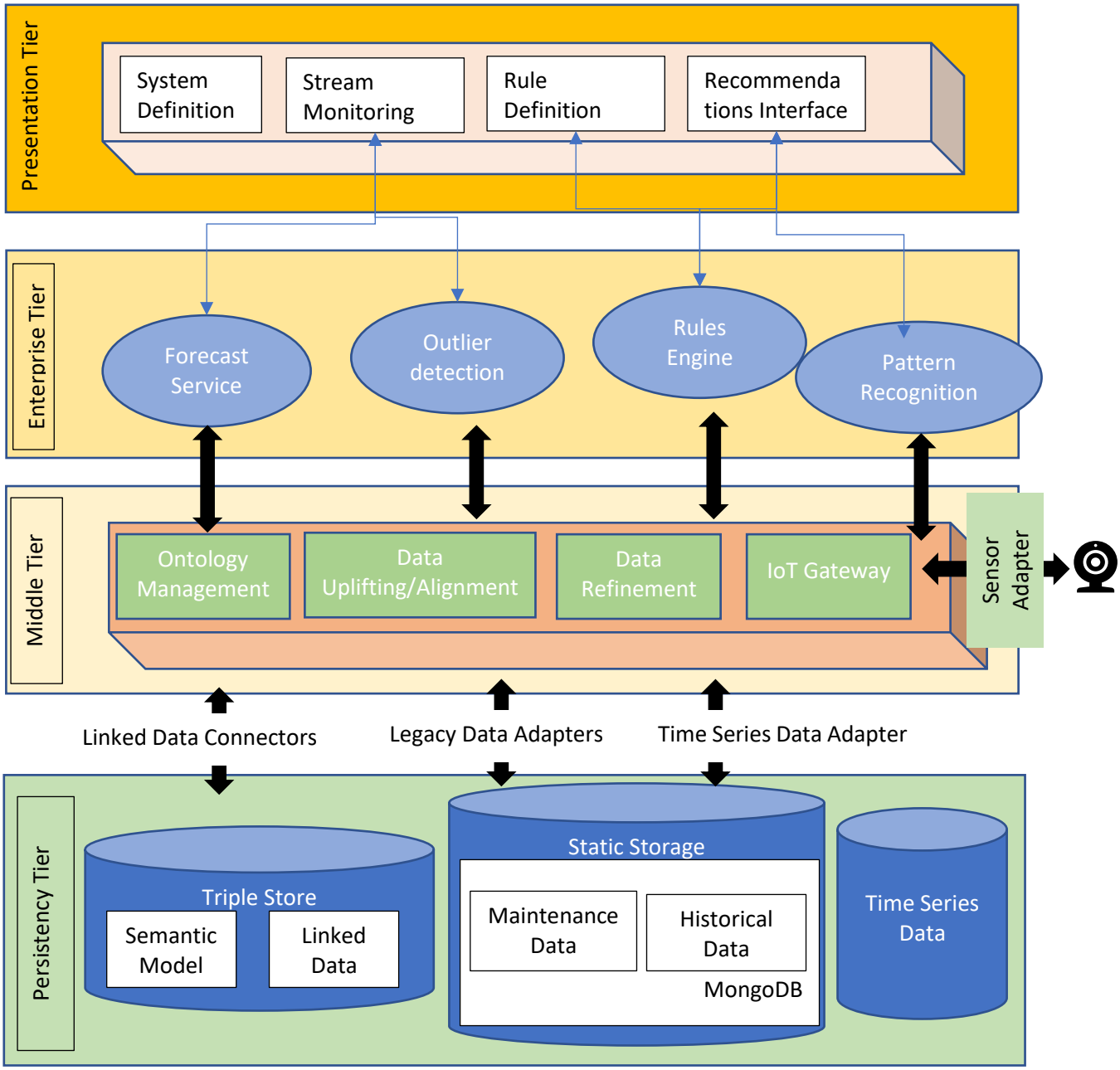

Figure 1: Conceptual Architecture

3. ARCHITECTURE OF DSS ENGINE

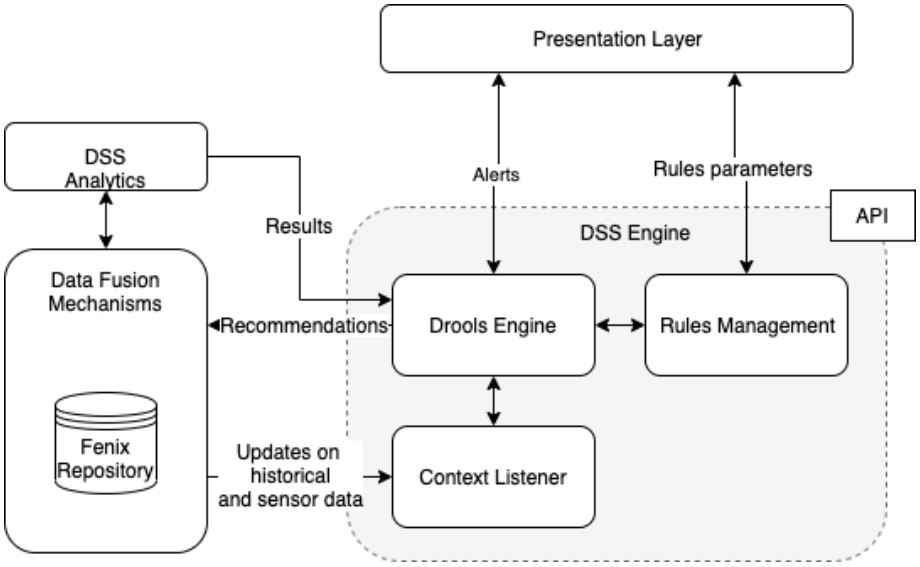

Figure 2 depicts the reference architecture of the DSS Engine. The DSS Engine consists of three core components:

- The **Drools Engine**: Fires and executes the rules in order to support decision making.
- The **Rules Management**: Provides an API which is used by the Presentation Layer in order for the definition of the rules that need to be executed by the Drools Engine. This rule set can be defined by the expert.
- The **Context Listener**: Connects with the Data Fusion Mechanisms layer and identifies any updates on real time and historical data.

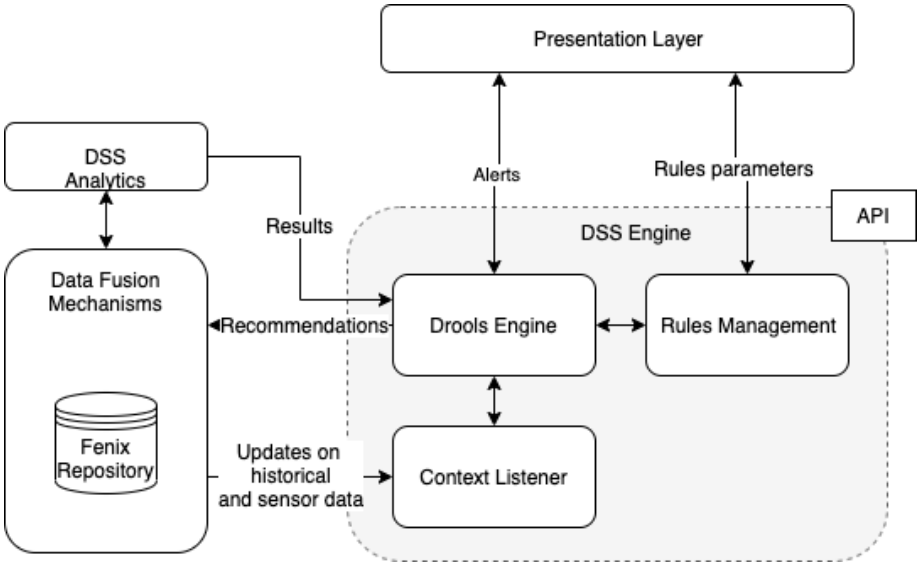

Figure 2: Reference Architecture of the DSS Engine

The Context Listener identifies updates on the real time and historical data which are provided to the Drools Engine. These updates may also include results of the Fenix Analytics Engine (outliers, values above specific thresholds, forecasts etc.). The Drools Engine processes the data retrieved by the Context Listener, as well as, by the Analytics Engine and fires the relevant rule from a set of rules, provided by the Rules Management component. Whenever a rule is executed a set of recommendations is generated which is stored into the Fenix Repository and an alert is provided to the end user, through the Presentation layer.

4. USED TECHNOLOGIES

The FENIX DSS platform is a spring boot[1] web-based application that aims to provide the decision support mechanism responsible for providing a set of recommendations based on the current status of the system under study. It is hosted on GitLab[2], a git version control repository server that enables development teams to collaborate, review and manage code. The platform consists of the recommendation engine, which is specified in 4.1, as well as a specific software stack, which is required by the deployment environment and is composed of Git[3], Docker[4], Docker Compose[5], Java SE JDK 8[6] and Maven[7]. Finally, the system integration methodology during the development phase is also presented in 4.2.

4.1. Recommendation Engine

With regards to the recommendation engine, the DSS Engine implementation relies on Drools[8], which is a Business Rules Management System (BRMS) solution. Drools is based on Java[9] and thus the implementation is fully consistent with the overall FENIX software stack. The main criterion for the selection of Drools for the implementation of the DSS Engine is the high performance and scalability that can be achieved. Rules are easier to understand for the administrator and are not hard-coded at the code part. Each FENIX recommender engine installation may be triggered based on different rules that can be graphically defined thought a unified user interface. Reasoning mechanisms, as extracted by the FENIX Ontology, are also incorporated for supporting the required reasoning functionalities within FENIX. The set of produced recommendations are also stored in the relevant MongoDB collection and made available to the set of FENIX services for further usage (e.g.

[1] https://spring.io/projects/spring-boot

[2] https://gitlab.com/

[3] https://git-scm.com/

[4] https://www.docker.com/

[5] https://docs.docker.com/compose/

[6] https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

[7] https://maven.apache.org/index.html

[8] https://www.drools.org/

[9] https://en.wikipedia.org/wiki/Java_(software_platform)

view history of recommendations, provide feedback on a recommendation). The services provided by the DSS Engine are implemented as micro-services[10], in order to provide smooth integration, scalability and easier maintenance. The supported REST API's[11] of the DSS Engine are provided in https://gitlab.com/fenixeu/fenix.

4.2. System Integration and Software Maintenance

The implementation of the FENIX DSS engine is based on the continuous integration methodology, which will be reflected into the development stages of the platform in order to ensure the adoption of an agile methodology which relies on multiple software releases during the development phase. Taking into account this approach and considering several software development aspects like code maintenance, continuous integration and delivery, quality control and bug resolution, the technology stack is composed of the concepts that has been introduced by GitLab CI/CD tool[12], as shown below.

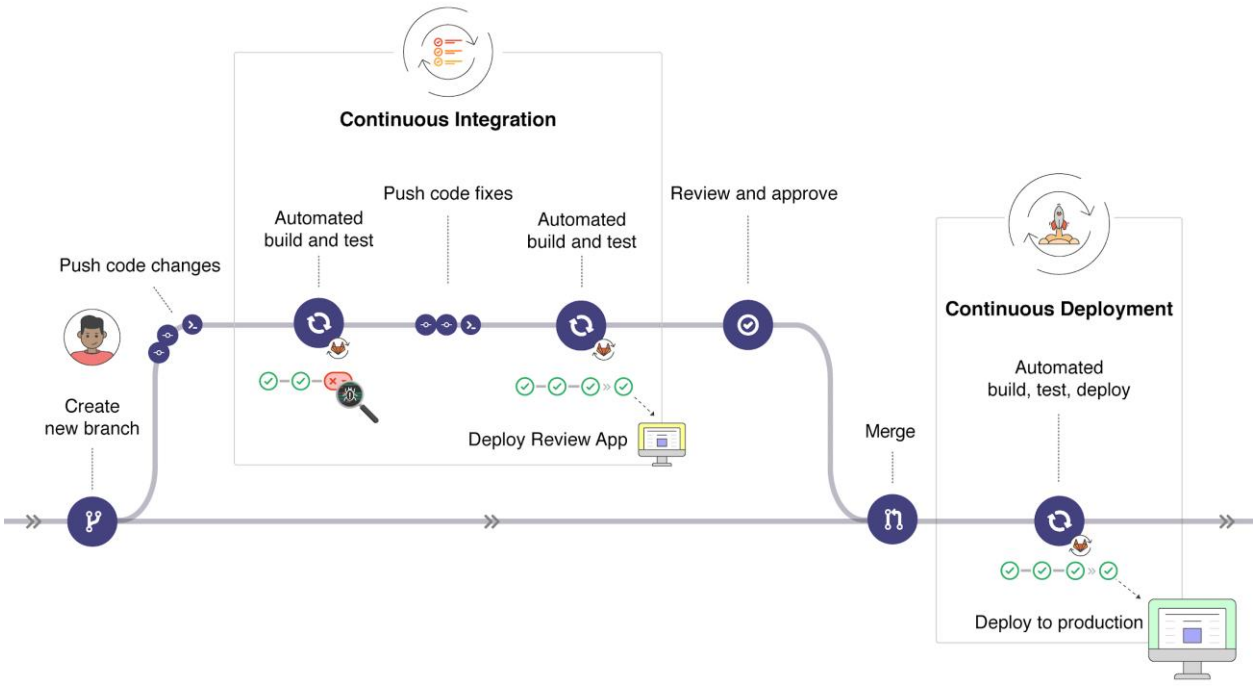

Figure 3: Reference Architecture of the DSS Engine

[10] https://microservices.io/

[11] https://en.wikipedia.org/wiki/Representational_state_transfer

[12] https://docs.gitlab.com/ee/ci/README.html

The GitLab CI/CD powerful tool allows developers to apply all the continuous methods (Continuous Integration[13], Delivery[14], and Deployment[15]) to the respective software. Finally, in the context of building agile software delivery pipelines, Docker is the software container platform for building and deploying which at the same time eliminates the risk of incompatibilities or version conflicts.

5. DSS ENGINE PROTOTYPE

The current section aims at offering the visual representation of the DSS Engine Prototype. The following diagrams illustrate the offered services of the DSS Engine. In summary, the DSS offers:

- Alerting capabilities in the form of messages that are displayed in the relevant panel to the user (Figure 4)
- The creation and editing of recommendation templates. These templates provide an interface to the recommendation engine from which the exact form of rules is translated and executed by the Rules Engine. Figure 5 depicts the relevant panel of the DSS platform that performs this operation.
- The Recommendation Rules panel. This gives an overview of the defined rules themselves and allows the user to enable/disable their rules (Figure 6)

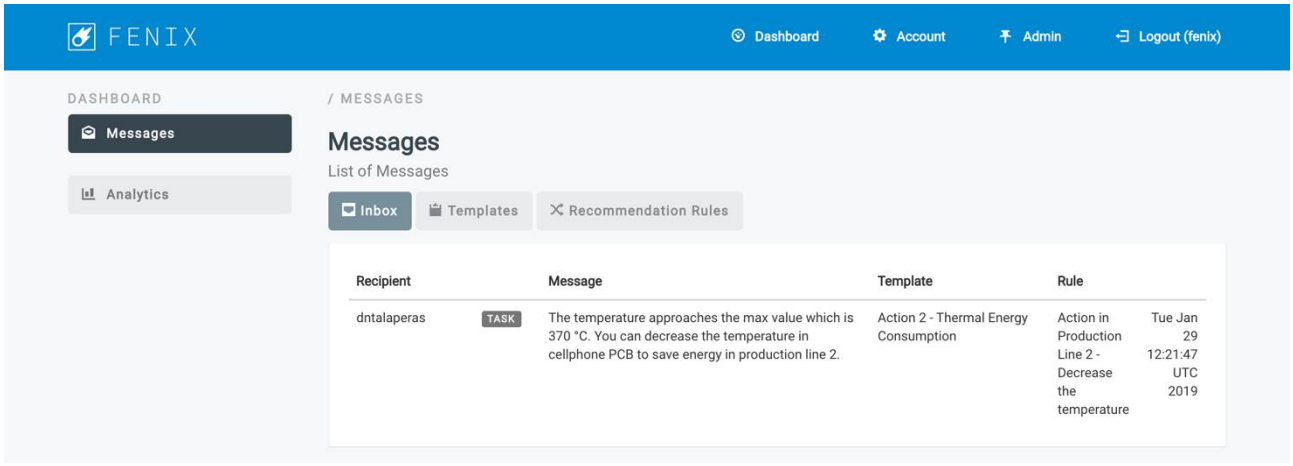

Figure 4: DSS Engine – Alerts inbox

[13] https://en.wikipedia.org/wiki/Continuous_integration

[14] https://en.wikipedia.org/wiki/Continuous_delivery

[15] https://en.wikipedia.org/wiki/Continuous_deployment

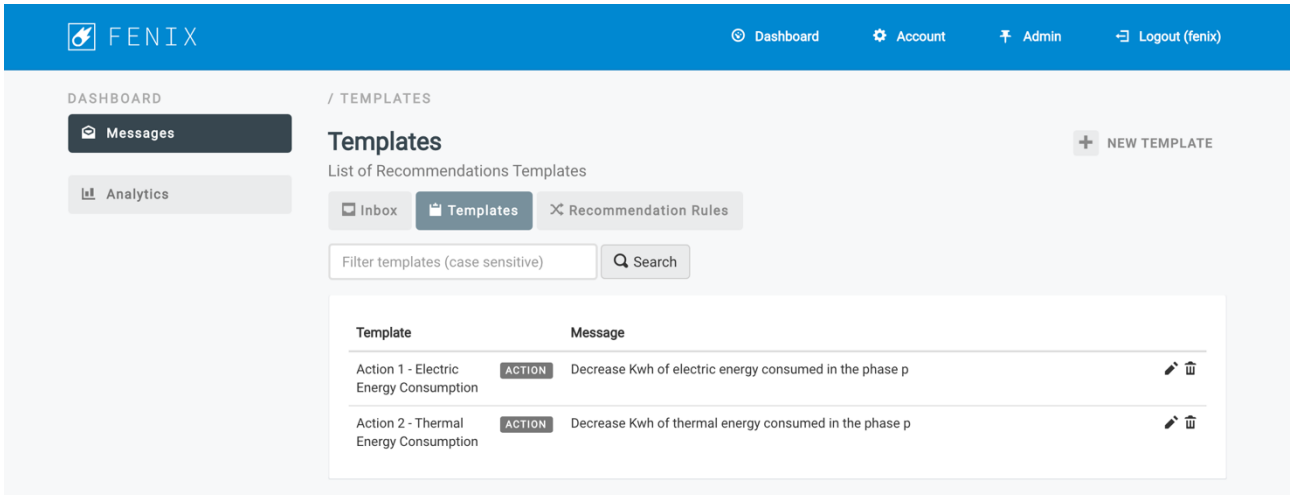

Figure 5: DSS Engine – Recommendations templates

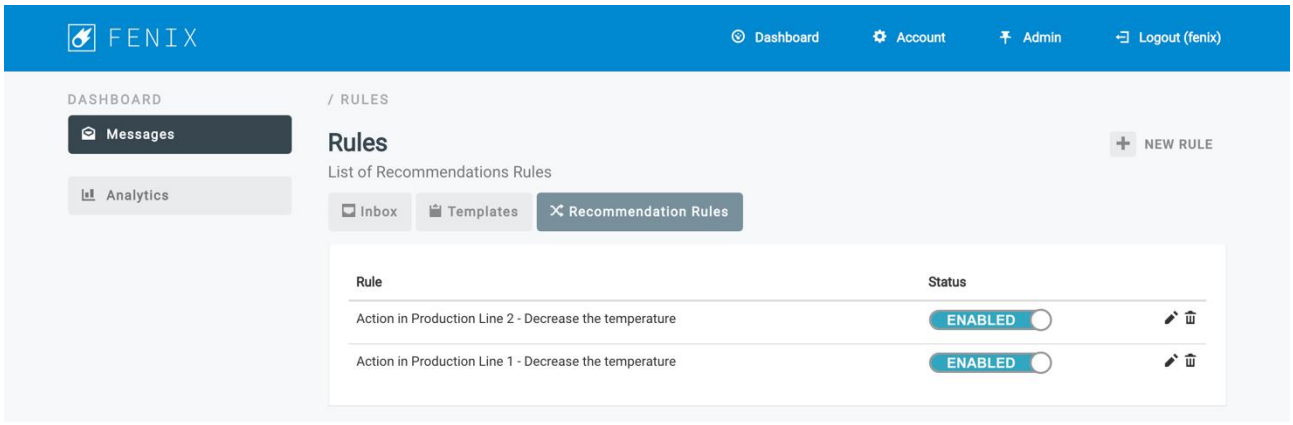

Figure 6: DSS Engine – Recommendations Rules

Furthermore, an analytics panel is also implemented as a template. This panel is to include the results of analytics that were defined in D4.3. These analytics provide both JSON and graphical output that can be embedded to the DSS. However, since the presentation of results depends upon the actual results that are going to be retrieved in the course of WP6 activities, the graphical output of analytics has not been integrated yet; it will be carried through during WP6 form input received both by the results and by the pilots.

6. CONCLUSION

The present document presented the design and the implementation of the Fenix DSS Engine-based on the functional requirements of WP1, WP2, WP3 and WP4. The DSS Engine exploits real time, historical data and results from the DSS Analytics and is ready for deployment and evaluation, as soon as the pilot activities of WP6 start to provide results. The DSS Engine is implemented using the Drools Engine which offers high performance and scalability. Moreover, the services of the DSS Engine are implemented as micro-services, in order to provide smooth integration, scalability and easier maintenance.